AI Magazine Volume 25 Number 2 (2004) (© AAAI)

Unrestricted Recognition of 3D Objects for Robotics Using Multilevel Triplet Invariants

Gösta H. Granlund and Anders Moe

■ A method for unrestricted recognition of three-dimensional objects was developed. By *unrestricted,* we imply that the recognition will be done independently of object position, scale, orientation, and pose against a structured background. It does not assume any preceding segmentation or 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 feedback learning procedure, percepts are mapped on system states corresponding to manipulation parameters of the object. The method uses a learning architecture with channel information representation. This article discusses how objects can be represented. We propose a structure to deal with object and contextual properties in a transparent manner.

Recognition of objects is the most fundamental mechanism of vision. By *object,* we mean some entity that is discernible from other entities, which includes the everyday notion of an object as something you can hold in your hand; however, it goes far beyond that. A certain object can contain parts, which, for convenience sake, we also want to denote as objects. Objects can consequently group to build up entities that we assign certain properties, and for that reason, we want to give a collective notion of object.

An extension of the discussion along these lines has the consequence that nearly anything can be viewed as an object—a line, a wheel, a bicycle, a room, a house, a landscape, and so on. This level transparency is probably inevitable as well as practical. The segmentation of the external world varies depending on the scale at which it is observed and what aspects are at the focus of attention.

Still, we have the idea that objects should be possible to sort into categories. The difficulty with the generation of a taxonomy is the ambition to obtain a conceptually manageable and simplified structure for something that is in reality a complex, multidimensional structure. In practice, there are for a given context, certain aspects that are more crucial than others, which will determine to what category we finally assign an object.

This irreducible multidimensional characteristic of objects determines how we have to approach the issue of what is one object and what is another: There are simply different ways to define an object, depending on the setting.

Object-Centered versus View-Centered Representation

Over the years, there has been increasing interest in research on invariants (Jacobsson and

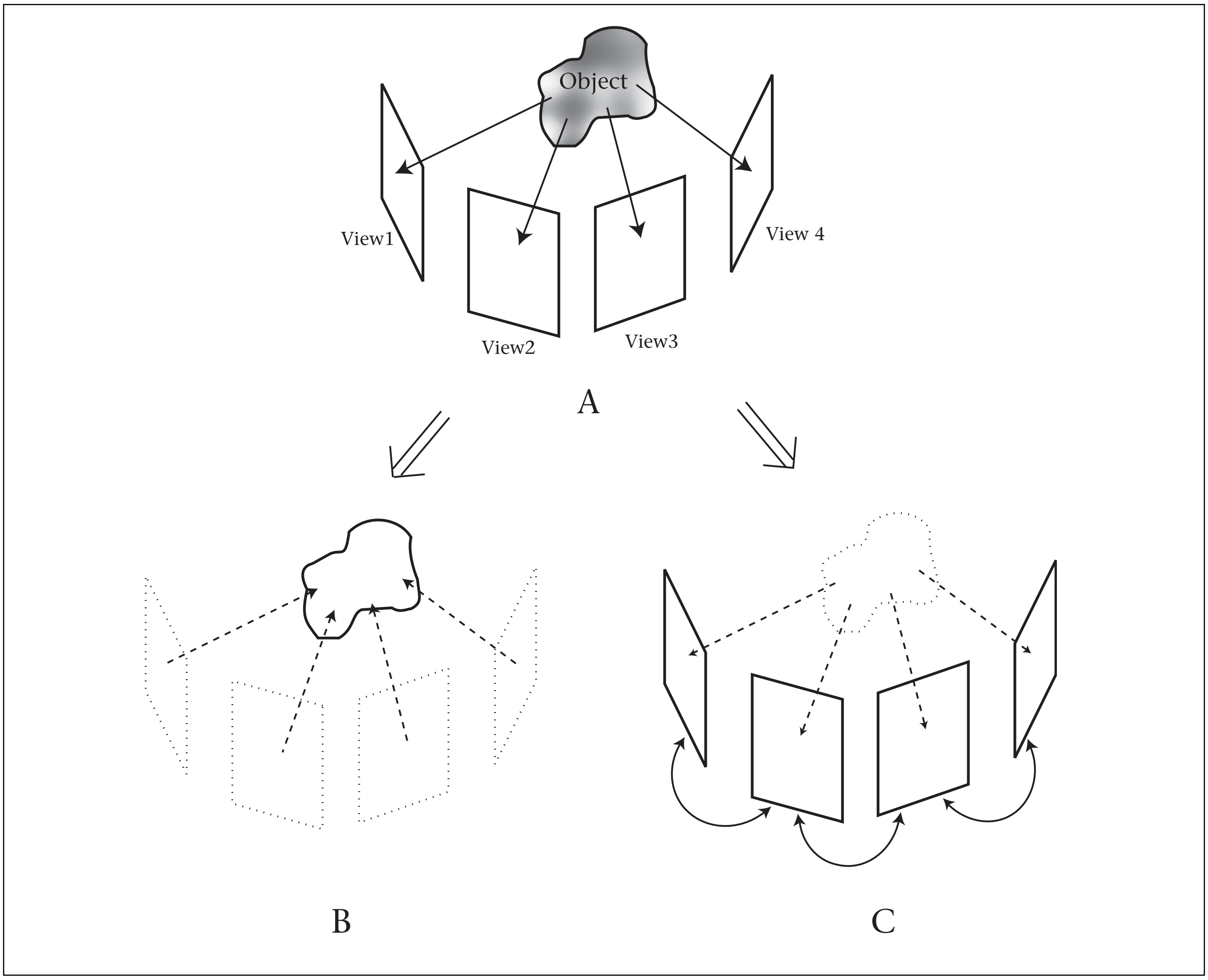


Figure 1. Object-Centered and View-Centered Representation of an Object.

A. Measurements produce information about different views or aspects of an object. B. Object-centered representation: The views are used to reconstruct a closed-form object representation. C. View-centered representation: The views are retained as entities that, linked together, form a representation of the object.

Wechsler 1982; Kanatani 1987; Koenderink and van Doorn 1975; Mundy and Lisserman 1992). Most of the methods proposed treat invariants as geometric properties, the rules for which should be input into the system. Theoretical investigation of invariance mechanisms is undoubtedly an important task because it gives clues to possibilities and limitations. It is not likely, however, that more complex invariants can be programmed into a system. The implementation of such invariance mechanisms in systems will have to be made through learning.

An important application of invariant representation is object description. To position ourselves for a thorough analysis, we look at two traditional major lines of approach that have been used for object description: (1) object-centered representation and (2) view-centered representation (Granlund 1999b; Riesenhuber and Poggio 2000a, 2000b) (figure 1).

From the real object, a number of measurements or projections are generated. They can be images of the object taken from different angles (figure 1a). From these measurements, we can proceed along one of two different tracks.

One of the tracks leads to the object-centered representation that combines these measurement views into some closed-form mathematical object (Grimson 1990) (figure 1b).

The image appearance of an instance of a particular orientation of the object is then obtained using separate projection mappings.

A view-centered representation, however, combines a set of appearances of an object, without trying to make any closed-form representation (Beymer and Poggio 1996; Poggio and Edelman 1990; Ullman and Basri 1991) (figure 1c).

Object-Centered Representation

The basic motive of the *object-centered representation* is to produce a representation that is as compact and as invariant as possible. It generally produces a closed-form representation, which can subsequently be subjected to interpretation. Thus, no unnecessary information is included about details on how the information was derived. A central idea is that matching to a reference should be easier because the object description has no viewpoint-dependent properties. A particular view or appearance of the object can be generated using appropriate projection methods.

We can view the compact invariant representation of orientation as vectors and tensors (Granlund and Knutsson 1995) as a simple variety of object-centered representation. Over a window of a data set, a set of filters are applied, producing a component vector of a certain dimensionality. The components of the vector tend to be correlated for phenomena of interest, which means that they span a lower-dimensional subspace. The components can consequently be mapped into some mathematical object of a lower dimensionality to produce a more compact and invariant representation, that is, a vector or a tensor (Granlund and Knutsson 1995).

A drawback of the object-centered representation is that it requires a preconceived notion about the object to ultimately be found, its mathematical and representational structure, and the way in which the observed percepts should be integrated to support the hypothesis of the postulated object. It requires that the expected types of relations are predefined and already existing in the system and that an external system keeps track of the development of the system, such as the allocation of storage and the labeling of information. Such a preconceived structure is not well suited for self-organization and learning. It requires an external entity that can "observe labels and structure" and take action on this observation. It is a more classical declarative representation rather than a procedural representation.

View-Centered Representation

In a *view-centered representation*, no attempt is made to generalize the representation of the entire object into some closed form. The different parts are kept separate but linked together using the states or responses, which correspond to or generate the particular views. The result is a representation that is not nearly as compact or invariant. However, it tells what the state of the system is associated to a particular percept state. A view-centered representation in addition has the advantage of being potentially self-organizing. This property will be shown to be crucial for the development of a learning percept-action structure. There are indications from perceptual experiments that the view-centered representation is the one used in biological visual systems (Riesenhuber and Poggio 2000a, 2000b).

