

A Probabilistic Framework for 3D Visual Object Representation

Renaud Detry Nicolas Pugeault Justus Piater

Abstract—We present an object representation framework that encodes probabilistic spatial relations between 3D features and organizes these features in a hierarchy. Features at the bottom of the hierarchy are bound to local 3D descriptors. Higher-level features recursively encode probabilistic spatial configurations of more elementary features. The hierarchy is implemented in a Markov network. Detection is carried out by a belief propagation algorithm, which infers the pose of high-level features from local evidence and reinforces local evidence from globally consistent knowledge, effectively producing a *likelihood* for the pose of the object in the detection scene. We also present a simple learning algorithm that autonomously builds hierarchies from local object descriptors. We explain how to use our framework to estimate the pose of a known object in an unknown scene. Experiments demonstrate the robustness of hierarchies to input noise, viewpoint changes and occlusions.

Index Terms—Computer vision, 3D object representation, pose estimation, Nonparametric Belief Propagation.

(c) 2009 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works.

1 INTRODUCTION

THE merits of part-based and hierarchical approaches to object modeling have often been put forward in the vision community [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]. Part-based models typically separate structure from appearance, which allows them to deal with variability separately in each modality. A hierarchy of parts takes this idea further, by introducing scale-dependent variability: small part configurations can be tightly constrained, while wider associations can allow for more variability. Furthermore, part-based models do not only allow for the detection and localization of an object, but also parsing of its constituent parts. They lend themselves to part sharing and reuse, which should help in overcoming the problem of storage size and detection cost in large object databases. Finally, these models not only allow for bottom-up inference of object parameters based on features detected in images, but also for top-down inference of image-space appearance based on object parameters.

A large body of the object modeling literature focuses on modeling the 2D projections of a 3D object. A major issue with this approach is that all variations introduced by projective geometry (geometrical transformations, self-occlusions) have to be robustly captured and

handled by the model. In the past few years, modeling objects directly in 3D has become increasingly popular [11], [12], [13], [14], [15], [16]. The main advantage of these methods lies in their natural ability to handle projective transformations and self-occlusions.

The main contribution of this paper is a framework that encodes the 3D geometry and visual appearance of an object into a part-based model, and mechanisms for autonomous learning and probabilistic inference of the model. Our representation combines local appearance and 3D spatial relationships through a hierarchy of increasingly expressive *features*. Features at the bottom of the hierarchy are bound to local 3D visual perceptions called *observations*. Features at other levels represent combinations of more elementary features, encoding probabilistic relative spatial relationships between their children. The top level of the hierarchy contains a single feature which represents the whole object.

The hierarchy is implemented in a Markov random field, where features correspond to hidden variables, and spatial relationships define pairwise potentials. To detect instances of a model in a scene, observational evidence is propagated throughout the hierarchy by probabilistic inference mechanisms, leading to one or more consistent scene interpretations. Thus, the model is able to suggest a number of likely *poses* for the object, a pose being composed of a 3D world location and a 3D world orientation. The inference process follows a nonparametric belief propagation scheme [17] which uses importance sampling for message products. The model is bound to no particular learning scheme. In this paper, we present an autonomous learning method that builds hierarchies in a bottom-up fashion.

Learning and detection algorithms reason directly on sets of local 3D visual perceptions which we will refer to as (*input*) *observations*. These observations should represent visual input in terms of 3D descriptors, i.e.

- R. Detry (Renaud.Detry@ULg.ac.be) and J. Piater are with the INTEL SIG Laboratory, Univ. of Liège, Belgium.
- N. Pugeault is with the Cognitive Vision Lab, The Maersk Mc-Kinney Møller Inst., Univ. of Southern Denmark, Denmark.

Manuscript received 16 Aug. 2008; revised 18 Dec. 2008; accepted 5 Mar. 2009; published online 17 Mar. 2009.

Recommended for acceptance by Q. Ji, A. Torralba, T. Huang, E. Sudderth, and J. Luo.

This is an author postprint.

For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMSI-2008-08-0529.

Digital Object Identifier no. 10.1109/TPAMI.2009.64.



Fig. 1. ECV observations from Krüger et al. [18]

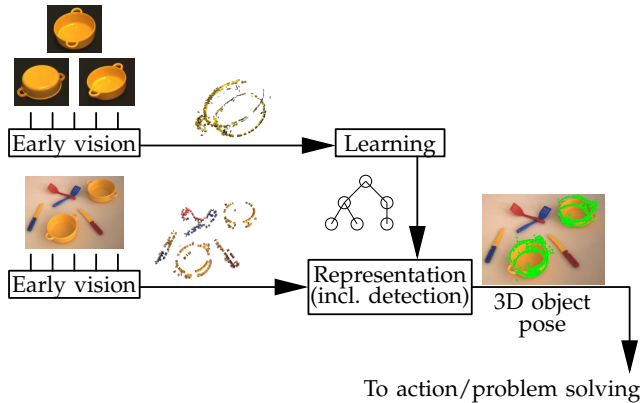


Fig. 2. Pose estimation system

points characterized by a 3D location, a local appearance, and possibly an orientation. Prior research on model detection directly in 3D data can be found e.g. in the work of Rogers et al. [13].

In this paper, we evaluate our object representation on input observations produced with the 3D Early-Cognitive-Vision (ECV) system of Krüger et al. [18], [19]. From stereo imagery, this system extracts pixel patches along image contours, and computes 3D patch positions and orientations by stereopsis across image pairs. The resulting reconstruction consists of a set of short edge segments in 3D space (Fig. 1), which we will refer to as *ECV observations*. Each ECV observation corresponds to a patch of about 25 square millimeters of *object* surface. Since image patches are extracted on edges, each ECV observation can be characterized with an appearance descriptor composed of the two colors found on the sides of the edge. An ECV reconstruction of a scene typically contains 1000–5000 observations.

The task on which we evaluate our model is object pose estimation. Fig. 2 illustrates the pose estimation process. Observations are provided by the ECV system. The learning algorithm builds a hierarchy from a set of observations from a segmented object; the hierarchy is then used to recover the pose of the object in a cluttered scene. Preliminary results appeared in conference and workshop proceedings [20], [21], [22]. We note that even though we concentrate our evaluation on 3D data from Krüger et al., our system can in principle be applied to other 3D sources, such as dense stereo, or range data.

We emphasize that we intend to develop *generative*

representations that allow for detection and localization of known objects within highly cluttered scenes, as opposed to object classification which is best achieved using *discriminative* models.

2 RELATED WORK

Recent work has shown many successful implementations of the hierarchical approach to object modeling in 2D, with both bottom-up [7], [9], [20] and top-down [8] unsupervised learning of object parts and structure. These generally aim at classifying objects into generic categories, and put a strong accent on learning in order to capture intra-class variability. Intra-class variability is very important for these methods also because it allows them to capture the affine transformations that the object undergoes when projected to 2D images. Our learning procedure is similar in spirit, although it requires much less sophistication. Since we work directly in 3D, projection-related issues are handled by the method that produces the observations, with e.g. stereopsis when using ECV observations. Our system is thus intrinsically robust to problems like the decreasing apparent 2D size of an object when moved away from the camera, and projective deformations do not need to be learned. Compared to these 2D methods, the most distinguishing aspects of our approach are its explicit 3D support and a unified probabilistic formalization through a Markov network.

Part-based models have proved very efficient for representing articulated objects in 2D [10], [23], [24], [25], [26] or 3D [13], [27], [28], and matching them in 2D images [10], [23], [24], [25], [26], [27], [28] or 3D range data [13]. Articulated objects are often formalized through a graphical model, and probabilistic inference algorithms such as belief propagation [29] or its variants are widely used for detection [23], [24], [27], [28]. The model topology, i.e. the parts, are typically defined by hand. Model parameters (compatibility and observation potentials) are also often defined by hand [10], [13], [24], although it has been shown that compatibility potentials can be learned from motion [26], [27], and observation potentials (image likelihoods) may emerge from annotated images [23], [26].

Despite the similar formalism, the issue addressed by these methods is rather different from ours. The work cited in the previous paragraph seeks *unique* matches of relatively *high-level* entities (body parts, ...), and represents *loose, one-to-one* relations between them. In our work, a part is an abstract concept that may have any number of instances; a parent-child potential thus encodes a one-to-many relationship, as a set of relationships between the parent part and many child part instances. Furthermore, our parts reach a much finer level of granularity; a part can for instance correspond to a “red-ish world-surface patch” as small as 5×5 mm.

Coughlan et al. [30] have demonstrated a 2D part-based object model where parts are relatively low-level

(e.g. short 2D edge segment). However, in the same way as the articulated models mentioned above, potentials encode one-to-one relationships between neighboring parts.

When targeting 3D objects, a crucial model property is viewpoint invariance, i.e. the ability to encode object information independently of the viewpoint. Although viewpoint invariance has been achieved in 2D [31], an increasingly popular means of achieving viewpoint invariance is to represent object geometry directly in 3D.

Very interesting work [11], [12], [32] takes the approach of representing an object with a large set of local affine-invariant descriptors and the relative 3D spatial relationships between the corresponding surface patches. Other methods organize local appearance descriptors on a 3D shape model, obtained by CAD [16] or 3D homography [14]. To detect an object in an image, these methods start with an appearance-based matching of image descriptors with model descriptors. In a second phase, global optimization is used to evaluate the geometrical consistency of the appearance-based matches using the geometrical information contained in the model, and compute a 2D bounding box [14] or a 3D pose [16]. Rothganger et al. [11] do not speak of 3D pose in their results, although it seems obvious that a 3D pose is implicitly computed during detection.

