Probabilistic Pose Recovery Using Learned Hierarchical Object Models

Renaud Detry, Nicolas Pugeault, Justus Piater

Probabilistic Pose Recovery Using Learned Hierarchical Object Models

Renaud Detry* — **Nicolas Pugeault**** — **Justus Piater***

* *Université de Liège, Liège, Belgium,*
Renaud.Detry@ULg.ac.be, Justus.Piater@ULg.ac.be

** *University of Southern Denmark, Odense, Denmark,*
The University of Edinburgh, Edinburgh, Scotland, UK,
npugeaul@inf.ed.ac.uk

ABSTRACT. *This paper presents a probabilistic representation for 3D objects, and details the mechanism of inferring the pose of real-world objects from vision. Our object model has the form of a hierarchy of increasingly expressive 3D features, and represents 3D relations between these probabilistically. Features at the bottom of the hierarchy are bound to local perceptions. While we currently only use visual features, our method can in principle incorporate features from diverse modalities within a coherent framework. Model instances are detected using a Nonparametric Belief Propagation algorithm which propagates evidence through the hierarchy to infer globally consistent poses for every feature of the model. We present an importance-sampling mechanism for belief updates that is critical for efficient and precise propagation. We finally present a series of pose estimation experiments on real objects, along with quantitative performance evaluation.*

RÉSUMÉ. *Ce texte présente une représentation probabiliste pour objets 3D, et détaille le mécanisme d'inférence de pose d'objets réels à partir d'observations visuelles. Notre modèle d'objet se présente comme une hiérarchie de caractéristiques 3D de plus en plus expressives, et représente de manière probabiliste les relations 3D entre ces dernières. Les caractéristiques au niveau inférieur de la hiérarchie sont associées à des perceptions locales. Nous limitions l'usage actuel aux caractéristiques visuelles ; cependant, le principe de notre méthode permet intrinsèquement d'incorporer des caractéristiques de diverses modalités au sein d'une même représentation. Les instances d'un modèle sont détectées à l'aide d'un algorithme de propagation de croyances non-paramétrique, qui propage l'information observée au travers de la hiérarchie pour inférer une pose globalement cohérente pour chaque caractéristique du modèle. Nous présentons un mécanisme de mise à jour de croyance par échantillonage par importance ; nous*

1re soumission à *International Cognitive Vision Workshop (ICVW) 2008*, le 10/02/2008

présentons également une série d'expérience d'estimation de pose d'objets réels, ainsi qu'une évaluation quantitative des performances atteintes.

KEYWORDS: *Computer vision, 3D object representation, pose estimation, Nonparametric Belief Propagation.*

MOTS-CLÉS : *Vision par ordinateur, représentation d'objets 3D, estimation de pose, Nonparametric Belief Propagation.*

1. Introduction

Representations of objects as configurations of parts have many potential advantages. Part-based representations are more robust to occlusions and viewpoint changes than global representations, and spatial configurations increase their expressiveness. Moreover, they not only allow bottom-up inference of object parameters based on features detected in images, but also top-down inference of image-space appearance based on object parameters.

The advantages of visual part-based representations naturally extend to multisensory cases. For example, haptic and proprioceptive information won't relate to an object as a whole. Instead, they typically emerge from specific grasps, on specific parts of the object. Part-based representation offer a neat way to *locally* encode cross-modal descriptions that emphasise the relations bewteen the different types of percepts.

We are currently developing a framework for object representation that combines local appearance and 3D spatial relationships, along with mechanisms for unsupervised learning and probabilistic inference of the model.

Our model has the form of a hierarchy. Features at the bottom of the hierarchy are bound to local visual perceptions. Pairs of correlated features are iteratively grouped into higher-level meta-features that encode probabilistic relative spatial relationships between their children.

To detect instances of a model in a cluttered scene, evidence is propagated throughout the hierarchy by probabilistic inference mechanisms, leading to one or more consistent scene interpretations.

In previous work (Detry *et al.*, 2007), we presented a learning method that constructs a hierarchy from a set of object observations. We also gave an overview of an inference process that followed a straightforward Nonparametric Belief Propagation scheme (Sudderth *et al.*, 2003) and allowed pose recovery of artificial objects. In this paper, we present in greater detail a significantly improved version of this inference process, along with practical considerations. We added an importance-sampling (IS) message product suggested in a similar form by Ihler et al. (Ihler *et al.*, 2003), and extended it to two-level IS sampling of *implicit* message products which is readily applicable to pose estimation on real-world objects.