An important reason for the view representation is that it allows an interpretation rather than a geometric description of an object that we want to deal with. By interpretation, we denote links to actions that are related to the object and information about how the object transforms under the actions.

Combination of Representation Properties

An object-centered representation is, by design, as invariant as possible with respect to contextual specificities. It has the stated advantage of being independent of the observation angle, distance, and so on. This independence has, however, the consequence of cutting off all links that it has to specific contexts or response procedures that are related to that context or view.

The generation of an invariant representation implies discarding information that is essential for the system to act using the information.

It is postulated that we can represent objects as invariant combinations of percepts and responses, suggesting that we will start out from the view-centered representation of objects.

The structure that results from the preceding model will be of type frames-within-frames, where individual transformations of separate objects are necessary within a larger scene frame. See figure 2 for an intuitive illustration of this frames-within-frames as a tree. The ability to handle local transformations is absolutely necessary and would not be possible with a truly iconic view representation. It is postulated that the frames-within-frames partitioning is isomorphic with the response map structure. In this way, the response map "reaches" into the frame in question to implement the percept-action invariance of a particular object aspect.

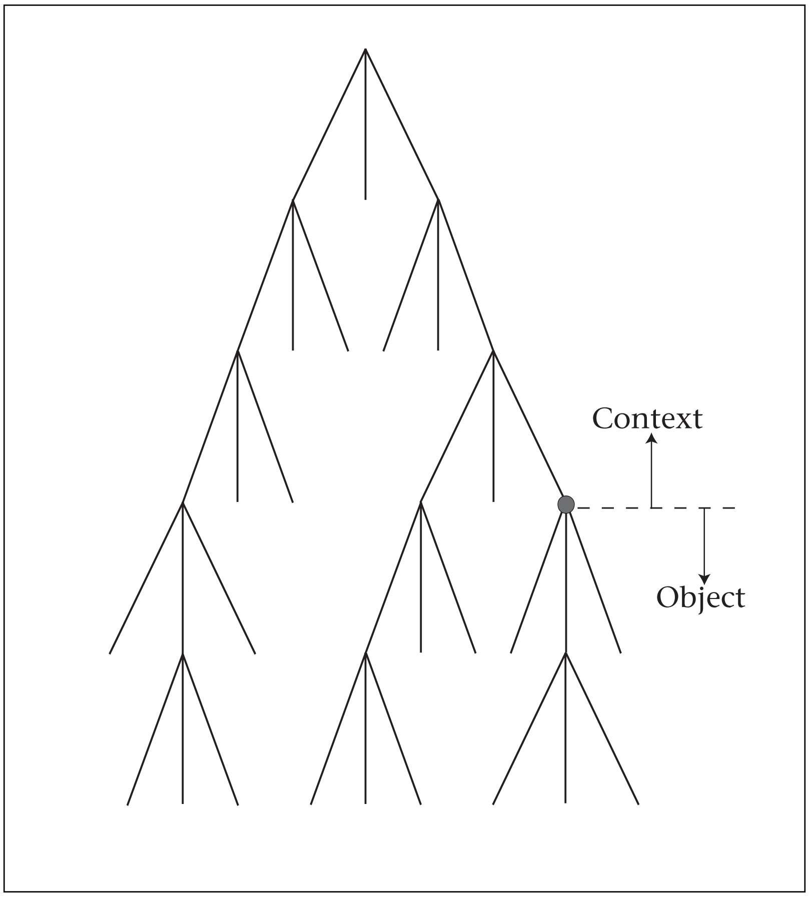


Figure 2. Tree Structure of an Object, Its Parts and Its Context.

What Constitutes an Object?

As apparent from the introduction, there is no simple definition of what constitutes an object. Like for most phenomena, the outcome of such a definition is dependent on the aspects focused on. Still, in a method for recognition of objects, there has to be some inherent strategy, which implies an operative definition for its implementation.

Traditionally in computer vision, objects have been characterized by geometric properties. In this article, we maintain that categories do not represent groups of percepts or geometric features but groups of actions and the associated contexts for these actions. These are, as we see, only partly dependent on percepts or geometric features.

It appears that an object is characterized partly by properties of its own and partly by the context in which it appears. We can intuitively visualize this characterization as different levels in a multilevel tree structure (figure 2).

What we in some situation denote as an object is represented as a node in this structure. Going downward in this structure, we find constituent parts of the object, which can be considered objects in their own right. Going upward, we find contextual properties, such as adjacent objects, background, and different aspects of setting—spatial and temporal. A head will be the usual context of the object eye. For simplicity, in figure 2, we have represented the context of an object as a single line; in reality, such a line represents a multitude of influences or signals.

This structure suggests a recursive property: At every node at every level, what is seen downward constitutes an object in a traditional sense, but what is seen upward is context.

It should be emphasized that what is represented at higher levels might not correspond to something that is termed an object in everyday language. Rather it might be a scene or some other complex situation. In addition, the everyday word object cannot be used in a consistent way in this discussion because it represents different things, and conceptually, we often include the context of an object in its categorization.

For that reason, we formulate two levels of definitions, which correspond to these two characterizations of an object: (1) level 1 (downward) and (2) level 2 (upward).

Object-Definition Level 1

An *entity* is viewed as an object, which is separate from other objects or the background if there is an action that has a separate influence on the particular object in relation to other objects or the background. Such an entity constitutes a separate object with respect to the particular action.

An example of the preceding is that the object in question can be moved in relation to other objects. Another object might require a screwdriver and a fairly complex action to transform it into two objects.

At first, it might seem peculiar that the definition of an object should partly depend on some action with which it is associated. There are three important reasons for this: First, objects become associated with particular actions possible to perform in relation to this object. This set of actions becomes the *signature* of the object in the external world. Geometrically similar objects can be associated with very different sets of actions. Second, actions are clues that can be learned by a system. Third, the stepwise learning of such transformations is the system's possibility to attach to and learn the external world to organize the action interface to it.

Thus, if the system can perform some action that makes certain parts of the visual field behave differently to other parts, then the system will consider this part as belonging to a separate object. We see later that this mechanism is an important property that allows learning.

Thus, in turn, another system might well get a different division into objects. One system might pass a shelf of books, viewing the shelf as one object, but a more educated system might know that it can pull out one of the items in the shelf, which is the object book. We must remember that a priori, the system has not a clue for the division into separate objects, unless it has already experienced them; the only possibility is to interact and explore.

This object definition leads to the situation that there will be several levels of objects or object parts, which are individually distinguishable but, in turn, are parts of more complex objects, forming a structure of several levels. An action on an object, which, in turn, acts on another object, will appear to the system as a single object, although maybe a complex, composite object with parts linked in ways that might not be established fully. Again, the system has no alternative for interpretation from its myopic view, and it might well make mistakes. From human psychology, it is well known how accidental coincidences between events can lead to erroneous associations and grave misunderstandings about how the world is related and what is termed *superstitious behavior.*

Object-Definition Level 2

The other characterizing property of objects is their context. We do not primarily deal with an object in its isolated generality but an object at a particular place and a particular context. This approach is totally consistent with the view-based representation, where we have seen that we do not represent an object as a generalization but as an object at a particular view. Contextual information has to be maintained, and a generalization of this rule is as follows:

> Objects, phenomena, and so on, are never represented in an isolated generalization but are attached to a background or context. The context consequently is an important part of the object definition.

We can give some instances of interpretation of this structure: Level 2: Object in a particular context, Level 1: Object at a particular angle or Level 2: Object at a particular position, Level 1: Object at a particular view.

An object will be characterized by what contextual links are generated and how the percept structure is affected by actions on the object. Part of the context is as well higher- level influences from what we term purpose, and so on. We do not deal with these high- level issues in this article. In the recognition process, influences from level 1 and level 2 properties will be combined. Their weighting and influence will vary from one situation to another. In certain cases, level-1 properties might be sufficient; in other cases, level-2 properties will be decisive, implying that the processing structure is both bottom up and top down. The identification of a particular object will, in return, evoke the context in which the object was learned, or the state of the system at the time. We can see that an important characterizing link of an object is the context in which the object was first observed.

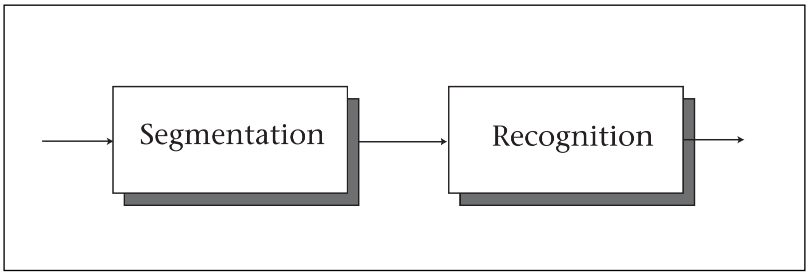


Figure 3. Classical Model for Object Identification.

The preceding discussion implies that a proper training process is not to present a system with a sequence of different isolated objects to be learned. Objects will, in the learning process, be differentiated by a context, implying some variable that describes a position, state, or response to be associated with the object in the learning process. The reference will in such a case be equivalent to "the object we had in that position." This observation state is generally an important part of the object description, although a recognition can sometimes be done in other contexts, dependent on the uniqueness of level-1 features. The recognition of an object in a wrong context is generally a difficult problem, where even humans often fail, unless the context tells us to pay special attention.