Instead of using a precise 3D shape model, objects have also successfully been represented by a set of near-planar *parts* connected through their mutual homographic transformations [15]. In this context, a part is a large region of the object represented with many local invariant descriptors.

Compared to our approach, the preceding 3D methods represent an object as a whole, and do not intrinsically allow for parsing an object into parts. Another distinguishing aspect of our work is its probabilistic formalization through a graphical model. Finally, the 3D methods cited above work with image data, and encompass the entire reasoning process from image pixels to object pose. In our case, a large part of the work is done by the upstream system that produces 3D observations – e.g. the ECV system, or range scanning. This allows us to concentrate on the encoding of 3D geometry and appearance, and yields a framework that can cope with different sources of 3D observations.

The performance of our system is strongly influenced by the characteristics of the 3D observations we use. While the 3D methods cited above rely on affine-invariant descriptors, for which textured objects are ideal, the ECV system used in this paper extracts surface edges, and thus strongly prefers objects with clear edges and little texture.

For prior work on top-down parsing of scenes and objects, we refer the reader to the work of Lee and Mumford [5], and of Tu et al. [6]. We note that our representation and methods are compatible [33] with the ideas developed by Lee and Mumford.

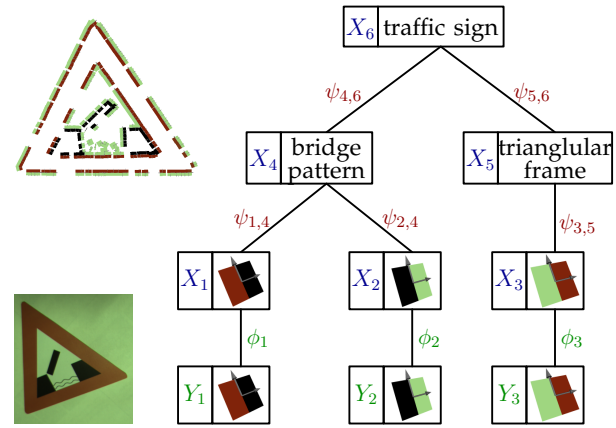


Fig. 3. On the right: an example of a hierarchy for the traffic sign shown in the bottom-left corner. X_1 through X_3 are primitive features; each of these is linked to an observed variable Y_i . X_4 through X_6 are meta-features. Top-left: ECV observations of the traffic sign.

3 HIERARCHICAL MODEL

Our object model consists of a set of generic *features* organized in a hierarchy. Features that form the bottom level of the hierarchy, referred to as *primitive features*, are bound to visual observations. The rest of the features are *meta-features* which embody relative spatial configurations of more elementary features, either meta or primitive.

A feature can intuitively be associated to a “part” of an object, i.e. a generic component instantiated once or several times during a “mental reconstruction” of the object. At the bottom of the hierarchy, primitive features correspond to local parts that each may have many *instances* in the object. Climbing up the hierarchy, meta-features correspond to increasingly complex parts defined in terms of constellations of lower parts. Eventually, parts become complex enough to satisfactorily represent the whole object. In this paper, a primitive feature represents a class of ECV observations of similar appearance, e.g. an ECV observation with colors close to red and white. Given the large number of observations produced by the ECV system, a primitive feature will usually have hundreds of instances in a scene.

Fig. 3 shows an example of a hierarchy for a traffic sign. Ignoring the nodes labeled Y_i for now, the figure shows the traffic sign as the combination of two features: a triangular frame (feature 5) and a bridge pattern (feature 4). The fact that the bridge pattern has to be in the center of the triangle to form the traffic sign is encoded in the links between features 4-6-5. The triangular frame is further encoded using a single (generic) feature: a short red-white edge segment (feature 3). The link between feature 3 and feature 5 encodes the fact that many short red-white edge segments (several hundreds of instances of feature 3, i.e. several hundreds of red-ish ECV observations) are necessary to form the triangular frame, and the fact that these edges have to be arranged

along a triangle-shaped structure. During instantiation, the activation of a (single) feature (e.g. feature 3) will represent *all* its instances (the pose of hundreds of instances of feature 3).

We emphasize that for us, a “feature” is an abstract concept that may have *any number of instances*. The lower-level the feature, the *larger* generally the number of instances. Conversely, the higher-level the feature, the *richer* its spatio-appearance description (because it represents a combination of lower-level features), and thus the lower generally the number of its instances. The next section explains how instances are represented as one spatial probability density per feature, therefore avoiding specific model-to-scene correspondences.

3.1 Parametrization

Formally, the hierarchy is implemented in a Markov tree (Fig. 3). Features correspond to hidden nodes of the network. When a model is associated to a scene (during learning or instantiation), the pose of feature i in that scene will be represented by the probability density function of a random variable X_i , effectively linking feature i to its instances. Random variables are thus defined over the pose space, which exactly corresponds to the Special Euclidean group $SE(3) = \mathbb{R}^3 \times SO(3)$.

The structure of the hierarchy is reflected by the edge pattern of the network; each meta-feature is thus linked to its child features. As noted above, a meta-feature encodes the relationship between its children, which is done by recording the relative relationships between the meta-feature and each of its children. The relationship between a meta-feature i and one of its children j is parametrized by a *compatibility potential function* $\psi_{ij}(X_i, X_j)$ which reflects, for any given relative configuration of feature i and feature j , the likelihood of finding these two features in that relative configuration. The potential between i and j will be denoted equivalently by $\psi_{ij}(X_i, X_j)$ or $\psi_{ji}(X_j, X_i)$. We only consider rigid-body, *relative* spatial configurations. A compatibility potential is thus equivalent to the spatial distribution of the child feature in a reference frame that matches the pose of the parent feature; a potential can be represented by a probability density defined on $SE(3)$.

Finally, each primitive feature is linked to an observed variable Y_i . Observed variables are tagged with an appearance descriptor, called a *codebook vector*, that defines a class of observation appearance. In the case of ECV observations, a codebook vector will be composed of two colors. The set of all codebook vectors forms a *codebook* that binds the object model to feature observations. The statistical dependency between a hidden variable X_i and its observed variable Y_i is parametrized by a potential $\psi_{ii}(X_i, Y_i)$. We generally cannot observe meta-features; their observation potentials are thus uniform.

In the applications we are considering, observations are assumed constant throughout inference. Furthermore, the observations we are dealing with reflect directly the state of their corresponding hidden variable;

$\psi_{ii}(X_i, Y_i)$ thus encodes an identity transformation. Consequently, the notation for the statistical dependency between a hidden variable X_i and its observed variable Y_i will be simplified in an *observation potential* $\phi_i(X_i)$, which corresponds to the spatial distribution of the input observations that hold an appearance descriptor resembling the codebook of Y_i . An observation potential $\phi_i(X_i)$ will also be referred to as *evidence* for X_i .

3.2 Instantiation

Model instantiation is the process of detecting instances of an object model in a scene. It provides pose densities for all features of the model, indicating where the learned object and its sub-parts are likely to be present.

Because a hierarchy is implemented with a Markov network, there is a clear separation between the model, and the algorithms that make use of the model. Fundamentally, instantiation involves two operations:

- 1) Define priors (observation potentials) from input observations;
- 2) Propagate this information through the graph using an applicable inference algorithm.

During the definition of priors, each observation potential $\phi_i(X_i)$ is built by means described in Section 4.2 from a subset of the input observations. The subset that serves to build the potential $\phi_i(X_i)$ linking X_i to Y_i is the subset of observations that hold an appearance descriptor that is close enough, in the appearance space, to the codebook vector associated to Y_i .

The inference algorithm we use to propagate information is currently the belief propagation (BP) algorithm [29], [34], [35], discussed in Section 5. BP works by exchanging *messages* between neighboring nodes. Each message carries the belief that the sending node has about the pose of the receiving node. Let us consider, for example, nodes X_3 and X_5 in the network of Fig. 3. Through the message that X_3 sends to X_5 , X_3 probabilistically votes for all the poses of X_5 that are *compatible* with its own pose; this compatibility is defined through the compatibility potential $\psi_{5,3}$. Through this exchange of messages, each feature probabilistically votes for all possible object configurations consistent with its pose density. During inference, a consensus emerges among the available evidence, leading to one or more consistent scene interpretations. The system never commits to specific feature correspondences, and is thus robust to substantial clutter and occlusions. After inference, the pose likelihood of the whole object can be read out of the top feature; if the object is present twice in a scene, the top feature density should present two major modes.

4 DENSITY REPRESENTATION

We opted for a *nonparametric* computational encoding of density functions, for both random variables and potentials. The idea behind nonparametric methods is to represent a density simply by the samples we see from

it. Formally, a density is represented by a set of *weighted* samples called *particles*. The probabilistic density in a region of space is given by the local density of the particles in that region. The continuous density function is accessed by assigning a kernel function to each particle, a technique generally known as *kernel density estimation* [36]. Evaluation is performed by summing the evaluation of all kernels. Sampling is performed by sampling from the kernel of a particle ℓ selected from $\mathbf{P}(\ell = i) \propto w^i$, where w^i is the weight of particle i .