The features organized in the hierarchies are not specially restricted to one input modality. We currently work with visual input only, but our model is intended to unite different types of perceptual information, e.g. vision plus haptic and proprioceptive inputs simultaneously. This will produce cross-modal descriptions and cross-modal behaviors directly applicable to robotic tasks such as grasping and object manipulation, as a grasp strategy may be linked directly to visual features that predict its applicability.

We emphasize that we are not developing an object classification framework. Object classification is best achieved using *discriminative* models and presupposes the presence of one object to be classified. Instead, we intend to develop *object-centric* representations that allow detection and localisation of known objects within a highly cluttered scene. Also, our representations lend themselves to applications other than mere classification (e.g. manipulation).

2. 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 spatial configurations of more elementary features, either meta or primitive. Thus, a meta-feature incarnates the relative configuration of two features from a lower level of the hierarchy.

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.

At the bottom of the hierarchy, primitive features are tagged with an appearance descriptor called a *codebook vector*. The set of all codebook vectors forms a *codebook* that binds the object model to the feature observations, by associating observations to primitive features.

Formally, the hierarchy is implemented using a Pairwise Markov Random Field. Feature i is associated to a hidden variable x_i, and the structure of the hierarchy is reflected by the edge pattern between them. Each meta-feature is thus linked to its two child features. Each primitive feature is also linked to an observed variable y_i.

When a model is associated to a particular scene (during construction or instantiation), features are associated to corresponding instances in that scene. The correspondence between a feature i and its instances is represented by the random variable x_i defined over the pose space $SE(3) = \mathbb{R}^3 \times SO(3)$.

As noted above, a meta-feature encodes the relationship between its two children. However, the graph records this information in a slightly different but equivalent way: instead of recording the relationship between the two child features, the graph records the two 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* $\psi_{ij}(x_i, x_j)$ associated to the edge e_{ij}. A compatibility potential specifies, for any given pair of poses of the features it links, the probability of finding

that particular configuration for these two features. We only consider rigid-body relationships. Moreover, relationships are *relative* spatial configurations. Compatibility potentials can thus be represented by a probability density over the feature–to–feature transformation space $SE(3)$.

Finally, the statistical dependency between a hidden variable x_i and its observed variable y_i is parametrized by an *observation potential* $\phi_i(x_i)$, also referred to as *evidence* for x_i, which corresponds to the spatial distribution of x_i's observations.

3. Inference

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 is likely to be present. Instantiating a model in a scene amounts to inferring posterior marginal densities for all features of the hierarchy.

The first step of inference is to define priors (observation potentials, evidence) for all features (hidden nodes) of the model. For primitive features, evidence is estimated from feature observations. Observations are classified according to the primitive feature codebook; for each primitive feature i, its observation potential $\phi_i(x_i)$ is estimated from observations that are (softly) associated to the i^{th} codebook vector. For meta-features, evidence is uniform.

Once priors have been defined, instantiation can be achieved by any applicable inference algorithms. We currently use a Belief Propagation algorithm of which we give a complete, top-down view below.

3.1. *Belief Propagation*

Belief Propagation (BP) (Pearl, 1988; Yedidia *et al.*, 2002; Jordan *et al.*, 2002) is based on incremental updates of marginal probability estimates, referred to as *beliefs*. The belief at feature i is denoted by

$$b(x_i) \approx \mathbf{p}(x_i|y) = \int ... \int \mathbf{p}(x_1, ..., x_N|y)\, dx_1 ... dx_{i-1} dx_{i+1} ... dx_N$$

where y stands for the set of observations. During the execution of the algorithm, *messages* are exchanged between neighboring features (hidden nodes). A message that feature i sends to feature j is denoted by $m_{ij}(x_j)$, and contains feature i's belief about the state of feature j. In other words, $m_{ij}(x_j)$ is a real positive function proportional to feature i's belief about the plausibility of finding feature j in pose x_j. Messages are exchanged until all beliefs converge, i.e. until all messages that a node receives predict a similar state.

At any time during the execution of the algorithm, the current pose belief (or marginal probability estimate) for feature i is the normalized product of the local evidence and all incoming messages, as

$$b_i(x_i) = \frac{1}{Z}\phi_i(x_i) \prod_{j \in \text{neighbors}(i)} m_{ji}(x_i), \quad [1]$$

where Z is a normalizing constant. 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

$$m_{ij}(x_j) = \int \psi_{ij}(x_i, x_j)\phi_i(x_i) \prod_{k \in \text{neighbors}(i) \setminus j} m_{ki}(x_i)dx_i. \quad [2]$$

As we see, the computation of a message doesn't directly involve the complete local belief [1]. In general, the explicit belief for each node is computed only once, after all desirable messages have been exchanged.