Object Identification

The classical model for object identification consists of two steps (figure 3): (1) segmentation of the object from the background and (2) recognition of the segmented object.

This approach assumes that it is possible to determine what pixels belong to a particular object, although the object is not known. In most cases of interest, this assumption is unrealistic. The approach might work in exception-

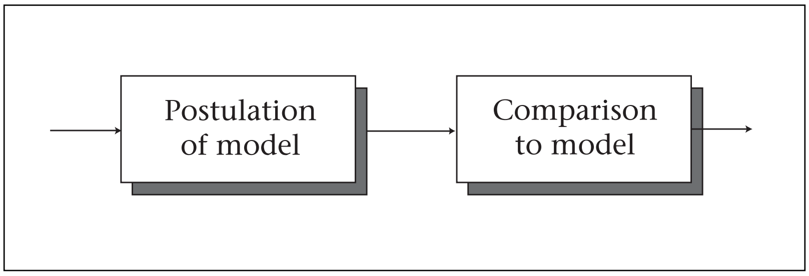


Figure 4. Two-Step Model for Object Identification.

Var	Object Characterization
ω	Object class
x	Horizontal position of object
y	Vertical position of object
ϕ	Horizontal 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.

ally simple cases, where there are globally distinguishable features, such as a distinctive color or density of an entire object, which makes it stand out from the background. Objects generally do not consist of homogeneous regions or globally distinctive features, rather they might appear toward a structured background of a similar character or mixed with other similar objects.

This condition illustrates well the fundamental inverse problem of vision: It is not possible to directly find the class membership (object category) for a particular pixel (feature). As a consequence, it is not possible to find with any confidence the pixels that belong to an object before the object has been recognized. It is necessary to somehow perform a segmentation and a recognition in the same process.

The case considered here is the recognition of a three-dimensional (3D) object given a 2D projection, such as a camera image, which gives a large variation in the appearances of just a single object. Various approaches to this problem have been explored by Besl and Jain (1985); Lowe (2001, 1999); Mamic and Bennamoun (2002); Matas, Burianek, and Kittler (2000); Murase and Nayar (1995); Mikolajczyk and Schmid (2001); Pulli and Shapiro (1996); Schiele and Pentland (1999); Schmid and Mohr (1997); Schmid, Mohr, and Bauckhage (2000); Weiss and Ray (2001); and Yuan and Niemann (2001).

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

Because object identification is an inverse problem, it is necessary that a hypothesis of structure or model is first made. Measurements performed on the actual data are then compared to the reference data belonging to the hypothesized model.

No change implies a two-step process (figure 4): (1) postulating a certain model and (2) performing measurements and comparing these with a reference under the assumption of the particular model.

We can observe the isomorphy between figures 3 and 4. The second block is identical between the two structures. The major difference is in the first block of the structures. In the classical version according to figure 3, the ambition is to hypothesize a model consisting of the entire object. As stated earlier, this strategy is not realistic.

Rather, it is necessary to devise a structure that can recognize potentially useful model fragments, which can be combined to build up increasingly complex models, representing the object in question. We can view the structure in figure 3 as fragmented into a number of structures according to figure 4.

The main issue is how to select hypothetical models that are descriptive; models that can deal with a structure of a high complexity and can be handled in an efficient computational manner.

To use learning for the acquisition of models of sufficiently complex objects, a new structure was developed using a hierarchy of partially invariant triplet primitives, describing an object in a fragmented, view-centered fashion (Granlund 1999b). This structure can be used to assign objects to a class, but it is believed that the important issue is to establish a relation between perceptual clues and states of the object and the observing system (Granlund 1999a), according to table 1. These relations can then be viewed as system states to be estimated at run time, given a set of percepts.

Characteristics of Model Structure

For a long time, it has been believed in vision research that a robotics system should have a structure as shown in figure 5. The first part

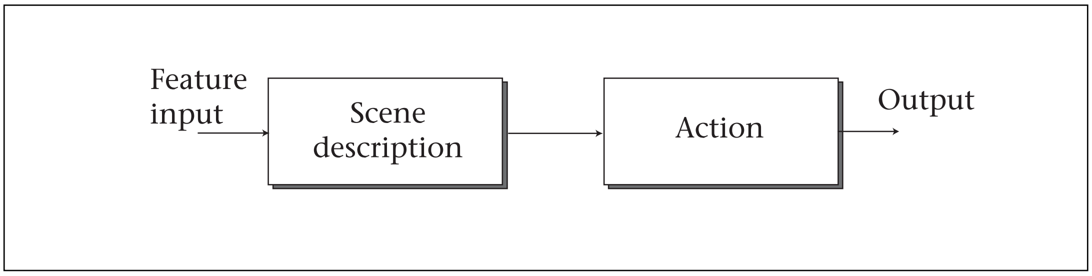


Figure 5. Classical Robotics Model.

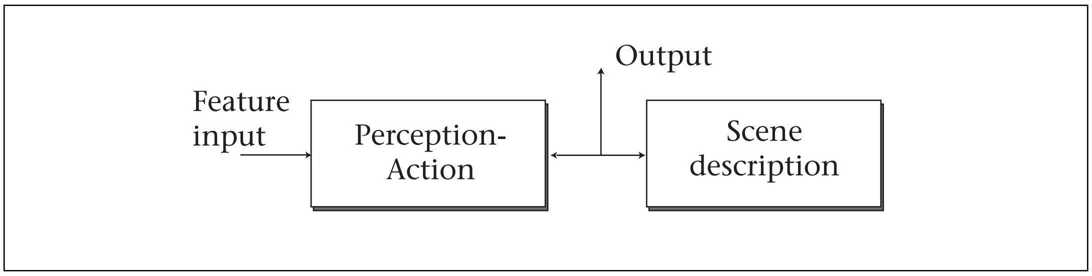


Figure 6. Perception-Action Robotics Model.

should use the incoming percepts of types edges, lines, and color to generate a description. Such a description has typically been in geometric terms and often had similarities to a computer-aided–design (CAD) representation. The goal has typically been that the model should describe the object as accurately as possible.

Given this description of the image, objects will be recognized and assigned to the proper categories, together with information about position and other relevant parameters. A second unit would then use this information to produce actions in the physical world, for example, to implement a robot.

This structure has not worked out very well for several reasons, which we deal with in subsequent sections. In brief, it is because the jump between the abstract description and the action implementation is too large. A large number of important contextual qualifiers necessary for precise action have been lost in the abstraction process for description. It turns out that it is necessary to break up the big jump between percepts and action into a sequence of small steps, where percepts and functions are step by step related to states in the external world. We see that this stepwise relation to the external world is, in addition, allowing learning or self-organization (Granlund 1999a, 1999b).

The result is that the order between the parts in figure 5 should in fact be the opposite (figure 6).

The first part of the system is a reactive percept-to-action mapper. After this mapper follows—if necessary for the application—a part that performs a symbolic processing for categorization, reasoning, and communication.

The distinctive characteristic of such a structure is that percepts are mapped directly onto system states or actions rather than descriptions. The reason is that this strategy allows the system to learn step by step how objects and other aspects of the environment relate to the system's own actions. This mapping between actions and percepts is used to subsequently invoke appropriate actions as the corresponding percepts reappear. In contrast, 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 be able to identify its position, pose, and orientation to learn what action complexes it can be linked to for manipulation. This linkage is what understanding of an object implies.

For a useful, extendable model structure, we propose a number of characteristics: First, models should be fragmentable, such that a certain model can be part of a more complex or higher-order model. A relative simplicity of individual models is required to allow a fast optimization in learning. Second, learning of models should proceed from lower levels to higher levels. Third, acquired lower-level models should be usable as parts of several different higher-order models. Fourth, a particular model is only acquired once, and its first occurrence is used as

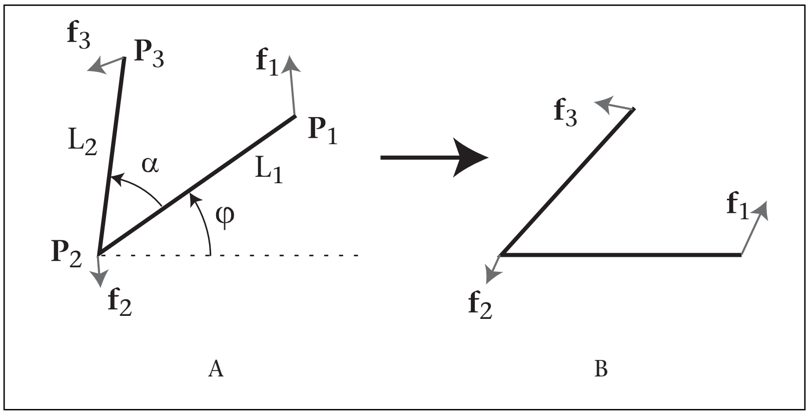


Figure 7. Triplet with Parameters Indicated.