The expressiveness of a single “nonparametric” kernel is generally limited; the kernel we use models location and orientation independently, and location/orientation components are both isotropic (see below). Nonparametric methods account for the simplicity of individual kernels by employing a large number of them: in this framework, a density will typically be supported by 500 particles.

Compared to traditional parametric methods, the nonparametric approach eliminates problems such as fitting of mixtures or the choice of a number of components. Also, no assumption concerning density shapes (normality, ...) has to be made.

4.1 Kernel Definition

Random variables and potentials are defined over the Special Euclidean group $SE(3) = \mathbb{R}^3 \times SO(3)$, where $SO(3)$ is the Special Orthogonal group (the group of 3D rotations). We use a kernel that factorizes into two functions defined on \mathbb{R}^3 and $SO(3)$, which is consistent with the policy of kernel simplicity in the nonparametric approach. Denoting the separation of $SE(3)$ elements into translations and rotations by

$$x = (\lambda, \theta), \quad \mu = (\mu_t, \mu_r), \quad \sigma = (\sigma_t, \sigma_r),$$

we define our kernel with

$$\mathbf{K}(x; \mu, \sigma) = \mathbf{N}(\lambda; \mu_t, \sigma_t) \cdot \mathbf{\Theta}(\theta; \mu_r, \sigma_r) \quad (1)$$

where μ is the kernel mean point, σ is the kernel bandwidth, $\mathbf{N}(\cdot)$ is a trivariate isotropic Gaussian kernel, and $\mathbf{\Theta}(\cdot)$ is an orientation kernel defined on $SO(3)$. Denoting by θ' and μ'_r the quaternion representations of θ and μ_r [37], we define the orientation kernel with the Dimroth-Watson distribution [38], [39]

$$\mathbf{\Theta}(\theta; \mu_r, \sigma_r) = \mathbf{W}(\theta'; \mu'_r, \sigma_r) = C_w(\sigma_r) \cdot e^{\sigma_r(\mu'_r{}^\top \theta')^2} \quad (2)$$

where $C_w(\sigma_r)$ is a normalizing factor. This kernel corresponds to a Gaussian-like distribution on $SO(3)$. The Dimroth-Watson distribution inherently handles the double cover of $SO(3)$ by quaternions [40]. Similar approaches have been explored in prior work [28].

The bandwidth σ associated to a density should ideally be selected jointly in \mathbb{R}^3 and $SO(3)$. However, this is difficult to do. Instead, we set the orientation bandwidth σ_r to a constant allowing about 10° of deviation; the location bandwidth σ_t is then selected using a k -nearest neighbor technique [36].

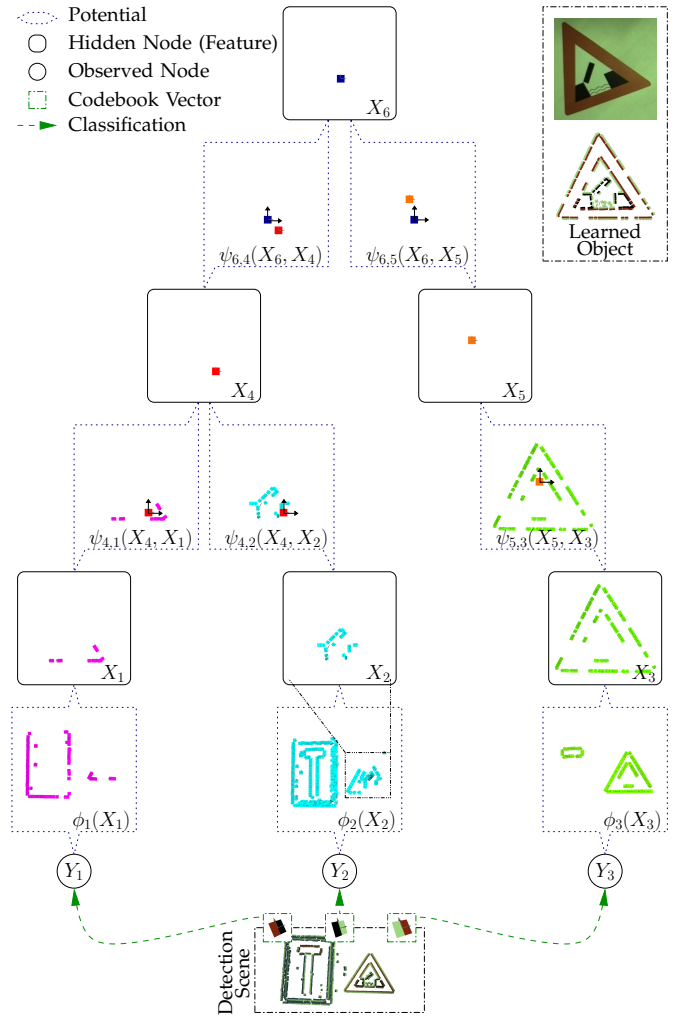


Fig. 4. Example of a hierarchical model of a traffic sign, with explicit potentials and feature pose densities.

4.2 Evidence

In Section 3.2, we explain that an observation potential is built from a set of input observations. Given the nonparametric representation, building an observation potential is straightforward: input observations are defined in 3D, they can thus directly be used as particles representing the observation potential. However, input observations may not provide a full $SO(3)$ orientation. For example, an ECV observation has an orientation parametrized by the local tangent to the 3D edge it is extracted from ($\theta \in S^2_+$); the orientation about this 3D edge is unspecified. Observation potentials will thus generally be defined on a subspace of $SE(3)$.

Section 5.1 and Section 5.2 describe inference in the general case of $SE(3)$ densities. Section 5.3 refine the mechanisms of Section 5.2 to handle a network in which potentials are defined on different domains.

4.3 A Complete Example

Fig. 4 shows in greater detail the hierarchy of Fig. 3. Feature 2 is a primitive feature that corresponds to a

local black-white edge segment (the white looks greenish in the picture). The blue patch pattern in the $\phi_2(X_2)$ box is the non-parametric representation of the evidence distribution for feature 2, which corresponds to black-white edge segments produced by the ECV system. The blue patch pattern in the X_2 box is the non-parametric representation of the posterior density of X_2 , i.e. the poses in which the part “feature 2” is likely to be found. Feature 4 corresponds to the opening-bridge drawing. Its inferred pose in the scene is shown by the red patch in the X_4 box. For clarity, only one patch is shown; the density would normally be represented by a large number of patches around that point. The $\psi_{4,2}(X_4, X_2)$ box shows the encoding of the relationship between features 4 and 2: for a fixed pose for feature 4 (in red), it shows the likely poses for feature 2 (in blue).

5 INFERENCE

Inference provides marginal pose densities for all features of the model, given the observations extracted from a scene. After inference, the top feature indicates where the learned object is likely to be present.

Inference is preceded by the definition of evidence for all features (hidden nodes) of the model (Section 3 and Section 4.2). Once these priors have been defined, inference can be carried out with any applicable algorithm. We currently use a belief propagation algorithm of which we give a complete, top-down view below.

5.1 Belief Propagation

In the case of a Markov tree, belief propagation (BP) [29], [34], [35] is based on an exchange of *messages* between neighboring hidden nodes. A message that feature i sends to feature j is denoted by $m_{ij}(X_j)$, and carries feature i 's belief about the state of feature j . To prepare a message for feature j , feature i starts by computing a “local pose belief estimate”, as the product of the local evidence and all incoming messages *but* the one that comes from j . This product is then multiplied with the compatibility potential of i and j , and marginalized over X_i . The complete message expression is

$$m_{ij}(X_j) = \int \psi_{ij}(X_i, X_j) \phi_i(X_i) \prod_{k \in N(i) \setminus j} m_{ki}(X_i) dX_i \quad (3)$$

where $N(i)$ denotes the set of features adjacent to feature i . When all messages have been exchanged once, a *belief* is computed for each feature as

$$b_j(X_j) = \frac{1}{Z} \phi_j(X_j) \prod_{i \in N(j)} m_{ij}(X_j), \quad (4)$$

where Z is a normalizing constant. The belief $b_j(X_j)$ corresponds to the posterior marginal pose density of feature j .

BP propagates the collected evidence from primitive features to the top of the hierarchy, permitting the inference of the object pose likelihood. Furthermore, the

global belief about the object pose is also propagated from the top node down the hierarchy, reinforcing globally consistent evidence and permitting the inference of occluded features.

5.2 Nonparametric Belief Propagation

Several instantiations of the BP algorithm for continuous, non-Gaussian potentials are available in the literature, e.g. NBP [17], PAMPAS [41], or more recently MSBP [24]. These methods support nonparametric density representations, and are applicable to our graphical model.

We are currently using the NBP algorithm. Since NBP tends to outperform PAMPAS for graphical models with multimodal potentials containing large numbers of kernels [17], [41], it is a good choice for our models where potentials linking primitive features to their parents are inherently multimodal and are usually represented with more than a hundred particles. Park et al. recently presented the MSBP algorithm [24], which uses mean-shift to perform nonparametric mode-seeking on belief surfaces generated within the belief propagation framework. While MSBP has been shown to outperform NBP in terms of speed and accuracy, the ability to compute complete posteriors with NBP is a strong advantage for some applications.