When BP is finished, collected evidence has been propagated from primitive features to the top of the hierarchy, permitting inference of marginal pose densities at top-level features. Furthermore, regardless of the propagation scheme (message update order), the iterative aspect of the message passing algorithm ensures that global belief about the object pose – concentrated at the top nodes – has at some point been propagated back down the hierarchy, reinforcing globally consistent evidence and permitting the inference of occluded features. While there is no theoretical proof of BP convergence for loopy graphs, empirical success has been demonstrated in many situations.

3.2. *Nonparametric Representation*

We opted for a nonparametric approach to probability density representation for all entities of the model, i.e. random variable and functions of random variables, including potentials, messages, and evidence. A density is simply represented by a set of (possibly weighted) particles; the local density of these particles in a given region is proportional to the actual probabilistic density in that region. The number of particles supporting a density is fixed, and will be denoted by n. Whenever a density has to be evaluated, traditional kernel density estimation methods can be used. Compared to usual parametric approaches that involve a limited number of parametrized kernels, a nonparametric approach eliminates problems like fitting of mixtures or the choice of a number of components. Also, no assumption concerning the shape of the density has to be made.

Figure 1 shows an example of a hierarchy for a traffic sign. Feature 2 is a primitive feature that corresponds to a local black-white edge segment – the white looks greenish on the picture. The blue patch pattern in the $\phi_2(x_2)$ box is the non-parametric