A. Triplet in arbitrary orientation. B. Triplet rotated to normalized orientation.

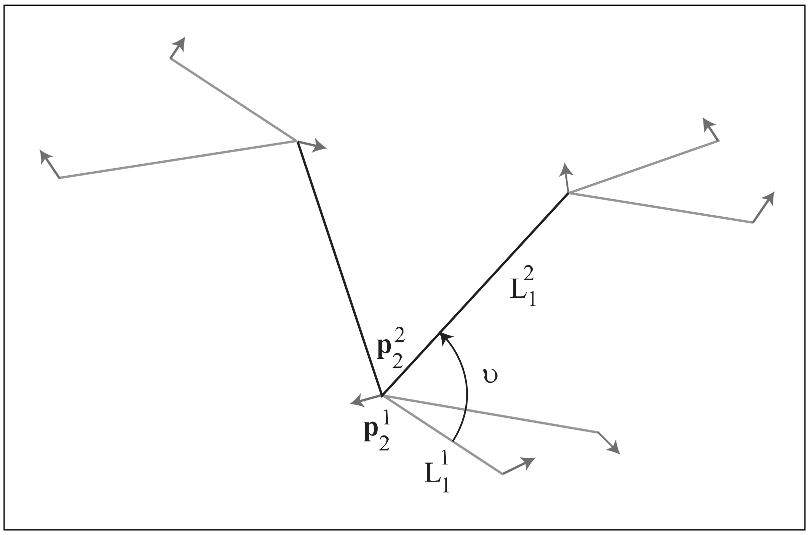


Figure 8. Two Levels of Triplets, with Relational Parameters Indicated.

the representation of that model. Fifth, a certain model can be substituted by a better model at the interface to a higher-level model.

Because of this recursive character of models, we simply denote them all as models, be it parts or combinations. At the lowest level, a model can imply the output from a feature detector.

We can see that the development of models can progress both upward (second item) and downward (fifth item). If a totally "clean slate" system experiences the external world, some of its exploratory actions will ultimately correlate sufficiently consistently with some set of percepts to form a model (moving the arm in front of the eyes). This model and others acquired can then provide parts for models to deal with more complex actions in relation to more complex perceptual structures.

The downward buildup of the model structure is done by substituting an input to a model by the output from a better model. We do not deal with this issue here because it is largely unexplored for technical systems. The correspondence in biological systems is that neural structures continually sense input and output, and as such, a unit finds that it gives a better (more consistent) mapping of a phenomenon that it takes over.

The third item is necessary for a learning process to be cumulative, and the system should be able to use earlier acquired knowledge. Otherwise, the learning of a new object would require the learning of all its constituent primitive models anew, even those that it encountered earlier.

Triplet Models

The basic model, or primitive chosen, consists of a set of three point features, $\mathbf{f}_k$, at positions $\mathbf{p}_k$, joined to form a triplet (figure 7).

Point features are vectors, representing sparse, localized properties of an image such as corners, curvature, centers of homogeneous regions, mid-points of lines, and so on.

A point feature, $\mathbf{f}_k$, can also represent an entire lower-level triplet attached, whereby multilevel triplets are formed (figure 8).

There are many ways in which a number of points can be brought into groups. The number of points should be low to limit the complexity of the models to allow optimization in learning, and the second and third items come to mind. We have in parallel developed a structure combining two points, or *duplets*, and a similar discussion can be made around that case.

The triplet structure can be viewed as a planar patch in a multidimensional space, joining the three feature points. It has some attractive properties of invariance at the same time as it ensures a certain descriptive power and degree of uniqueness. A triplet can be characterized in a number of equivalent fashions (Isaksson 2002). The components used are as illustrated in figure 7: First, it allows a unique ordering of the feature points, which is implemented such that the triplet is "right oriented," that is, the angle $\alpha < \pi$, as defined in figure 7. Second, the distance between the two feature points not connected by the triplet must be shorter than the two other distances between feature points (figure 7). The triplet structure allows us to define a scale invariant structure parameter $\gamma = \frac{L_1 - L_2}{L_1 + L_2}$.

Third, the triplet can be brought into a normal orientation by aligning leg L_1 to make $\varphi = 0$ (figures 7a and 7b).

The preceding properties make the following parameter variations trivial: (1) orientation in the image plane, (2) scale, and (3) object position in *x* and *y*. This fact reduces the dimensionality of the total system, such that the variations in these parameters can be handled without extending the training space. This independence effectively implies an invariance of the primitives with respect to these parameters.

To decrease the combinatorial complexity and improve the robustness, additional restrictions, grouping rules, and criteria for acceptance of triplets have been devised. Only three are mentioned here: (1) spatial grouping range, (2) object closure criteria, and (3) symmetry.

Spatial grouping range: We expect primitives to increase in spatial size going toward 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 adaptive generation of these thresholds 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, indicate parts of a common object or region. Tests for texture or conflicting structures can reject the hypothesis of primitive or constitute an additional descriptive feature.

Symmetry: Symmetric structures, such as parallel lines or similar regions, are grouped.

Channel Information Representation

Information representation is an important issue in general but more so if learning is to be used.

The information representation used for all parameters is a monopolar channel representation (Granlund 2000). 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.

Examples of suitable kernels for channel representations include Gaussians, B-splines (Felsberg, Forssen, and Scharr 2004), $\cos^2$, and other localized windowing functions with a shape similar to the kernels in figure 9.

In this article, channel properties are exemplified using the $\cos^2$ family of kernel functions. The output from channel *k* representing scalar *x* is

$$x^k(x) = \begin{cases} \cos^2\big(\omega d(x,k)\big) & \text{if} \quad \omega d(x,k) \le \frac{\pi}{2} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Variable *k* is the *kernel center*, ω is the *kernel width,* and *d(x, d)* is a *distance function.* For variables in linear spaces, the Euclidean distance is used:

$$d(x, k) = |x - k| \quad (2)$$

The total interval of a signal *x* can be seen as cut into a number of local but partially overlapping intervals, $d(x, k) \le \frac{\pi}{2\omega}$.

For periodic spaces with period *K,* a modular Euclidean distance is used[1]

$$d_K(x,k) = \min(\mathrm{mod}(x - k, K), \mathrm{mod}(k - x, K)) \quad (3)$$

Representation of an angle is a typical example of a variable in a periodic space.

To simplify the notation, we defined the numbers *k* as consecutive integers, directly corresponding to the indexes of consecutive kernel functions. We are obviously free to scale and translate the actual signal value in any desired way before we apply the set of kernel functions. For example, a signal value ξ can be scaled and translated in the desired way

$$x = \text{scale} \cdot (\xi - \text{translation}) \quad (4)$$

to fit the interval spanned by the set of kernel functions $\{x^k(x)\}_1^K$. Nonlinear mappings $x = f(\xi)$ are, of course, also possible, but they should be monotonous for the representation to be unambiguous.

The channel representation allows a space variant processing, implying the use and the generation of different models for different parts of the input feature space. The representation also allows the implementation and learning of nonlinear models using linear mappings. In addition, it allows the representation of multiple values of a variable. The simultaneous representation of two values of a 1D scalar variable appears in figure 9.

The channel representation represents a variable as the relation between the nonzero channel values. No change provides for a noise immunity and a possibility to make confidence statements.

It is possible to define channels in 2D and in multiple dimensions by making extensions to equation 1. A 2D version of this representation is directly available from wavelets or filter output. For a more extensive discussion, see Granlund (2000).

The monopolar property implies that data only utilize one polarity, for example, only positive values in addition to zero. This property 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

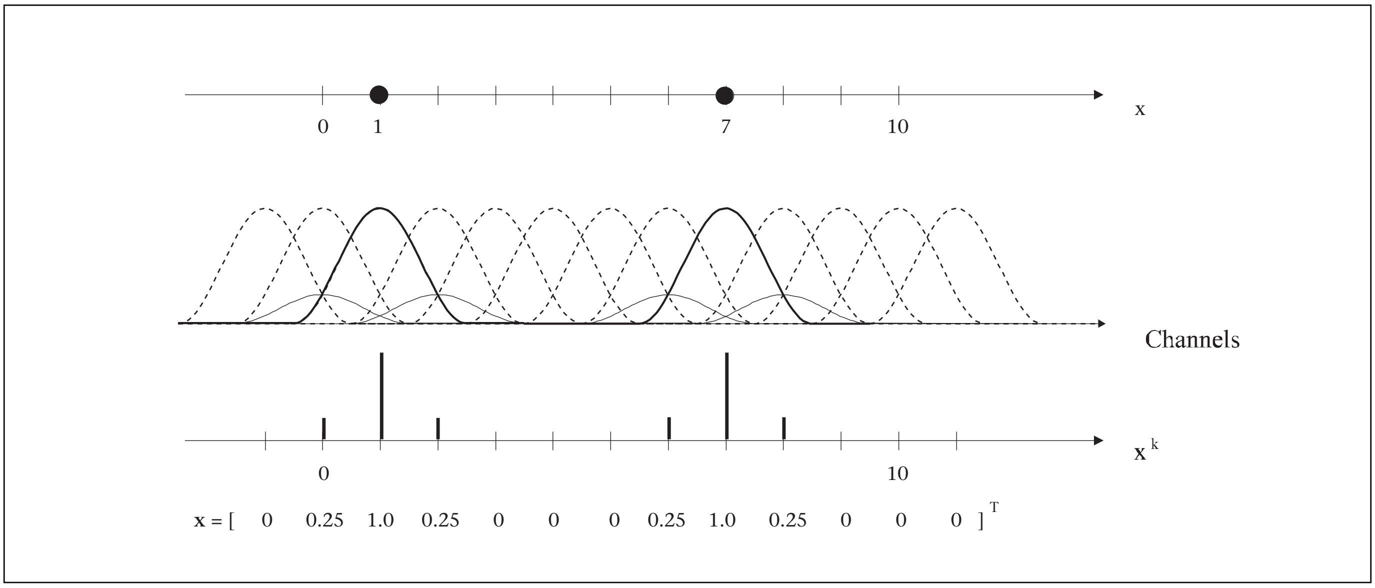


Figure 9. Channel Representation as a Vector **x** *of Two Scalars x = 1 and x = 7.*

point features, only limited parts of the spatial domain will have nonzero contributions. No change provides the basis for a sparse representation, which gives improved efficiency in storage and better performance in processing.