NBP is easier to explain if we decompose the analytical message expression (3) into two steps:

- 1) Computation of the local belief estimate:

$$\beta_{ts}(X_t) = \phi_t(X_t) \prod_{i \in N(t) \setminus s} m_{it}(X_t), \quad (5)$$

- 2) Combination of β_{ts} with the compatibility function ψ_{ts} , and marginalization over X_t :

$$m_{ts}(X_s) = \int \psi_{ts}(X_t, X_s) \beta_{ts}(X_t) dX_t. \quad (6)$$

NBP forms a message by first sampling from the product (5) to collect a non-parametric representation of $\beta_{ts}(X_t)$, then sampling from the integral (6) to collect a non-parametric representation of $m_{ts}(X_s)$.

Sampling from the message product (5) is a complex task for which Ihler et al. explored multiple efficient solutions [42], one of which is to sample from the product using importance sampling (IS). Importance sampling can be used to produce a nonparametric representation of an unknown distribution $\mathbf{p}(x)$ by correctly weighting samples from a known distribution $\mathbf{q}(x)$: IS accounts for the difference between the *target distribution* $\mathbf{p}(x)$ and the *proposal distribution* $\mathbf{q}(x)$ by assigning to each sample x^ℓ a weight defined as $w^\ell = \mathbf{p}(x^\ell)/\mathbf{q}(x^\ell)$. Since $\mathbf{p}(x)$ is often rather different from $\mathbf{q}(x)$, many samples may end up with a very small weight. To compute a set of n representative samples of $\mathbf{p}(x)$, one usually takes rn samples from $\mathbf{q}(x)$ ($r \geq 1$), computes their weights as defined above, and eventually resamples to a size of n . The closer $\mathbf{q}(x)$ is to $\mathbf{p}(x)$, the better $\{(x^\ell, w^\ell)\}$ will approximate $\mathbf{p}(x)$.

To sample from a message product (5) with IS, we use a proposal function that corresponds to the sum of the factors of the product; importance weights are computed as

$$w^\ell = \frac{\phi_t(x_t^\ell) \prod_{i \in \mathcal{N}(t) \setminus s} m_{it}(x_t^\ell)}{\phi_t(x_t^\ell) + \sum_{i \in \mathcal{N}(t) \setminus s} m_{it}(x_t^\ell)}.$$

The second phase of the NBP message construction computes an approximation of the integral (6) by stochastic integration. Stochastic integration takes a set of samples $\{x_t^i\}$ from $\beta_{ts}(X_t)$, and propagates them to feature s by sampling from $\psi_{ts}(x_t^i, X_s)$ for each x_t^i . It would normally also be necessary to take into account the marginal influence of $\psi_{ts}(X_t, X_s)$ on X_t . In our case however, potentials only depend on the difference between their arguments; the marginal influence is a constant and can be ignored. In NBP inference, the efficiency trade-off is largely governed by the number of particles n supporting density functions: while the success of NBP inference highly depends on a sufficient density resolution, increasing n has a hard impact on computational cost and memory footprint.

5.3 Analytic Messages

In Section 4.2, we mention that observation potentials may be defined on a subspace of $SE(3)$. This can effectively be represented in $SE(3)$ by setting the free dimensions to a uniform distribution. Each observation would be represented with multiple $SE(3)$ particles which, in the case of a completely non-oriented observation, would each be assigned a random orientation. However, the number of particles needed to account for as little as one degree of freedom in orientation without imposing too much noise on other dimensions may already be quite high, and will surely impede inference. Another way to approach the problem is to define observation-related densities on the specific $SE(3)$ subspace D on which observations are defined. For example, with completely non-oriented observations, the domain would simply be $D = \mathbb{R}^3$. This way, each observation can be represented by a single particle. Primitive features are the hidden counterpart of observations; if a different domain D is used for observation potentials, primitive features will also be defined on D . Finally, the domain of potentials linking primitive features to second-level features should be changed. Since a potential holds the spatial distribution of the child feature in a reference frame that corresponds to the pose of the parent feature, these potentials will also be defined on D . The next paragraphs show how *analytic messages* [28], [43] can be used to allow NBP to efficiently manage messages exchanged between pairs of variables defined on different domains.

Let us turn to the propagation equation (3), which we *analytically* decomposed into a multiplication (5) and an integration (6). We explained that NBP implements BP by *numerically* performing the same decomposition, i.e. computing explicit nonparametric representations for messages and local estimates alternately. Forming an

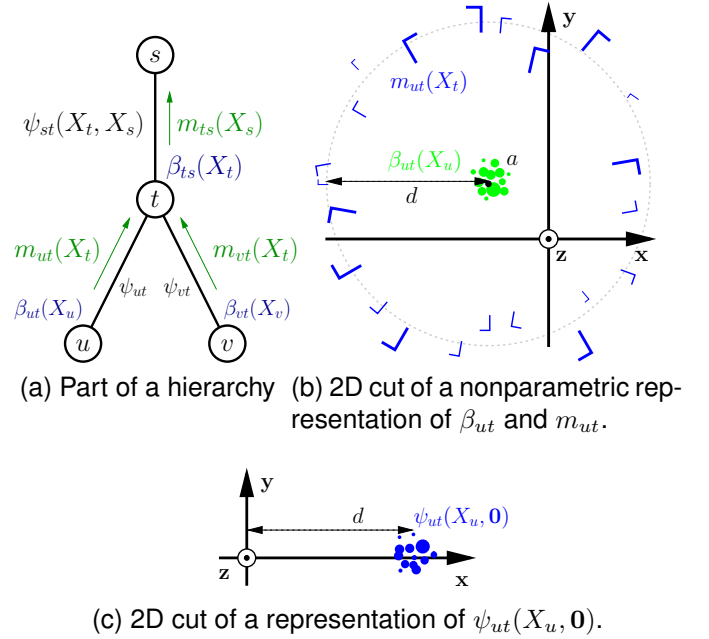


Fig. 5. Illustration of the issue of sending a message from a feature u defined on $D \subset SE(3)$ to a feature t defined on $SE(3)$. We note that in Fig. (b), m_{ut} should be distributed on the surface of a sphere centered at a . For the purpose of this illustration, we only show the particles that are near the intersection of the sphere with the xy plane. Since X_t is defined on $SE(3)$, its particles have a full 3D orientation. Given the information contained in β_{ut} and ψ_{ut} , we know that the particles in m_{ut} should have their x axis oriented towards a . As for their y and z axis, there is no evidence. This degree of freedom is illustrated in Fig. (b) by the random direction of the smaller line of the frame picturing each particle.

explicit representation of a message sent from a feature u defined on $D \subset SE(3)$ to a feature t defined on $SE(3)$ unfortunately leads to representing in $SE(3)$ a distribution that is likely to span a large space. Let us consider the example of Fig. 5a where u is defined on $D = \mathbb{R}^3$ and is located at a distance d from t in the x direction. The potential $\psi_{ut}(X_u, X_t)$, which represents the distribution of u in a reference frame defined by t , will be a unimodal distribution centered at $(d, 0, 0)$ (Fig. 5c). Let us assume we know that u is located around a center point a , i.e. $\beta_{ut}(X_u)$ is a unimodal distribution centered at a . In forming $m_{ut}(X_t)$, all we can say is that t will be located about the surface of a sphere of radius d centered at a (Fig. 5b). In general, $m_{ut}(X_t)$ will be a lot more expensive to represent than $\psi_{ut}(X_u, X_t)$ and $\beta_{ut}(X_u)$ together. Yet, the representational cost will typically be much smaller for $\beta_{ts}(X_t)$ than for $m_{ut}(X_t)$, thanks to the disambiguation brought by $m_{vt}(X_t)$. The following paragraphs show how to avoid $m_{ut}(X_t)$ by using directly $\psi_{ut}(X_u, X_t)$ and $\beta_{ut}(X_u)$ in computing $\beta_{ts}(X_t)$.

Let us assume we are in the process of constructing a

nonparametric representation of $\beta_{ts}(X_t)$, i.e. the local estimate of feature t that includes all incoming information but that from s . In general IS-based NBP, we first form a proposal function $q(X_t)$ from the sum of incoming messages and local evidence; then, we repetitively take a sample x_t^ℓ from $q(X_t)$, and compute its importance weight by multiplying the evaluations of the incoming-message nonparametric representations at x_t^ℓ :

$$w^\ell = \frac{1}{q(x_t^\ell)} \cdot \phi_t(x_t^\ell) \prod_{i \in N(t) \setminus \{s\}} m_{it}(x_t^\ell). \quad (7)$$

However, this expression can be rewritten

$$w^\ell = \frac{\phi_t(x_t^\ell)}{q(x_t^\ell)} \prod_{i \in N(t) \setminus \{s\}} \int \psi_{it}(X_i, x_t^\ell) \beta_{it}(X_i) dX_i, \quad (8)$$

and w^ℓ can be computed directly from $\beta_{it}(X_i)$ and $\psi_{it}(X_i, X_t)$ ($i \in N(t) \setminus \{s\}$), i.e. by using *analytic* messages [28], [43]. Evaluating each integral is achieved by sampling p times an example x_i^k from $\psi_{it}(X_i, x_t^\ell)$, evaluating $\beta_{it}(x_i^k)$, and taking the average over k .

When $\beta_{it}(X_i)$ is defined on a proper subspace of $SE(3)$, forming an explicit message representation to compute importance weights (7) would suffer from the issue illustrated in Fig. 5. In (8), we *evaluate* $\beta_{it}(x_i^k)$, which means that its degrees of freedom are taken into account *analytically*. We can thus afford to use a rather sparse proposal q , computed from explicit messages, because we then precisely weight these samples directly through potentials and local estimates.