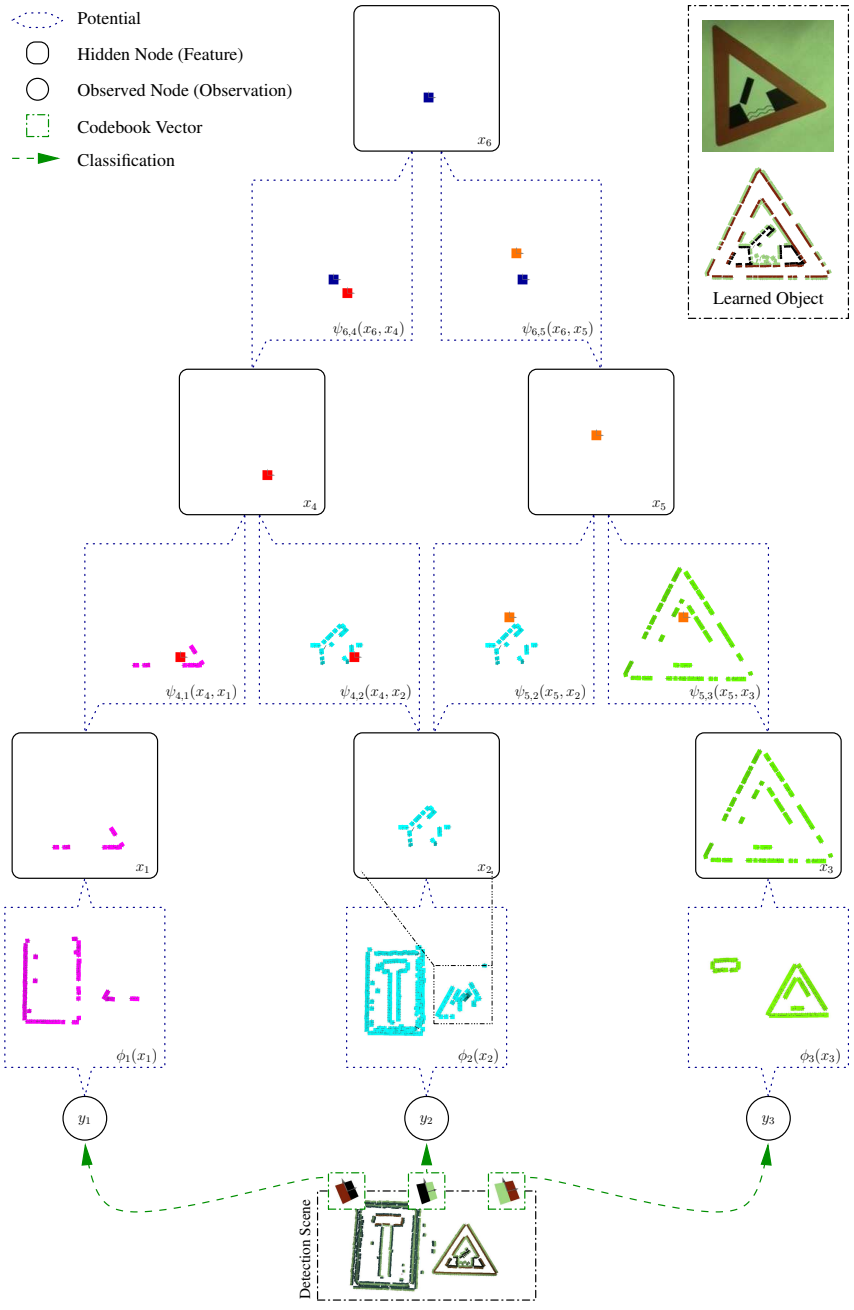


Figure 1. *Example of a hierarchical model of a traffic sign.*

representation for the evidence distribution for feature 2. The blue patch pattern in the x_2 box is the non-parametric representation for the posterior density of x_2, i.e. the poses in which the *traffic sign parts* "feature 2" are likely to be found. Feature 4 is the combination of primitive features 1 and 2. The red patch in the x_4 box shows its inferred pose in the scene. 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). The sign itself corresponds to feature 6, denoted by its random variable x_6. It is the composition of two features, one representing the central "opening bridge" pattern *and* the corners of the inner triangle (feature 4), the other representing the central pattern *and* the outer edges (feature 5).

3.3. *Nonparametric Belief Propagation*

For inference, we use a variant of BP, Nonparametric Belief Propagation (NBP), an algorithm for BP message update in the particular case of continuous, non-Gaussian potentials (Sudderth *et al.*, 2003). The underlying method is an extension of particle filtering; the representational approach is thus nonparametric and fits our model very well.

NBP is easier to explain if we decompose the analytical message expression [2] into two steps:

1) Computation of the local belief estimate

$$\beta_{ts}(x_t) = \phi_t(x_t) \prod_{i \in N(t) \setminus s} m_{it}(x_t), \qquad [3]$$

2) Combination of β_{ts} with the compatibility function ψ_{ts}, and marginalisation over x_t

$$m_{ts}(x_s) = \int \psi_{ts}(x_t, x_s) \beta_{ts}(x_t) dx_t. \qquad [4]$$

NBP forms a message by first sampling from the product [3] to collect a non-parametric representation for $\beta_{ts}(x_t)$, it then samples from the integral [4] to collect a non-parametric representation for $m_{ts}(x_s)$. These two operations are executed alternately: transform local estimate to form a message, merge messages to form a local estimate, etc...

Sampling from the message product [3] is conceptually straightforward. Using Gaussian kernel density estimation, each factor (messages and evidence) can be represented by a weighted sum of n Gaussians. The product of a series of Gaussians is also a Gaussian, and the parameters (mean, variance, weight) of the product Gaussian can easily be computed from the parameters of the factor Gaussians. Hence, letting $d = (N(t) - 1) + 1$ denote the number of factors in the product [3], $\beta_{ts}(x_t)$ can be expressed as a weighted sum of n^d Gaussians (Sudderth *et al.*, 2003). A nonparametric representation for $\beta_{ts}(x_t)$ can thus be constructed by sampling from a mixture of

n^d Gaussians, which amounts to repetitively selecting one Gaussian at random and taking a random sample from it. The computational cost of this exhaustive approach is $O(n^d)$. Clearly, exhaustive product implementations will suffer from overly long computation times.

The second phase of the NBP message construction computes an approximation for the integral [4] by stochastic integration. Stochastic integration takes a series of samples $\hat{x}_t^{(i)}$ from $\beta_{ts}(x_t)$, and propagates them to feature s by sampling from $\psi_{ts}(\hat{x}_t^{(i)}, x_s)$ for each $\hat{x}_t^{(i)}$. 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.

3.4. *Importance Sampling*

The computational bottleneck of NBP clearly lies in message products. Ihler et al. explored multiple improvements over the exhaustive product (Ihler *et al.*, 2003), one of which is to sample from the product using Importance Sampling (IS). IS is a technique for sampling from an unknown distribution $p(x)$ by sampling a series of examples $\hat{x}^{(\ell)}$ from a known distribution $q(x)$ ideally similar to p. IS accounts for the difference between the target distribution p and the proposal distribution q by assigning to each sample a weight defined as

$$w^{(\ell)} = \frac{p(\hat{x}^{(\ell)})}{q(\hat{x}^{(\ell)})}.$$

To produce a sample of size n, one usually takes rn weighted examples from q, where $r > 1$, and eventually resamples them to a size of n. The closer q is to p, the better $\{\hat{x}^{(\ell)}\}$ will approximate p.