The locality of the channel representation allows for a fast convergence in the learning process to solve for the linkage matrixes (Granlund 2000).

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 (Granlund and Knutsson 1995; Johansson and Granlund 2000), but any local interest points or sparse features representable in a single vector can be used (figure 10). 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 $\mathbf{f}_k$ (figure 7).

$\mathbf{f}_k$: Point feature vectors, $k = 1,2,3$, each one coded with h_f channels

α: Angle between triplet legs 1 and 2, coded with h_α channels

γ: Relative length of triplet legs, $\gamma = \frac{L_1 - L_2}{L_1 + L_2}$, coded with h_γ channels.

A triplet is represented by the Kronecker product, **a**, between the preceding components:

$$\mathbf{a} = \mathbf{f}_1 \otimes \mathbf{f}_2 \otimes \mathbf{f}_3 \otimes \alpha \otimes \gamma \qquad (5)$$

We denote **a** a *triplet vector*. Feature vectors f_k, as well as the other parameters, can typically each be represented by 8 to 12 channels. For a case where $h_f = h_\alpha = h_\gamma = 8$, for **a**, we obtain $h_a = (h_f)^3\, h_\alpha\, h_\gamma = 85 = 32768$ channels or components. This number might seem large, but the sparse character of the representation makes computations fast because the number of nonzero components is maximally $3^5 = 243$ for a channel distance of $\pi/3$, or $2^5 = 32$ for a channel distance of $\pi/2$.

Mapping to Dynamic Binding Variables in the Form of System States

As stated in the introductory discussion, a mapping is desired of the purely geometric and feature-related entities into variables that correspond to states of the object. These variables fulfill two requirements: First, they change as a consequence of manipulation of the object, which is essential to separate the object from its background, and second, 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 that might 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 (Fodor and Pylyshyn 1988). The change of state, which is common to sev-

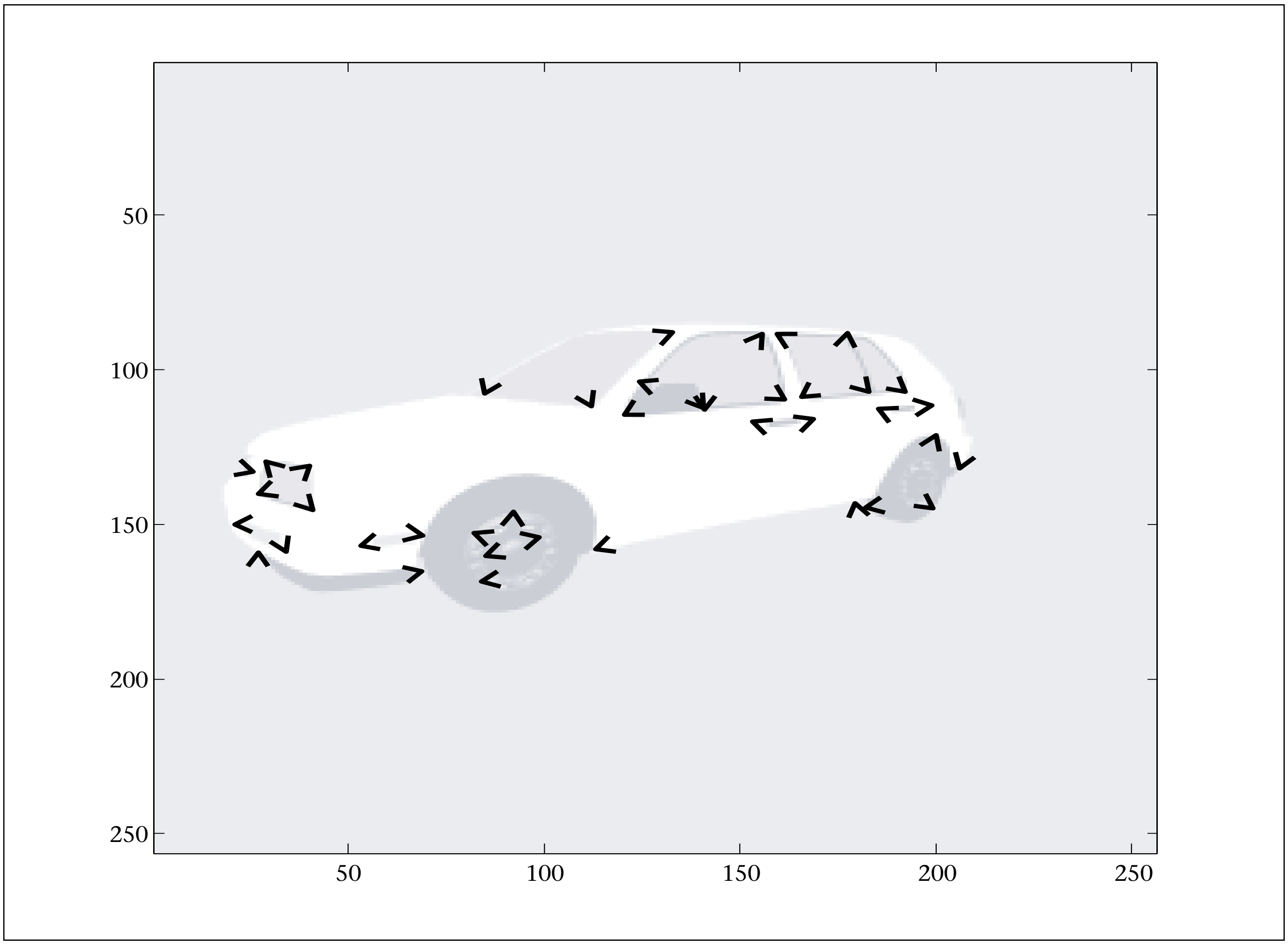


Figure 10. Example of Sparse Curvature Features of a Car.

eral models, can be viewed as a dynamic binding variable.

Object states that might be suitable as dynamic binding variables can be found in table 1. Suitable choices are all or a subset of a state vector **u**:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_\phi \\ \mathbf{u}_\theta \\ \mathbf{u}_\varphi \\ \mathbf{u}_{L_1} \end{bmatrix} = ch \begin{bmatrix} \phi \\ \theta \\ \varphi \\ L_1 \end{bmatrix} \qquad (6)$$

where *ch* stands for channel coding of the scalar variables. Vector components $\mathbf{u}_\varphi$ and $\mathbf{u}_{L1}$ can be viewed as local varieties of ψ and *s*.

We can view the triplet vector **a** as a function *f* of the total system state vector **u**:

$$\mathbf{a} = f(\mathbf{u}) \qquad (7)$$

We assume that the system state, **u**, can be expressed as a linear mapping:

$$\mathbf{u} = \mathbf{Ca} \qquad (8)$$

where **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 matrixes **A** and **U**. Matrix **C** is the solution of

$$\mathbf{U} = \mathbf{CA} \qquad (9)$$

The linkage matrix **C** contains hundreds or thousands of different models, each one valid within some subspace of the vector space of **A** mapping onto some subspace of the vector space of **U**. The localized channel representation allows the implementation of nonlinear models using a linear mapping. The effect of the fragmentation into the channel representation is to generate separable subspaces for the original, scalar variables. In addition, this representation leads to a fast optimization.

The details of this solution procedure are omitted in this presentation, but additional details are given in Granlund (2000).

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 $\mathbf{f}^{\kappa}_{\mathbf{k}}$ are constructed by combining the following channel-coded features:

$$\hat{\mathbf{u}}^{\kappa-1}_{\kappa\phi}, \hat{\mathbf{u}}^{\kappa-1}_{\kappa\theta}, \hat{\mathbf{u}}^{\kappa-1}_{\kappa\varphi}, \hat{\mathbf{u}}^{\kappa-1}_{k\mathrm{L}_1}, \upsilon^{\kappa}_{k}, \Gamma^{\kappa}_{k},$$

where κ is the level of the triplet, υ^{κ} is the relative orientation between the level κ triplet and level κ–1 triplet, and

$$\Gamma^{\kappa} = \frac{L_1^{\kappa} - L_1^{\kappa-1}}{L_1^{\kappa} + L_1^{\kappa-1}}$$

(figure 8). Currently, the combination is done in the following way

$$\mathbf{f}^{\kappa}_{\mathbf{k}} = \begin{bmatrix} \hat{\mathbf{u}}^{\kappa-1}_{k\phi} \\ \hat{\mathbf{u}}^{\kappa-1}_{k\theta} \\ \hat{\mathbf{u}}^{\kappa-1}_{k\varphi} \\ \hat{\mathbf{u}}^{\kappa-1}_{k\mathrm{L}_1} \end{bmatrix} \otimes \Gamma^{\kappa}_{k} \otimes \upsilon^{\kappa}_{k} \tag{10}$$

but there are other possibilities given the computational complexity accepted.