5.4 Two-Level Importance Sampling

One known weakness of IS-based NBP is that it cannot intrinsically concentrate its attention on the modes of a product, which is an issue since individual messages often present many irrelevant modes [42]. We overcome this problem with a two-level IS: we first compute an intermediate representation of the product with the procedure explained above, we then use this very representation as the proposal function for a second IS that will be geared towards relevant modes. Let us denote by n the number of particles supporting a density. The intermediate representation is obtained with sparse analytic messages ($p \ll n$) but many importance samples ($r \gg 1$), while the second IS uses rich analytic messages ($p \approx n$) but a low value for r . We typically choose r and p such that $r \times p = n$. There are rn particles to weight in the proposal density. Denoting by d the number of messages in the product (5), weighting one particle (8) involves the evaluation of d densities at p points, the evaluation of the evidence, and the evaluation of the proposal. Density evaluation is logarithmic in the number of density particles if implemented with kd -trees [42]. The product of d analytic messages is thus

$$O(dn^2 \log n). \quad (9)$$

We note that comparing this with the algorithmic complexity of other NBP implementations would need to be

done with care, because the magnitude of n required to achieve a given resolution depends on the method.

The two-level IS described above and the product of analytic messages have proved crucial for the successful application to real-world objects such as those presented in Section 7.

6 LEARNING

Model learning is the process of building a hierarchy from a set of input observations. The information contained in a hierarchy is held in the topology of the network, in the compatibility potentials, and in the codebook. A learning procedure is responsible for defining these three elements; there is no additional constraint. We are primarily interested in unsupervised algorithms, which can be divided into top-down and bottom-up approaches. A top-down approach would recursively divide the set of available observations following structural or semantic criteria, each division giving rise to a new meta-feature. When reaching observation groups that are simple enough, primitive features would be formed and codebook vectors would possibly be extracted. A bottom-up approach works the other way around. Observations are first separated into k sets, each of which gives rise to a primitive feature. If appearance descriptors are used in the separation process, the codebook is trivially defined from the appearance of each set. To form the hierarchy, features are iteratively combined with one another based again on structural or semantic criteria. Previous work on top-down [8] and bottom-up [7], [9], [20] approaches is available in the literature.

In the next sections, we develop a bottom-up learning method which autonomously builds a tree-structured hierarchy from a set of observations extracted from a segmented object. The method was designed with simplicity in mind. Despite this simplicity, it allows us to produce hierarchies that can readily be used for object pose estimation, as shown in Section 7.

6.1 Primitive Features

Learning starts with a set of n observations

$$O = \{(x^i, w^i, a^i)\}_{i \in [1, n]}$$

where x^i contains geometric information, w^i is a weight, and a^i is an appearance descriptor. The geometric information includes a 3D location and an orientation. As discussed in previous sections, the orientation does not need to be a full 3D orientation; partially defined orientations also work. The appearance descriptor is defined in \mathbb{R}^d ($d \geq 0$), and it is assumed that the Euclidean metric in \mathbb{R}^d is a meaningful measure of the closeness of two descriptors. There is no particular requirement on the quality of the observations used for learning, but accurate observations naturally produce better models.

To define primitive features, we essentially divide O into k subsets. The division criterion we are currently

using is exclusively appearance-based. K -Means clustering is applied in the appearance space on the set of appearance descriptors $\{a^i\}$, producing k cluster centers A_1, \dots, A_k . These k cluster centers will directly form the codebook; each primitive feature i will correspond to a generic element characterized by an appearance A_i [7], [20].

The next section explains how meta-features are created. Meta-features encode relative spatial relationships. For the purpose of their creation, pose densities have to be available for all existing features. The pose distribution of each primitive feature i is computed from the spatial distribution of the observations that were associated to the i^{th} cluster during the k -means clustering. Since distributions are encoded in a nonparametric way, feature pose densities are computed simply by resampling the set of corresponding observations. These pose densities will only serve to build the hierarchy, they are discarded at the end of the learning process.

To illustrate the previous paragraphs, we can come back to Fig. 4, in which feature 3 is a generic, red-white edge segment. During the construction of the model, the density associated to feature 3 would show that this feature is mainly distributed along the outer edges of the traffic sign.

6.2 Meta-Features

After primitive features have been defined, the graph is built incrementally, one level at a time. The learning algorithm works by repeatedly grouping sets of features into a higher-level meta-feature. We are currently restricting the number of features that can be combined into a meta-feature to 2. Limiting the number of children of a meta-feature is a rather soft restriction though, since wider combination can intuitively be traded for deeper hierarchies.

A new level is built on top of the previous level, which we will refer to as the *supporting* level. Meta-features encode geometric configurations; when starting a new level, the pose of each feature of the supporting level is assumed to be already defined. The definition of primitive feature poses has been explained in the previous section; the definition of meta-feature poses is explained below. We note again that the purpose of defining feature pose densities is to extract *relative* configurations and define the potentials linking to the new level. Feature densities will be specific to the learning scene, they cannot help in the inference of the model in a different scene.

To create an additional level, the features of the supporting level are randomly grouped into pairs. (If there is an odd number of features, one will stand alone.) Each pair forms a meta-feature. The pose of a meta-feature in the scene can be defined in an arbitrary way. Since a meta-feature encodes the composition of its child features, its pose is defined as the mean of the densities of its children, which helps in interpreting the final

hierarchy. Each meta-feature i thus has a single pose x_i^p in the learning scene.

Compatibility potentials are then defined through the extraction of relative spatial relations between each new meta-feature and its children. The potential $\psi_{ij}(X_i, X_j)$ between meta-feature i and its child j is formed by repeatedly sampling from the relative transformation between i and j . Drawing a sample x_ψ from the relative transformation is achieved by drawing a sample x_j from feature j and combining it with the pose x_i^p of its parent i . Denoting the separation of pose variables into translations and rotations by

$$x_\psi = (\lambda_\psi, \theta_\psi), \quad x_j = (\lambda_j, \theta_j), \quad x_i^p = (\lambda_i, \theta_i),$$

the sample x_ψ from the transformation is formed from x_i^p and x_j as

$$\lambda_\psi = \theta_i^\top (\lambda_j - \lambda_i), \quad \theta_\psi = \theta_i^\top \theta_j,$$

where $SO(3)$ orientations are represented as rotation matrices for simplicity. In this last expression, x_ψ corresponds to the projection of x_j into the reference frame defined by x_i^p . The set $\{x_\psi\}$ represents the pose distribution of feature j in the reference frame defined by the pose of its parent i which actually corresponds to $\psi_{ij}(X_i = \mathbf{0}, X_j)$. During inference, to evaluate $\psi_{ij}(X_i = x_i, X_j = x_j)$, we first need to transform $\psi_{ij}(X_i = \mathbf{0}, X_j)$ into $\psi_{ij}(X_i = x_i, X_j)$ by applying the transformation defined by x_i on all the supporting particles

$$\lambda'_\psi = \theta_i \lambda_\psi + \lambda_i, \quad \theta'_\psi = \theta_i \theta_\psi$$

then evaluate $\psi_{ij}(X_i = x_i, X_j = x_j)$ with kernel density estimation on the set $\{x'_\psi = (\lambda'_\psi, \theta'_\psi)\}$. (For efficiency, we actually evaluate $\psi_{ij}(X_i = \mathbf{0}, X_j = x'_j)$ where x'_j is the result of the transformation of x_j by the inverse of the transformation defined by x_i .)

Finally, the pose of each new meta-feature is encoded in the feature pose density, to provide the next level-building iteration with spatial information. The pose of meta-feature i is encoded as a Dirac peak with a set of particles concentrated at x_i^p .

The procedure described above divides the number of nodes by two from one level to the next. It stops when there is only one node at the top level. The extraction of spatial relations constitutes its principal outcome. The complexity (number of features) of the models it produces is mainly influenced by color variety within objects. Objects with a larger color variety will give rise to more primitive features, which in turn leads to wider and taller hierarchies.

We note that speaking about the pose of an object implies that a *reference pose* (Fig. 6) has been defined for that object. Within a hierarchical model, the reference pose of an object is defined through the pose x_t^p associated to the top feature t during model learning.

7 EVALUATION

This section illustrates the applicability of our model to pose estimation.

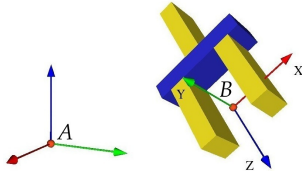


Fig. 6. Reference pose of an object. The reference frame marked A corresponds to the coordinate system of the scene. The relative configuration of the frame B and the gripper is the *reference pose* of the gripper. The *pose* of the gripper corresponds to the pose of B in A , i.e. the transformation between A and B .

7.1 Early Cognitive Vision

In this work, observations are provided by the ECV system presented in Section 1, which extracts local 3D edge segments from stereo imagery. Each edge segment has an appearance descriptor composed of two colors. The orientation θ associated to these observations is a 3D line ($\theta \in S_+^2$). The space on which observation-related densities are defined is $D = \mathbb{R}^3 \times S_+^2$. The Dimroth-Watson kernel (2) is directly applicable in S_+^2 , and the kernel we use on D is

$$\mathbf{K}_D((\lambda, \theta); \mu, \sigma) = \mathbf{N}(\lambda; \mu_t, \sigma_t) \cdot \mathbf{W}(\theta; \mu_r, \sigma_r),$$

with $\theta \in S_+^2$, $\mu = (\mu_t, \mu_r)$, and $\sigma = (\sigma_t, \sigma_r)$.