Sampling from a message product [4] with IS works by selecting one of the messages $m_{ut}(x_t)$ (or the evidence) as proposal distribution, the rest of the factors providing importance weights:

$$w^{(\ell)} \;=\; \frac{\phi_t(\hat{x}_t^{(\ell)}) \prod_{i \in N(t) \setminus s} m_{it}(\hat{x}_t^{(\ell)})}{m_{ut}(\hat{x}_t^{(\ell)})} \;=\; \phi_t(\hat{x}_t^{(\ell)}) \prod_{i \in N(t) \setminus \{s,u\}} m_{it}(\hat{x}_t^{(\ell)}).$$

IS produces n samples from a product of d factors in $O(rdn^2)$ time. From here on, we will consider that the number of neighbors a node may have is bounded and typically low, and ignore it in complexity statements. IS thus produces n samples from a product of d factors in $O(rn^2)$ time.

4. Efficient Importance Sampling of Message Products

The success of NBP inference highly depends on a sufficient density resolution, i.e. having enough particles to support the different modes of potentials, local estimates, and messages. Moving to more complex applications will generally require an increase of n, which has a hard impact on computational time and memory needs. This section presents a variant of the IS-based NBP algorithm that yields a significant improvement of the inference power without any memory impact. Its computational behavior is close to original IS-based NBP, with some interesting benefits.

4.1. *Representational Constraints*

As explained above, *A message that feature i sends to feature j – denoted by $m_{ij}(x_j)$ – contains feature i's belief about the state of feature j.* Feature i will often possess a rather inaccurate local estimate, e.g. at the beginning of propagation when each bottom feature receives observations from the whole scene surrounding an object of interest. Additionally, even if a local estimate was exact, transforming it with ψ_{ij} will generate a large number of possible states for feature j, only a fraction of which will eventually become confirmed by other messages incoming to j – the job of message products precisely is to extract sections that overlap between incoming messages. Generating a message from local estimates can be pictured as an exploration process, while merging messages together would be a confirmation/concentration process. From these observations, it intuitively follows that one may achieve better performance by increasing the resolution of messages only, leaving potentials and local estimates at their initial resolution.

4.2. *Implicit Messages*

Let us now turn to the propagation equation [2], which we *analytically* decomposed into a multiplication [3] and an integration [4]. We explained that NBP implements BP by *physically* performing the same decomposition, i.e. computing explicit nonparametric representations for messages and local estimates alternately. In this section, we propose a somewhat different implementation, in which explicit representations are only computed for local estimates.

Let us assume we are in the process of constructing a nonparametric representation for $\beta_{ts}(x_t)$, i.e. the local estimate of feature t that includes all incoming information but that from s. In typical IS-based NBP, we first choose one incoming message $m_{ut}(x_t)$ at random ($u \neq s$) as IS proposal density; then, we repetitively take a sample $\hat{x}_t^{(\ell)}$ from $m_{ut}(x_t)$ and compute its importance weight

$$w^{(\ell)} = \phi_t(\hat{x}_t^{(\ell)}) \prod_{i \in N(t) \setminus \{s,u\}} m_{it}(\hat{x}_t^{(\ell)}). \qquad [5]$$

One can notice though that neither of these two operations do actually need an explicit expression for incoming messages. Producing $\hat{x}_t^{(\ell)}$ from $\beta_{ut}(x_t)$ and $\psi_{ut}(x_u, x_t)$ is straightforward. In turn, Expression [5] can be rewritten

$$w^{(\ell)} = \phi_t(\hat{x}_t^{(\ell)}) \prod_{i \in N(t) \setminus \{s,u\}} \int \psi_{ts}(x_i, \hat{x}_t^{(\ell)}) \beta_{it}(x_i) dx_i. \qquad [6]$$

Evaluating each integral is achieved by sampling p times an example $\hat{x}_i^{(k)}$ from either $\psi_{ts}(x_i, \hat{x}_t^{(\ell)})$ or $\beta_{it}(x_i)$, evaluating $\beta_{it}(\hat{x}_i^{(k)})$ or $\psi_{ts}(\hat{x}_i^{(k)}, \hat{x}_t^{(\ell)})$ respectively, and taking the average over k.