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

Mapping for Higher-Level Triplets

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

$$a^{\kappa}_{k} = \mathbf{f}^{\kappa}_{k} \otimes \alpha^{\kappa} \otimes \gamma^{\kappa} \quad k = 1, 2, 3 \tag{11}$$

A training process generates one linkage matrix $\mathbf{C}_{\mathbf{k}}$ for each one of the nodes:

$$\mathrm{U} = \mathbf{C}_k\, \mathbf{A}_k \quad \mathrm{k} = 1, 2, 3 \tag{12}$$

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

$$\hat{\mathbf{u}}_k = \mathbf{C}_k \mathbf{a}_k \quad k = 1, 2, 3 \tag{13}$$

Thus, a feature vector $\mathbf{f}_k$ is interpreted under the contextual restriction or modification $\alpha \otimes \gamma$. Given that we deal with measurements on the same object, there are parameters that should be estimated to the same value, for example, the pose angles ϕ and θ.

In an ideal case, such state estimates should all be equal because they represent different measurements of the same property on the same object:

$$\hat{\mathbf{u}}_{1\phi} = \hat{\mathbf{u}}_{2\phi} = \hat{\mathbf{u}}_{3\phi} \tag{14}$$

$$\hat{\mathbf{u}}_{1\theta} = \hat{\mathbf{u}}_{2\theta} = \hat{\mathbf{u}}_{3\theta} \tag{15}$$

In reality, there is noise, which might be from minor variations in measurements to an erroneous hypothesis about a model. To reduce this noise, a consistency check between the three statements is made, and confidence measures can be derived from the similarity of statements:

$$\left. \hat{\mathbf{u}}_{\phi k} - \hat{\mathbf{u}}_{\phi(k+1)_{\mathrm{mod}\,3}} \right| < c_{\phi k(k+1)_{\mathrm{mod}\,3}} \quad k = 1, 2, 3 \tag{16}$$

This procedure 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.

Removal of Multiple Models

A model constitutes a subset of the linkage matrix **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

$$\frac{\langle \mathbf{a}(n_2) \mid \mathbf{a}(n_1) \rangle}{|\mathbf{a}(n_2)||\mathbf{a}(n_1)|} > a_c \tag{17}$$

and

$$\frac{\langle \mathbf{u}(n_2) \mid \mathbf{u}(n_1) \rangle}{|\mathbf{u}(n_2)||\mathbf{u}(n_1)|} > u_c \tag{18}$$

Subsets of samples that fulfill this continuity 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 **C** to contain thousands of different models that are each continuous but form discrete patches in both spaces.

Every model in the set must be unambiguous in that it maps only onto a single state **u** for a given input **a.** However, different input, **a**, might map onto the same state **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 17 is satisfied but not inequality 18.

The preceding procedure allows us to use lower-level models in the assembly of higher-level models for entirely different objects than where the low-level model was derived. Thus, the output state from the low-level model will no longer be equivalent to the actual state of the new object. Because a new remapping is used when the low-level triplet is used in a new higher-level triplet, this confusion is no problem. In this case, the intermediary triplet-output variables will have nothing to do with the actual output parameters in the current training but act as a state and object identity in a "local language" that is retransformed in the training to the next-level triplet. This retransformation is facilitated because of the use of the channel representation.

Responses

Fundamental to the strategy is that several of the hypothetical models might be erroneous, but there should be a sufficient number of selected hypothetical models that are correct. The way they know that they are correct is that they are saying the same thing; that is, output must cluster.

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 that can cluster to robust estimates of rotation and scale.

Clustering of the responses pose-x ($\hat{\phi}$(*n*)) and pose-y (θ (*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 on 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 that give the responses in the clusters. This clustering gives the ability to find several objects with the same pose angles in the same image.

If a certain multilevel triplet gives a statement that 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 its scale. Thus, the triplet is trained to map onto these variables.

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 $\varphi(n)$.

$$\hat{\psi}(n) = \hat{\varphi}(n) - \varphi(n) \qquad (19)$$

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

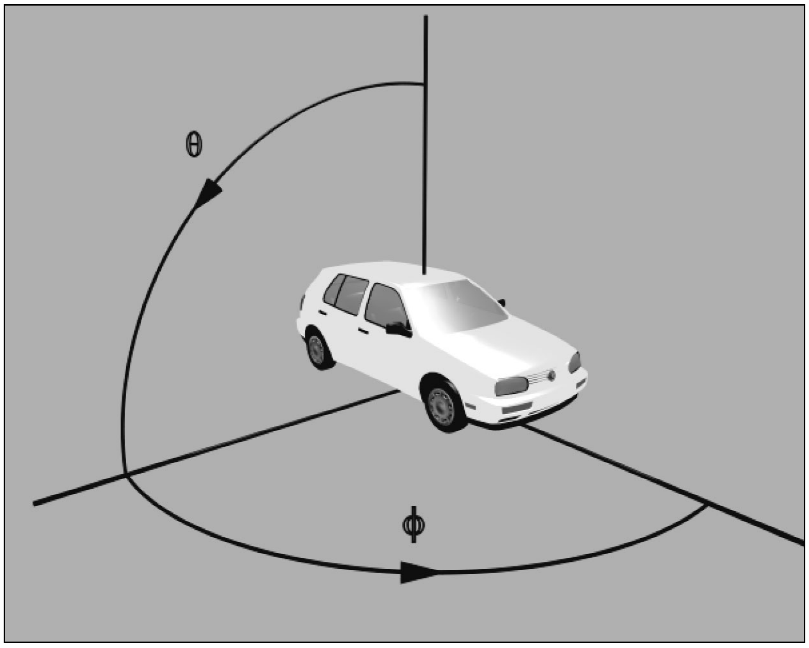


Figure 11. Object Training Setup: θ and φ Are the Two Pose Angles.

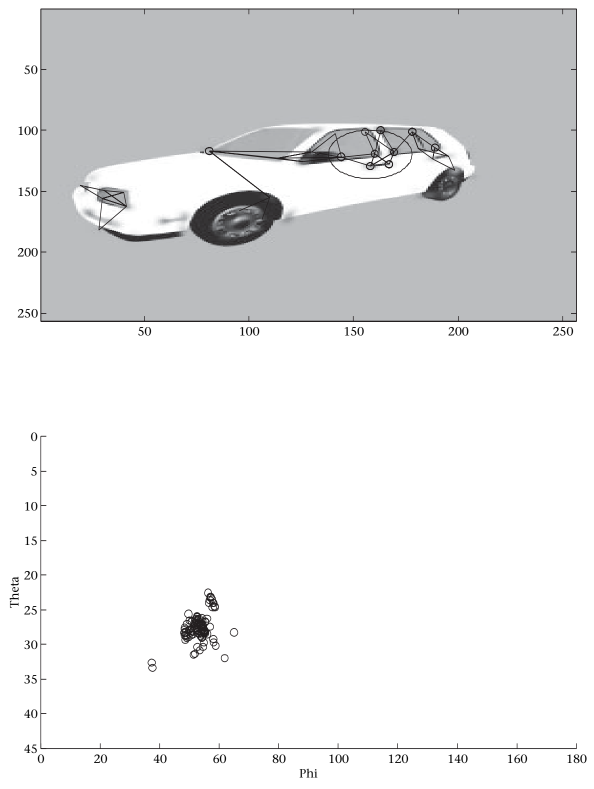


Figure 12. Example of Triplets Found in a Certain Pose.

The lines are the first-level triplets, the small circles indicate 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.

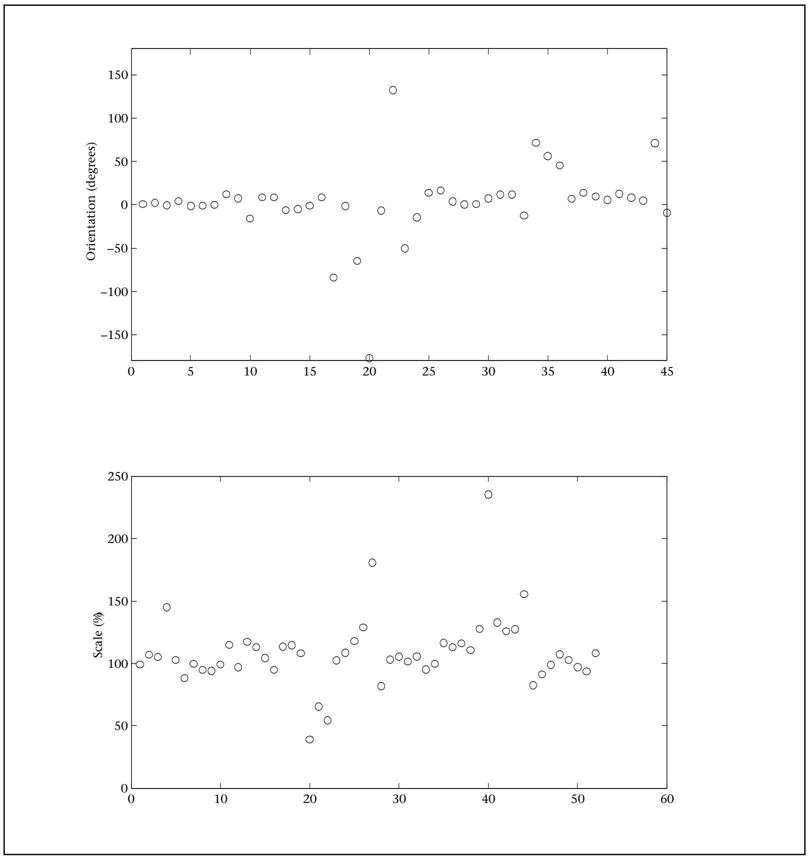


Figure 13. Estimates of the Orientation and Scale for Accepted Triplets in Figure 12.