Fig. 1 illustrates typical ECV reconstructions. The system can work in two different modes. In the first mode (Fig. 1, left), it produces an ECV reconstruction from a single stereo view of a scene. In the second mode (Fig. 1, right), it integrates information through a set of stereo views to produce a reconstruction that is much less affected by self-occlusions [19], [44]. We will be referring to the latter as *accumulated* reconstruction. ECV accumulated reconstructions are typically computed from a dozen stereo views.

We usually provide the ECV system with stereo images of 1280×960 pixels. In the following illustrations, we only show the left image of a stereo pair.

7.2 Pose Estimation

Model learning is ideally done from an accumulated ECV reconstruction. Learning from single-view ECV reconstructions is also possible, but the resulting models are less robust to large viewpoint changes. Detection is always performed on a single view (see Fig. 2). Pose estimates are inferred by sending messages from primitive features to the top feature. Each feature sends one message to each of its parents. Inference produces an object pose likelihood in the density associated to the top feature of the object model. When instantiating a model in a scene in which exactly one instance of the object is present, the top feature density should generally present a single mode showing the pose of the object. However, if an object very similar to the one we are searching for appears in the scene, the top density will present a

second, weaker mode. Consequently, densities have to be clustered to extract prominent modes. We are currently limiting experiments to scenes which include a known number p of object instances. If p pose estimates have to be produced, the p largest modes are selected.

Visualizing a pose estimate is done through top-down propagation. To visualize a pose estimate x , we fix the evidence of the top feature to x , and we propagate this information to the bottom of the graph to produce a nonparametric representation of primitive feature densities. Using camera calibration parameters, particles from these densities are projected onto the 2D images, as in Fig. 7b. We can then easily verify that the projection matches the object we are looking for. We usually remove primitive feature evidence before down-propagating an estimate, which allows down-propagation to clearly reveal occluded object portions (Fig. 7, 9–11). Alternatively, if evidence is kept, primitive feature posteriors show particles that are coherent with both the model and scene evidence (Fig. 12, second and third images).

We typically use hierarchies of 2 or 3 levels, very similar to the example of Fig. 3. We fix the number of particles supporting feature densities and potentials to 400. We keep the number of particles in observation potentials n_e equal to the number of observations produced by ECV – typically 1000–5000 observations. It is necessary to use a larger number of particles in observation potentials because in a scene containing many different objects, only a small fraction of the particles comes from the object we are trying to detect. Having a larger number of particles in observation potentials is acceptable because this number only appears as its logarithm in the algorithmic complexity of inference (9). Inference within a hierarchy of M meta-features is $O(Mn^2 \log n_e)$ where $n < n_e$. We are developing a computer implementation of our learning and inference algorithms that, using a few reasonable approximations, allows us to detect hierarchies similar to that of Fig. 3 ($M = 3$) in about 10 seconds on a standard desktop computer. The memory footprint of the pose estimation process is always below 50MB. The cost of detecting multiple objects is linear in the number of objects.

7.3 Experimental Setups

This section presents experiments through which we evaluate the performance impact of various parameter setups. Our input data consist of a sequence showing a plastic pot undergoing a rotation of 360° [45]. The sequence contains 72 stereo frames; the object is rotated of 5° between two successive views (Fig. 7). We learned two models of the pot: one was learned from observations extracted from the first view of the sequence (Fig. 7a), another was learned from observations accumulated over 10 stereo pairs of the sequence. Both models were then instantiated in all views; the results are summarized in Fig. 8 and Tab. 1. The single view model only works for rotations of about 40° around the learning

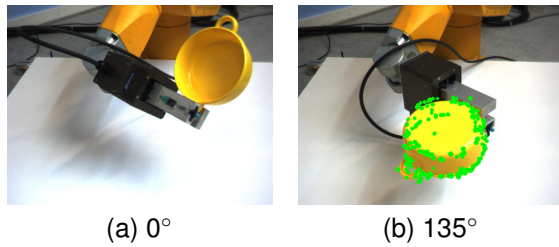


Fig. 7. Example frames from the rotating pot sequence. Fig. (b) shows a correct estimate of the pose of the pot obtained with a hierarchy learned from an accumulated ECV reconstruction. See text for details.

#	ECV	IS	n	time	suc. rate	loc. err.	ori. err.
1	acc.	2-lev	400	11s	66/72	10mm	4.9°
2	single	2-lev	400	11s	40/72	9mm	5.4°
3	acc.	1-lev	400	11s	41/72	18mm	13°
4	acc.	2-lev	200	3.4s	52/72	12mm	5.2°

TABLE 1

Running time per frame, success rate, and absolute error (location and orientation) for different experimental setups in the rotating pot sequence (Fig. 7), including learning from accumulated or single-view ECV, 2-level or 1-level importance sampling NBP, and 400 or 200 particles per density. Location and orientation errors are averaged over *successful* estimations.

view. When the opening of the pot faces away from the camera, the system fails to make a correct estimate. Fig. 8 shows that the model learned from accumulated observations performs better for a much wider range of viewpoints. Robustness drops around 90° and 270°, where the pot is viewed from the side; good stereopsis is then difficult to achieve and the ECV system is unable to produce a reliable 3D structure. As shown in Tab. 1, the average accuracy of successful estimates is similar for both models.

The third row of Tab. 1 summarizes the result of the detection of the model learned from accumulated ECV when a single-level importance sampling is used instead of the two-level IS of Section 5.4. For the same running time as the two-level IS, single-level IS presents a substantially lower detection rate, and the error in correct estimates is about twice as large.

The fourth row of Tab. 1 summarizes the result of the detection of the model learned from accumulated ECV with only half the number of particles of other setups. Pose estimates are less accurate; Fig. 8 shows that the system mainly loses robustness around difficult viewpoints (90° and 270°). Detection is however much faster.

7.4 Detection In Cluttered Scenes

To evaluate the robustness of our model to clutter and occlusions, we have computed the pose of various objects in several cluttered scenes [45], [46]. For technical

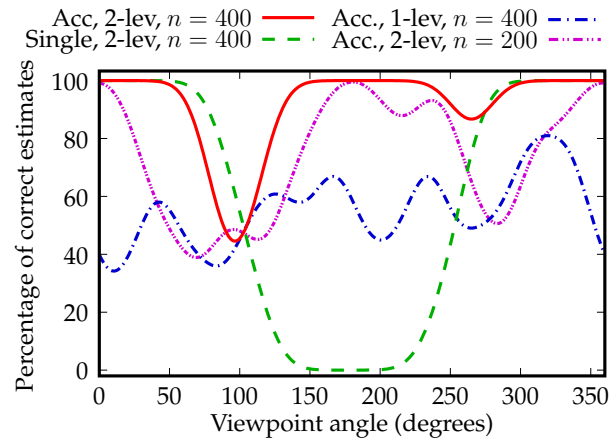


Fig. 8. Proportion of correct estimates in the rotating pot sequence (Fig. 7) as a function of the viewpoint. The 4 plots correspond to the 4 cases of Tab. 1.

reasons, most of the models were built from single-view ECV observations, or ECV observations accumulated from two views. Only the basket model was built from ECV observations accumulated through 10 stereo pairs. Fig. 9 shows a set of scenes for which all estimates are correct. The first three images of the top row demonstrate that our system can handle some object deformations. Small deformations are integrated through the soft probabilistic reasoning. Larger deformations (e.g. towel folding) are handled as occlusions.

Fig. 10 shows scenes in which an estimation failed. Failures are usually due to one of two reasons. One cause of failure is a poor ECV reconstruction, which may be due to stereopsis-related issues, low contrast, or poor image resolution. For example, in the third image of Fig. 10, the handle of the brush is almost horizontal which makes it almost impossible for ECV to find correct stereo matches. The other cause of failure is the presence in a scene of a structure that has the same geometry and appearance as the object. This is for example the case in the first image of Fig. 10, where half the pink kitchen cloth is occluded, and a better match is found in the pink structure of the towel.