The computational complexity of importance weight computation with explicit messages [5] is $O(n)$, because of linear iteration through all messages and evidence which are of size n. The computational complexity with implicit messages [6] is $O(pn)$, because of p linear iterations through potentials or the local estimates. However, implicit messages effectively achieve the same resolution as explicit messages would *if* these explicit messages were supported by pn particles, *while keeping memory needs at* $O(n)$. Importance weight computation with implicit or explicit messages are thus expected to display processing times of the same order, while the implicit method will categorically require less memory.

4.3. *Practical considerations*

Ihler et al. draw a lot of attention towards kd-trees (Ihler *et al.*, 2003), which provide logarithmic access to elements enclosed in a given region of $\mathbb{R}^k$. We are less enthusiastic about them, for several reasons.

1) They are expensive to build.

2) The population size s for which kd-trees become faster than linear search grows as k grows. For $k = 3$, s is at least of the order of 1000. Moreover, our particle space is $SE(3)$, which includes the non-Euclidean subspace $SO(3)$, making for even more difficult search.

3) Computational details need to be kept in mind, too. For instance, n elements may very well fit in processor cache, but pn may continuously trigger cache misses. This of course is very dependant on the hardware, but it may motivate an implicit-message implementation, instead of explicit messages [5] of size pn organized in a kd-Tree.

4) Finally, kd-trees , and in particular the dual trees (Ihler *et al.*, 2003), represent a considerable implementation effort.

We intend to further investigate kd-trees in the near future, and expand the previous points with experimental benchmarks.

4.4. *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 (Ihler *et al.*, 2003). We overcome this problem with a two-level IS: we first compute an intermediate representation for the product with the procedure explained above, we then use this very representation as the proposal distribution for a second IS that will be geared towards relevant modes. The intermediate representation is obtained with sparse implicit messages ($p \ll n$) but many importance samples ($r \gg 1$), while the second IS uses rich implicit messages ($p \approx n$) but a low value for r. Denoting by $\beta^*_{ts}(x_t)$ the intermediate product representation, importance weights for the second IS are computed as

$$w^{(\ell)} = \frac{\phi_t(\hat{x}_t^{(\ell)}) \prod_{i \in N(t) \setminus s} m_{it}(\hat{x}_t^{(\ell)})}{\beta^*_{ts}(\hat{x}_t^{(\ell)})}.$$

In the equation above, messages are implicit.

The two-level IS described above and the high-resolution messages have been crucial elements of the successful application to real-world object presented at Section 6.

5. Pose Estimation

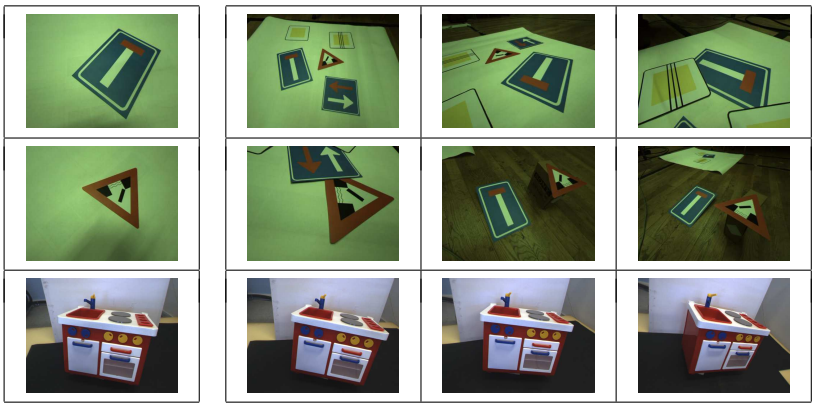
Features at the top of an object model represent the whole object, and they will present relatively concentrated densities that are unimodal if exactly one instance of this object is present in the scene. These densities can be used to estimate the object pose. Let us consider a model for a given object, and a pair of scenes where the object appears. In the first scene, the object is in a reference pose. In the second scene, the pose of the object is unknown. The application of our method to estimate the pose of the object in the second scene goes as follows:

1) Instantiate the object model in the reference scene. For every top-level feature i of the instantiated graph, compute a *reference aggregate feature pose* π_1^i from its unimodal density.

This step is necessary because even though the top-level features all represent the whole object, they come from different *unsupervised* recursive combinations of features of various poses (Detry *et al.*, 2007). Even though the *object* is in a reference pose, top feature poses $\{\pi_1^i\}$ are not expected to be located at $(0, 0, 0)$ or aligned with $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, and are not expected to be equal.

2) Instantiate the object model in the unknown scene. For every top feature of that graph, compute an *aggregate feature pose* π_2^i.