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	1.3°
pose-y	1.2°
Orientation	4.4°
Scale	4.2%

Table 3. Average Error for Evaluation Images Scaled 10 Percent and Rotated 60 Degrees.

triplet during the training and should consequently be the same for all responses obtained from an unknown 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 $\hat{L}_1$ of the original triplet at training by the actual length of the current triplet L_1

$$\hat{s}(n) = \frac{\hat{L}_1(n)}{L_1(n)} \tag{20}$$

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

$$\hat{s} = \frac{\sum_n \hat{L}_1(n) L_1(n)}{\sum_n L_1(n)^2} \tag{21}$$

Experiments

The recognition structure has been trained on computer-generated images of a car. Because the triplet representation is invariant to translation, rotation, and scale, it is only necessary to train for different pose angles (θ,ϕ) (figure 11). The pose angles are varied with 5-degree increments between 50 and 90 degrees for θ and between 0 and 180 degrees for ϕ, which gives 333 images. With about 20 first-level triplets in each image, we get 5781 first-level triplets and 27,750 second-level triplet structures.

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 and the obtained pose estimates are shown in figure 12. The orientation estimates and the scale estimates are shown in figure 13. One can see that the estimates of the pose angles and the orientation are quite stable, but 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 degrees, 0.4 degrees, 4.0 degrees, and 2.5 percent 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.

Because the curvature features are not totally scale invariant, we will not get estimates that are as good if we scale the object. Figure 14 shows the car rotated 60 degrees and scaled 20 percent. The image size is kept constant when the object is scaled, so occlusion will occur and affect the estimates as well. Figure 15 shows the orientation and scale estimates. The estimation errors for this image are 1.4 degrees, –2.3 degrees, 8.5 degrees, and –2.4 percent for pose-x, pose-y, orientation, and scale, respectively. The average estimation errors for the object rotated

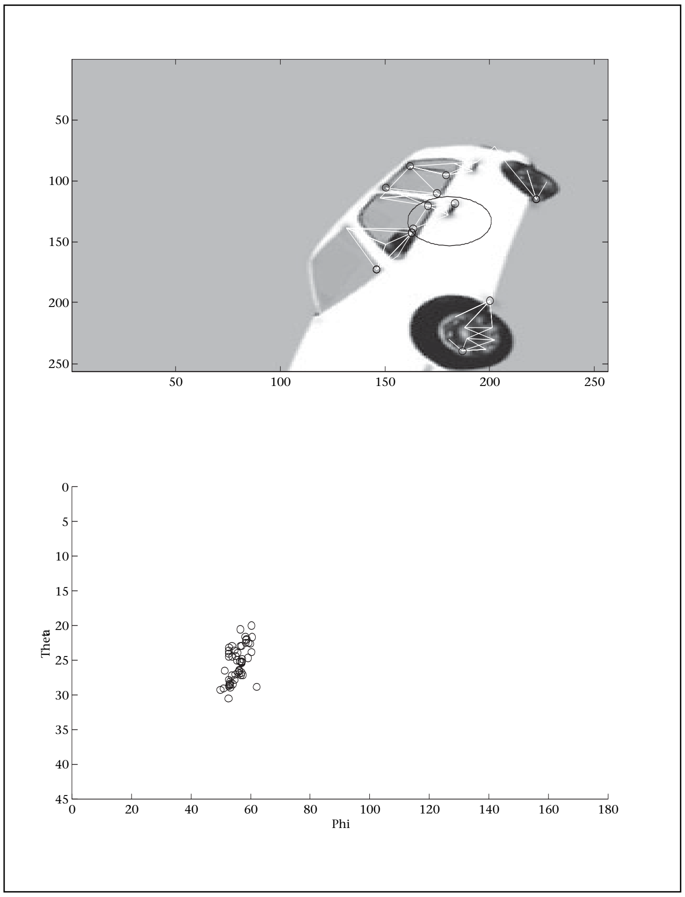


Figure 14. Example of Recognition of Pose for a Rotated, Partially Occluded Object at a Scale Different from Training.

The lines are the first-level triplets, the small circles indicate the accepted triplets, and the large circle is the estimated position of the object.

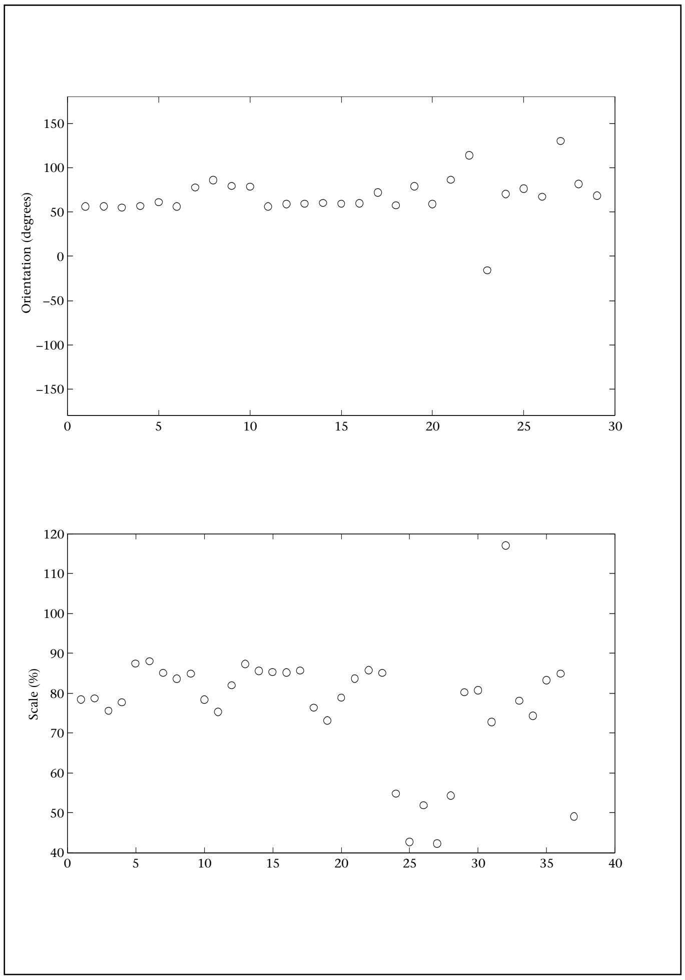


Figure 15. Estimates of the Orientation and Scale for the Accepted Triplets of the Object in Figure 14.

60 degrees and scaled 10 percent are shown in table 3.

In figure 16, two car objects were 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 little influence on the results. The obtained estimation errors for the right car are –0.9 degrees, 1.5 degrees, 8.6 degrees, and 1.1 percent and for the left car –0.6 degrees, 0.8 degrees, 0.1 degrees, and 5.5 percent for the pose-x, pose-y, orientation, and scale, respectively.

Recognition of Object Class

In this article, there has not been much discussion about mapping into object classes. Given the fact that we obtain sets of triplets that consistently point out a certain state, these same triplets can be used to map into an object class. An object belonging to a different class is unlikely to activate the same set of triplets and, furthermore, obtains consistent statements of state from all triplets.

The crucial mechanism is the use of the dynamic binding parameters to establish if correct hypothetical models for different levels have been selected. If they have been selected, we can use the same triplet models to map onto a suitable class membership variable.