Fig. 11 shows a similar experiment with traffic signs that share a common outer structure, but present variation in their central pattern. The fourth image illustrates the utility of color and of a hierarchy of multiple parts. It shows the same scene as the third image, but in greyscale. When color is not used, learned hierarchies degenerate into a two-feature network. All observations are associated to a single primitive feature, the geometry of the object is encoded in a single potential ψ . The number of particles of ψ that encode the pattern is small proportionally to the particles encoding the triangular frame, and their weight in the message sent through ψ is rather small. In that context, we observed that the pose likelihood of a sign generally presented several major modes, some corresponding to the correct sign, others centered on wrong signs. The largest estimate doesn't

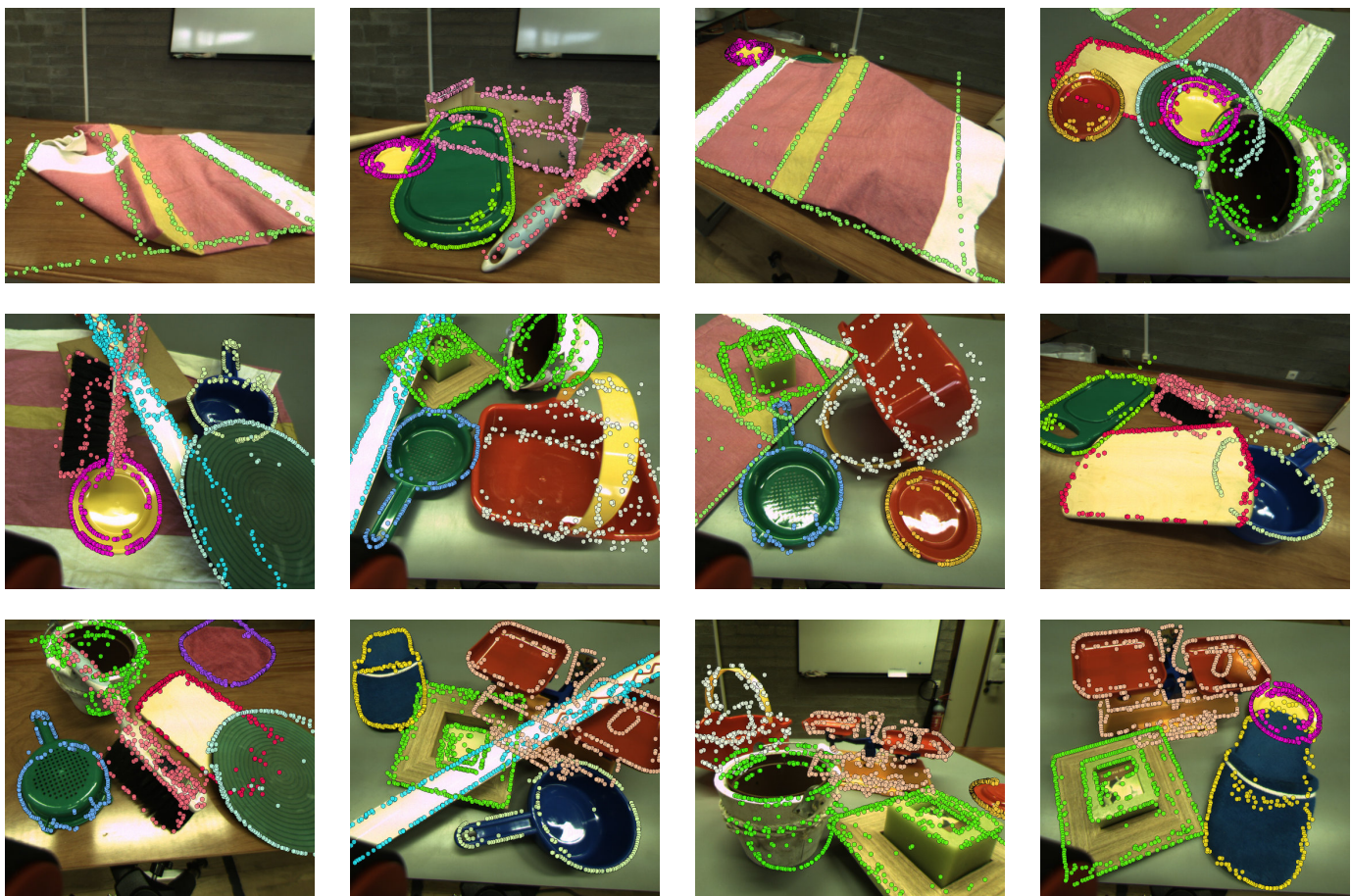


Fig. 9. Cluttered scenes with correct pose estimates for all objects. Colors are used to identify the object to which 2D-projected particles belong. See text for details.

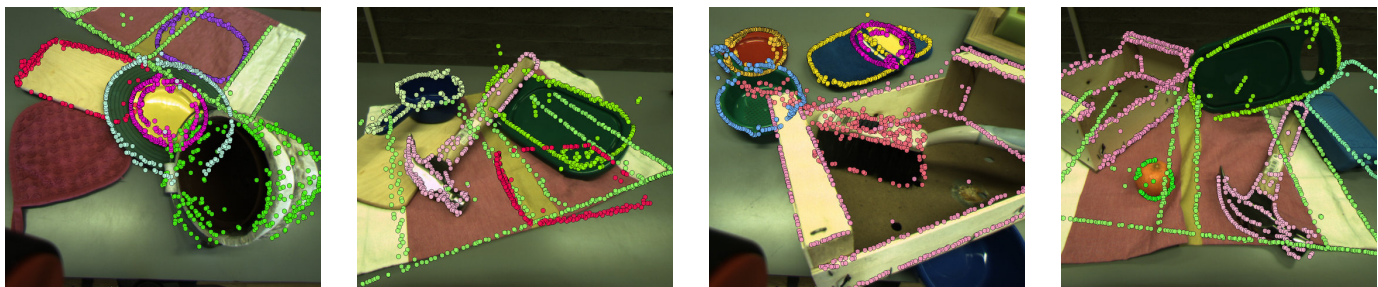


Fig. 10. Cluttered scenes where an estimation fails. Estimations failed respectively for the pink kitchen cloth, the breadboard, the brush, and the blue ice brick. Colors are used to identify the object to which 2D-projected particles belong. See text for details.

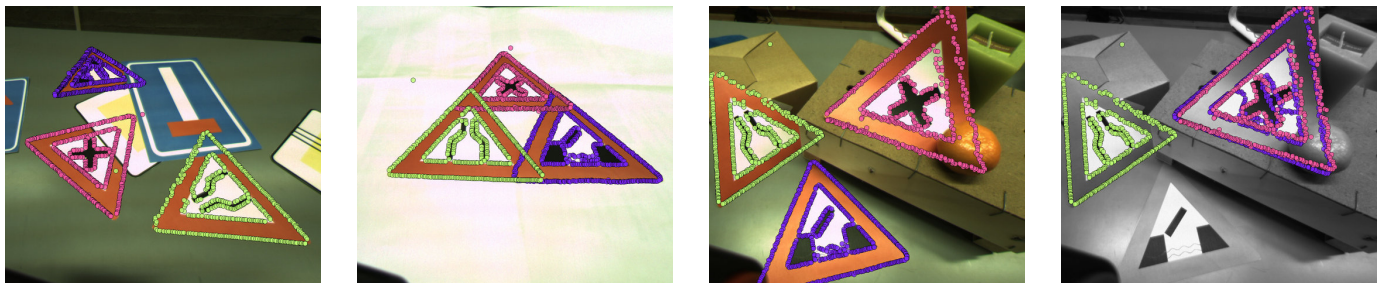


Fig. 11. Detection of objects of similar shapes. The last image shows pose estimates computed without taking color into account. See text for details.



Fig. 12. Detection of an articulated object. See text for details.

always correspond to the correct sign, as illustrated in the fourth image. Computing estimates in the first three images *without* using color led to only 4 correct poses out of 9. The results shown in these three images correspond to experiments where color is taken into account, all estimates are correct. When color is used, a multi-part hierarchy is built, and the central pattern is encoded in its own potential; during inference, it can contribute to the correct pose through a distinct message, which effectively discards incorrect modes.

7.5 Articulated Objects

This section explains how this framework can be used to represent loose objects by providing training data that reflect articulations. We chose to use a toy weighing scale (Fig. 12). Our training data are a set of ECV observations extracted from several images exclusively showing the scale. The pose of the camera and toy scale are identical in all images, but each image shows the scale in a different equilibrium position. A model learned from such data will represent all equilibrium configurations simultaneously and indistinctly. Fig. 12 shows pose estimates obtained from this model. The first image corresponds to a top-down propagation with prior deletion of primitive feature evidence (see Section 7.2); it clearly shows the different configurations supported by the model. The other two images correspond to a top-down propagation including all evidence, which reveals object evidence that effectively contributed to the final pose estimate.

8 CONCLUSION AND FUTURE WORK

We presented an object representation framework that encodes probabilistic relations between 3D features organized in a Markov-tree hierarchy. We started with a discussion of nonparametric function representation in the Special Euclidean group $SE(3)$. We then showed how an object can be detected in a novel scene, and how to build a hierarchy for a given object. We finally demonstrated the applicability of our method through a set of pose estimation experiments.

Detection largely amounts to inferring the pose of the top feature using nonparametric belief propagation. We detailed an importance sampling implementation of NBP message products, along with means of efficiently supporting networks that mix different density domains. Inference produces a *likelihood* of the object pose in the entire scene.

Learning is currently handled by a simple procedure which first clusters object observations according to their appearance, then randomly combines these to form a hierarchy. The learning procedure leaves room for improvement, and we plan to explore several approaches in the near future. Yet, despite its simplicity, the current method is able to produce models that allow for rather robust pose estimation.

Our method is readily applicable to object pose estimation. Experiments demonstrate a high level of robustness to input noise, and show that a satisfactory accuracy can be reached within reasonable computational constraints. We are thus able to achieve pose recovery without prior object models, and without explicit point correspondences.

Our method can in principle incorporate features from other perceptual modalities than vision. Our objective is to observe haptic and kinematic features that correlate with successful grasps, and integrate them into the feature hierarchy. Grasps will generally not relate to an object as a whole. Our part-based representation offer an elegant way to link grasps to the 3D visual features that predict their applicability, and *learn* visual features that permit the generalization of grasps across objects. Grasps success likelihoods can then emerge from visual evidence through top-down inference.