3) For all top level features i, the transformations from π_1^i to π_2^i should be very similar; let us denote the mean transformation by t. This transformation corresponds



(a) Learning (b) Evaluation

Figure 2. *Input imagery (only the left image in each stereo pair is presented). Effective resolution is* 1280×960 *pixels.*

to the rigid body motion between the pose of the object in the first scene and its pose in the second scene. Since the first scene is a reference pose, t is the pose of the object in the second scene.

A prominent aspect of this procedure is its ability to recover an object pose without explicit point-to-point correspondences. The estimated pose emerges from a negotiation involving all available data.

To compute the pose of an object that appears once in a cluttered scene, Step 2 of the above procedure mentions calculating an aggregate pose for the "unimodal" density of each top-level feature. While this unimodal hypothesis makes sense for very simple cases, it may very well be incorrect in more complex situations. For example, if an object very similar to the one we are searching for also appears in the scene, the top densities should present a second, weaker mode. A second mode could also appear when an object has some kind of symmetry, like the triangular traffic sign of Figure 2 that has similar appearances when rotated by 120° around its normal. Consequently, instead of simple aggregation at top level features, we cluster each density to extract its prominent mode.

6. Experiments

In this section, we demonstrate the applicability of our model with a series of pose estimation experiments in various cluttered scenes. We chose to learn models for the three objects presented at Figure 2(a). We then tried to estimate their poses in the scenes of Figure 2(b).

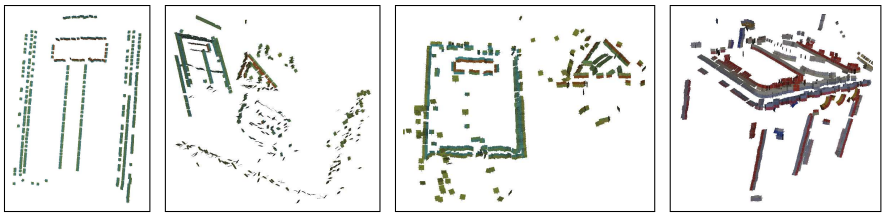


Figure 3. *Examples of ECV representations, extracted from scenes of Figure 2.*

Input 3D features for the bottom levels (primitive feature observations) are provided by an early-cognitive-vision (ECV) system (Krüger *et al.*, 2005), which extracts 3D primitives from stereo views of a scene. The quality of such ECV representations varies as a function of local visual signal quality. Figure 3 illustrates the ECV primitives for certain scenes of Figure 2.

Models for the three objects of Figure 2(a) were learned following the procedure mentioned above (Detry *et al.*, 2007). These models were learned from a clean view of each object (the reference scene), for example from the ECV representation in the first image of Figure 3. Each model has also been instantiated in its reference scene to compute a reference aggregate feature pose π_1^i for every top feature i.

The three models were all instantiated in the test scenes of Figure 2(b), using observations like these of Figure 3 as evidence. Evidence was propagated through the hierarchy, and we eventually got top-feature pose densities. Looking closer at the instantiation of one model in one scene, there are two cases to consider. First, the model had no instance in the scene. The top-level densities were relatively uniform, and the experiment was not needed to go any further. In the second case, an instance was present. Its was then always verified that each top feature i did present a principal mode π_2^i. We could thus compute the transformations t_i between π_1^i and π_2^i for every top feature i. As noted in Section 5, all t_i were very similar; we denote the mean transformation by t, which corresponds to the *estimated* rigid body motion between the pose of the object in the reference scene, and to its pose in the noisy scene.

We can evaluate the success of the experiment by transforming the reference scene with t, and superimposing it to the test scene; if the experiment is successful, the object of interest should overlap with its instance. Such evaluations are presented at Figure 4. All the experiments that we ran ended with successful pose recovery. For traffic signs, the worst estimate (Figure 4(d)) corresponds to the dead-end signal pose estimation in the sixth scene of Figure 2(b) (second row, third column). This is however one of the most difficult scenes: it has a brown background, thus changing the outside color of ECV primitives on the traffic sign contours. This induces wrong associations of observations to primitive features, and makes for harder inference. Estimation is still quite accurate given the difficulty of the scene. Other typical estimates are presented at Figure 4. In particular, 4(a) shows a good result despite occlusion.