Although 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 this article. What gives us a benefit with this approach is that lower-level models acquired earlier can be used in the recognition of other objects, which means that learning of objects can be made in an incremental, cumulative fashion. Thus, the complexity of data is expected to expand at a rate less than linear with respect to the number of

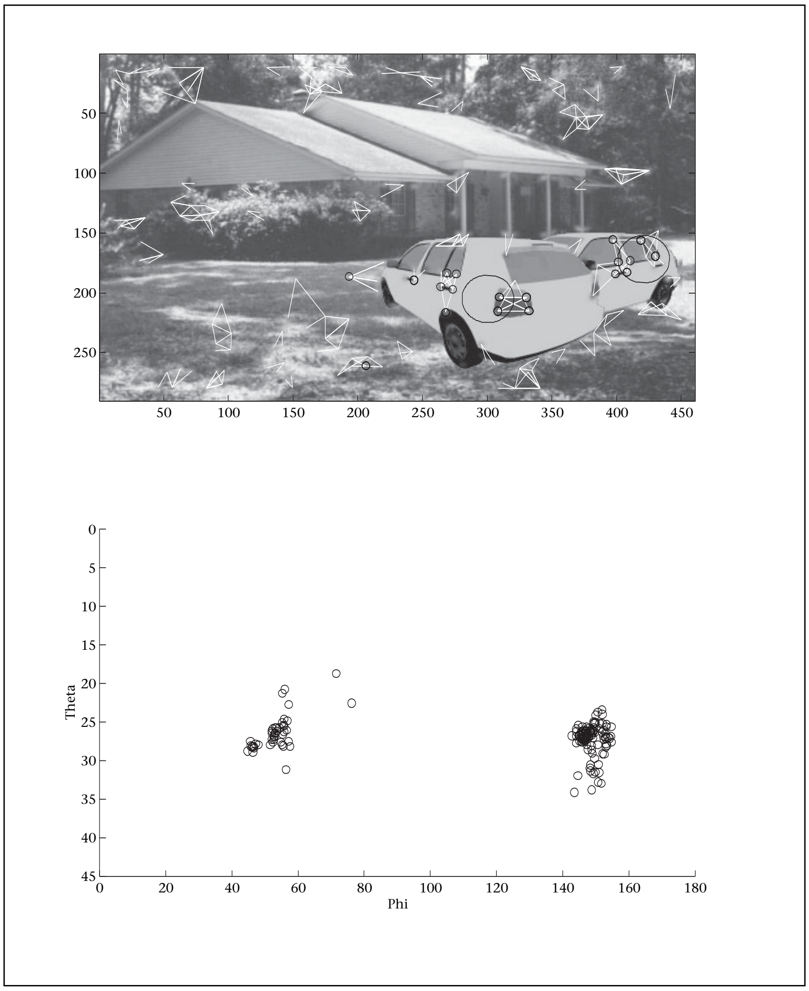


Figure 16. Test of Recognition of Pose for Two Partially Occluded Objects against a Structured Background.

The lines are the first-level triplets, the small circles indicate the accepted triplets, and the large circle is the estimated position of the object.

objects. How much less it expands is an issue for further research.

Acknowledgments

The work presented in this article was supported by the Wallenberg Laboratory on Information Technology and Autonomous Systems and by EU Project VISATEC (vision-based integrated system adaptive to task and environment with cognitive abilities), which is gratefully acknowledged. The work builds on several methodologies developed earlier at the Computer Vision Laboratory.

Notes

1. Using the modulo operation $\mathrm{mod}(x, K) = x - \lfloor x/K \rfloor K$.

References

Besl, P. J., and Jain, R. C. 1985. Three-Dimensional Object Recognition. *ACM Computing Surveys* 17(1): 75–145.

Beymer, D., and Poggio, T. 1996. Image Representations for Visual Learning. *Science* 272(28): 1905–1909.

Felsberg, M.; Forssen, P.; and Scharr, H. 2004. B-Spline Channel Smoothing for Robust Estimation. Technical Report, LiTH-ISY-R-2579, Department of Electrical Engineering, Linköping University.

Fodor, J. A., and Pylyshyn, Z. W. 1988. Connectionism and Cognitive Architecture: A Critical Analysis. In *Connections and Symbols,* eds. S. Pinker and J. Mehler, 3–71. Cambridge, Mass.: MIT Press.

Granlund, G. H. 2000. An Associative Perception-Action Structure Using a Localized Space Variant Information Representation. In *Algebraic Frames for the Perception-Action Cycle (AFPAC): Second International Workshop, AFPAC-2000,* eds. G. Sommer, Y. Zeevi, J. Van Leeuwen, J. Hartmanis, and G. Goos, 29–53. Lecture Notes in Computer Science 1888. New York: Springer-Verlag.

Granlund, G. H. 1999a. Does Vision Inevitably Have to Be Active? Paper presented at the Eleventh Scandinavian Conference on Image Analysis, 7–11 June, Kangerlussuaq, Greenland.

Granlund, G. H. 1999b. The Complexity of Vision. *Signal Processing* 74(28): 101–126.

Granlund, G. H., and Knutsson, H. 1995. *Signal Processing for Computer Vision.* Dordrecht, The Netherlands: Kluwer Academic.

Grimson, W. E. L. 1990. *Object Recognition by Computer: The Role of Geometric Constraints*. Cambridge, Mass.: MIT Press.

Isaksson, M. 2002. Face Detection and Pose Estimation using Triplet Invariants. Master's thesis, Department of Electrical Engineering, LiTH-ISY-EX-3223, Linköping University.

Jacobsson, L., and Wechsler, H. 1982. A Paradigm for Invariant Object Recognition of Brightness, Optical Flow, and Binocular Disparity Images. *Pattern Recognition Letters* 1(1): 61–68.

Johansson, B., and Granlund, G. H. 2000. Fast Selective Detection of Rotational Symmetries Using Normalized Inhibition. In *Computer Vision—ECCV 2000, Sixth European Conference on Computer Vision,* ed. D. Vernon, 871–887. Lecture Notes in Computer Science 1842. New York: Springer-Verlag.

Kanatani, K. 1987. Camera Rotation Invariance of Image Characteristics. *Computer Vision, Graphics, and Image Processing* 39(3): 328–354.

Koenderink, J. J., and van Doorn, A. J. 1975. Invariant Properties of the Motion Parallax Field due to the Movement of Rigid Bodies Relative to an Observer. *Optica Acta* 22(9): 773–791.

Lowe, D. G. 2001. Local Feature View Clustering for 3D Object Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 682–688. Washington, D.C.: IEEE Computer Society.

Lowe, D. G. 1999. Object Recognition from Local Scale-Invariant Features. Paper presented at the International Conference on Computer Vision (ICCV-99), 20–25 September, Kerkyra, Corfu, Greece.

Mamic, G., and Bennamoun, M. 2002. Representation and Recognition of 3D Free-Form Objects. *Digital Signal Processing* 12(1): 47–76.

Matas, J.; Burianek, J.; and Kittler, J. 2000. Object Recognition Using the Invariant Pixel-Set Signature. Paper presented at the British Machine Vision Conference (BMVC2000), 11–14 September, Bristol, U.K.

Mikolajczyk, K., and Schmid, C. 2001. Indexing Based on Scale Invariant Interest Points. Paper presented at the International Conference on Computer Vision (ICCV2001), 9–12 July, Vancouver, Canada.

Mundy, J. L., and Zisserman, A. 1992. *Geometric Invariance in Computer Vision.* Cambridge, Mass.: MIT Press.

Murase, H., and Nayar, S. K. 1995. Visual Learning and Recognition of 3D Objects from Appearance. *International Journal of Computer Vision* 14(1): 5–24.

Poggio, T., and Edelman, S. 1990. A Network That Learns to Recognize Three-Dimensional Objects. *Nature* 343:263–266.

Pulli, K., and Shapiro, L. G. 1996. Triplet-Based Object Recognition Using Synthetic and Real Probability Models. In Proceedings of the Thirteenth International Conference on Pattern Recognition, 75–79. Washington, D.C.: IEEE Computer Society.

Riesenhuber, M., and Poggio, T. 2000a. Computational Models of Object Recognition in Cortex: A Review. Technical Report, 1695, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Riesenhuber, M., and Poggio, T. 2000b. Computational Models of Object Recognition in Cortex: A Review. Technical Report, 190, Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology.

Schiele, B., and Pentland, A. 1999. Probabilistic Object Recognition and Localization. Paper presented at the International Conference on Computer Vision, 20–25 September, Kerkyra, Corfu, Greece.

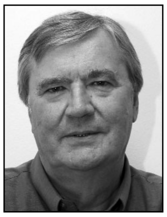
Schmid, C., and Mohr, R. 1997. Local Greyvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(5): 530–535.

Schmid, C.; Mohr, R.; and Bauckhage, C. 2000. Evaluation of Interest Point Detectors. *International Journal of Computer Vision* 37(2): 151–172.

Ullman, S., and Basri, R. 1991. Recognition by Linear Combinations of Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13(10): 992–1006.

Weiss, I., and Ray, M. 2001. Model-Based Recognition of 3D Objects from Single Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23(2): 116–128.

Yuan, C., and Niemann, H. 2001. Neural Networks for the Recognition and Pose Estimation of 3D Objects from a Single 2D Perspective View. *Image and Vision Computing* 19(9–10): 585–592.



Gösta H. Granlund heads the Computer Vision Laboratory of Linköping University, specializing in areas such as reconstruction, multidimensional information representation, adaptive feedback and relaxation methods, learning systems, and hierarchical parallel processing structures. He has coauthored or contributed to four books on computer vision methods. He has written or coauthored more than 120 articles or conference papers within the field of computer vision and pattern recognition, and he has given international courses within this field. He has 8 patents. He is member of the editorial board for several international journals. His e-mail address is gegran@isy.liu.se.

Anders Moe received his Licentiate of Technology from Linköping University in 2000. Currently, he is a research engineer at the Computer Vision Laboratory of Linköping University. His main research interests are object recognition and navigation from image data. His e-mail address is moe@ isy.liu.se.