ACKNOWLEDGMENTS

The authors warmly thank Prof. Norbert Krüger for his tireless support, and for very fruitful discussions. The authors also thank Dirk Kraft for providing us with object imagery annotated with ground-truth motion.

This work was supported by the Belgian National Fund for Scientific Research (FNRS) and the EU Cognitive Systems project PACO-PLUS (IST-FP6-IP-027657).

REFERENCES

- [1] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.
- [2] E. Bienenstock, S. Geman, and D. Potter, "Compositionality, MDL priors, and object recognition," in *Advances in Neural Information Processing Systems*, 1996.
- [3] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nat. Neurosci.*, 1999.
- [4] S.-C. Zhu, "Embedding gestalt laws in markov random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 11, pp. 1170–1187, 1999.
- [5] T. S. Lee and D. Mumford, "Hierarchical Bayesian inference in the visual cortex," *Journal of the Optical Society of America*, pp. 1434–1448, 7 2003.
- [6] Z. Tu, X. Chen, A. L. Yuille, and S.-C. Zhu, "Image parsing: Unifying segmentation, detection, and recognition," *Int. J. Comput. Vision*, vol. 63, no. 2, pp. 113–140, 2005.
- [7] G. Bouchard and B. Triggs, "Hierarchical part-based visual object categorization," in *Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 710–715.
- [8] B. Epshtein and S. Ullman, "Feature hierarchies for object classification," in *IEEE International Conference on Computer Vision*, 2005.
- [9] S. Fidler and A. Leonardis, "Towards scalable representations of object categories: Learning a hierarchy of parts," in *Computer Vision and Pattern Recognition*, 2007.

- [10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient matching of pictorial structures," in *Computer Vision and Pattern Recognition*, 2000, pp. 2066–.
- [11] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints," *Int. J. Comput. Vision*, vol. 66, no. 3, pp. 231–259, 2006.
- [12] A. Kushal and J. Ponce, "Modeling 3D objects from stereo views and recognizing them in photographs," in *European Conference on Computer Vision*, 2006, pp. 563–574.
- [13] J. Rodgers, D. Anguelov, H.-C. Pang, and D. Koller, "Object pose detection in range scan data," in *Computer Vision and Pattern Recognition*, 2006.
- [14] P. Yan, S. M. Khan, and M. Shah, "3D model based object class detection in an arbitrary view," in *International Conference on Computer Vision*, 2007, pp. 1–6.
- [15] S. Savarese and L. Fei-Fei, "3D generic object categorization, localization and pose estimation," in *International Conference on Computer Vision*, Oct. 2007, pp. 1–8.
- [16] J. Liebelt, C. Schmid, and K. Schertler, "Viewpoint-independent object class detection using 3D feature maps," in *Computer Vision and Pattern Recognition*, jun 2008.
- [17] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation," in *Computer Vision and Pattern Recognition*, Los Alamitos, CA, USA, 2003.
- [18] N. Krüger and F. Wörgötter, "Multi-modal primitives as functional models of hyper-columns and their use for contextual integration," in *BVAI*, ser. Lecture Notes in Computer Science, M. D. Gregorio, V. D. Maio, M. Frucci, and C. Musio, Eds., vol. 3704. Springer, 2005, pp. 157–166.
- [19] N. Pugeault, *Early Cognitive Vision: Feedback Mechanisms for the Disambiguation of Early Visual Representation*. Verlag Dr. Müller, 2008.
- [20] F. Scalzo and J. H. Piater, "Statistical learning of visual feature hierarchies," in *Workshop at Computer Vision and Pattern Recognition*, Washington, DC, USA, 2005.
- [21] R. Detry and J. H. Piater, "Hierarchical integration of local 3D features for probabilistic pose recovery," in *Robot Manipulation: Sensing and Adapting to the Real World (Workshop at Robotics, Science and Systems)*, 2007.
- [22] R. Detry, N. Pugeault, and J. H. Piater, "Probabilistic pose recovery using learned hierarchical object models," in *International Cognitive Vision Workshop (Workshop at the 6th International Conference on Vision Systems)*, 2008.
- [23] G. Hua, M.-H. Yang, and Y. Wu, "Learning to estimate human pose with data driven belief propagation," in *Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 747–754.
- [24] M. Park, Y. Liu, and R. Collins, "Efficient mean shift belief propagation for vision tracking," in *Computer Vision and Pattern Recognition*, June 2008, (to appear).
- [25] J. Zhang, Y. Liu, J. Luo, and R. Collins, "Body localization in still images using hierarchical models and hybrid search," in *Computer Vision and Pattern Recognition*, 2006, pp. 1536–1543.
- [26] L. Sigal, Y. Zhu, D. Comaniciu, and M. J. Black, "Tracking complex objects using graphical object models," in *First International Workshop on Complex Motion*, 2004, pp. 223–234.
- [27] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard, "Tracking loose-limbed people," in *Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. I-421–I-428 Vol.1.
- [28] E. B. Sudderth, "Graphical models for visual object recognition and tracking," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006, adviser-William T. Freeman and Adviser-Alan S. Willsky.
- [29] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- [30] J. Coughlan and H. Shen, "Dynamic quantization for belief propagation in sparse spaces," *Comput. Vis. Image Underst.*, vol. 106, no. 1, pp. 47–58, 2007.
- [31] M. Toews and T. Arbel, "Detecting and localizing 3D object classes using viewpoint invariant reference frames," in *International Conference on Computer Vision*, 2007, pp. 1–8.
- [32] I. Gordon and D. G. Lowe, "What and where: 3D object recognition with accurate pose," in *Toward Category-Level Object Recognition*, 2006, pp. 67–82.
- [33] J. Piater, F. Scalzo, and R. Detry, "Vision as inference in a hierarchical markov network," in *Twelfth International Conference on Cognitive and Neural Systems*, 2008.
- [34] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," Mitsubishi Electric Research Laboratories, Tech. Rep., 2002.
- [35] M. I. Jordan and Y. Weiss, "Graphical models: Probabilistic inference," in *The Handbook of Brain Theory and Neural Networks*, 2nd edition, M. Arbib, Ed. MIT Press, 2002.
- [36] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986.
- [37] J. Kuffner, "Effective sampling and distance metrics for 3D rigid body path planning," in *Int'l Conf. on Robotics and Automation*. IEEE, May 2004.
- [38] K. V. Mardia and P. E. Jupp, *Directional Statistics*, ser. Wiley Series in Probability and Statistics. Wiley, 1999.
- [39] I. L. Dryden, "Statistical analysis on high-dimensional spheres and shape spaces," *ArXiv Mathematics e-prints*, Aug. 2005.
- [40] C. de Granville, J. Southerland, and A. H. Fagg, "Learning grasp affordances through human demonstration," in *International Conference on Development and Learning*, 2006.
- [41] M. Isard, "PAMPAS: Real-valued graphical models for computer vision," in *Computer Vision and Pattern Recognition*, 2003, pp. 613–620.
- [42] A. T. Ihler, E. B. Sudderth, W. T. Freeman, and A. S. Willsky, "Efficient multiscale sampling from products of Gaussian mixtures," in *Neural Information Processing Systems*, 2003.
- [43] A. T. Ihler, J. W. Fisher III, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-calibration in sensor networks," *IEEE Journal of Selected Areas in Communication*, vol. 23, no. 4, pp. 809–819, 2005.
- [44] D. Kraft, N. Pugeault, E. Başeski, M. Popović, D. Kragic, S. Kalkan, F. Wörgötter, and N. Krüger, "Birth of the Object: Detection of Objectness and Extraction of Object Shape through Object Action Complexes," *Special Issue on "Cognitive Humanoid Robots" of the International Journal of Humanoid Robotics*, 2008, (accepted).
- [45] D. Kraft and N. Krüger, "Object sequences," 2009. [Online]. Available: <http://www.mip.sdu.dk/covig/sequences.html>
- [46] R. Detry and J. Piater, "A probabilistic framework for 3D visual object representation: Experimental data," 2009. [Online]. Available: <http://intelsig.org/publications/Detry-2009-PAMI/>



Renaud Detry earned his engineering degree at the University of Liège in 2006. He is currently a Ph.D. student at the University of Liège, working under the supervision of Prof. Justus Piater on multimodal object representations for autonomous, sensorimotor interaction. He has contributed probabilistic representations and methods for multi-sensory learning that allow a robotic agent to infer useful behaviors from perceptual observations.



Nicolas Pugeault obtained an M.Sc. from the University of Plymouth in 2002 and an Engineer degree from the Ecole Supérieure d'Informatique, Électronique, Automatique (Paris) in 2004. He obtained his Ph.D. from the University of Göttingen in 2008, and is currently working jointly as a lecturer at the University of Southern Denmark and as a research associate at the University of Edinburgh.



Justus H. Piater is a professor of computer science at the University of Liège, Belgium, where he directs the Computer Vision research group. His research interests include computer vision and machine learning, with a focus on visual learning, closed-loop interaction of sensorimotor systems, and video analysis. He has published about 80 technical papers in scientific journals and at international conferences. He earned his Ph.D. degree at the University of Massachusetts Amherst, USA, in 2001, where he held a Ful-

bright graduate student fellowship, and subsequently spent two years on a European Marie-Curie Individual Fellowship at INRIA Rhône-Alpes, France, before joining the University of Liège.