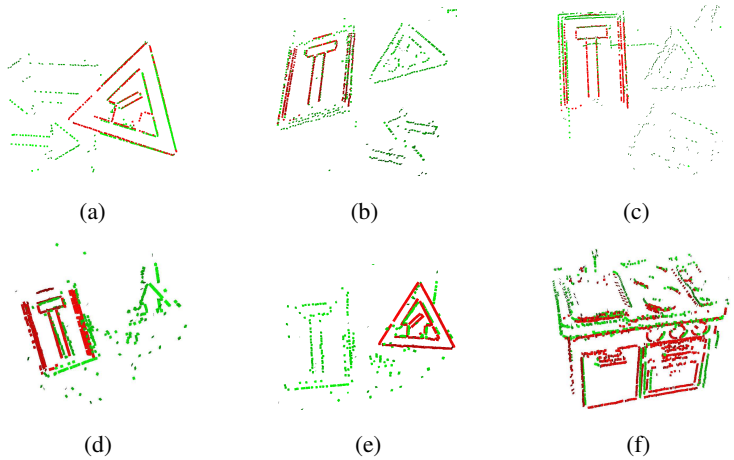


Figure 4. *Illustration of the pose estimation accuracy. Each picture shows in green a scene that contains one object of interest and in red the pose of that object inferred by our system.*

The accuracy of probabilistic pose estimation highly depends on the representation resolution. When an experiment lacks accuracy, retrying with more particles usually produces better results. Therefore, a meaningful quantitative evaluation must take into account the number of particles used. Figure 5 shows pose estimation error as a function of the number of particles per density. Because of the probabilistic nature of inference, runs with different software random seeds produces different results. Therefore, we run each experiment several times and study the mean error, plotted in red in the figure. The mean error decreases between 0 and 100 particles, and stabilizes for higher resolutions. We also plotted one standard deviation above the mean error, in dashed green. The error variance also decreases as the number of particles increases.

7. Conclusion

We presented an object representation framework that encodes probabilistic relations between 3D features. We discussed an Importance-Sampling–NBP inference process and outlined practical details. The inference process, together with the learning scheme of our previous work (Detry *et al.*, 2007), allow us to learn unsupervised part representations for real objects and instantiate them in cluttered scenes. 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 more perceptual modalities than vision. Our objective is to observe haptic and kinematic features that correlate

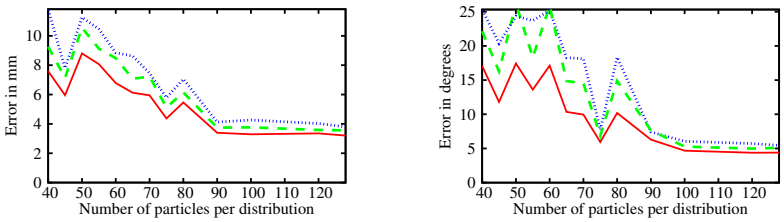


Figure 5. *Pose estimation accuracy as a function of the number of particles per density, for an instantiation of the opening-bridge traffic sign within the first scene of Figure 2(b). Left and right plots correspond to location and orientation error respectively. The red lines indicate the mean absolute error. The green and blue lines indicate the variance across runs and across top-level nodes. Location error can be compared to the traffic sign edge, which is 190mm long. See the text for details.*

with successful grasps, and integrate them into the feature hierarchy. Then, given a visual scene, grasp parameters can be suggested by probabilistic inference within the feature hierarchy.

Acknowledgment

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).

8. References

Detry R., Piater J. H., " Hierarchical Integration of Local 3D Features for Probabilistic Pose Recovery", *Robot Manipulation: Sensing and Adapting to the Real World (Workshop at Robotics, Science and Systems)*, 2007.

Ihler A. T., Sudderth E. B., Freeman W. T., Willsky A. S., " Efficient Multiscale Sampling from Products of Gaussian Mixtures.", *Neural Information Processing Systems*, 2003.

Jordan M. I., Weiss Y., " Graphical models: Probabilistic inference", *in* M. Arbib (ed.), *The Handbook of Brain Theory and Neural Networks, 2nd edition*, MIT Press, 2002.

Krüger N., Wörgötter F., " Multi-modal Primitives as Functional Models of Hyper-columns and Their Use for Contextual Integration.", *in* M. D. Gregorio, V. D. Maio, M. Frucci, C. Musio (eds), *BVAI*, vol. 3704 of *Lecture Notes in Computer Science*, Springer, p. 157-166, 2005.

Pearl J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.

Sudderth E. B., Ihler A. T., Freeman W. T., Willsky A. S., " Nonparametric Belief Propagation", *cvpr*, vol. 01, p. 605, 2003.

Yedidia J. S., Freeman W. T., Weiss Y., Understanding Belief Propagation and its Generalizations, Technical report, Mitsubishi Electric Research Laboratories, 2002.