

TCV Final Project Report

Nonparametric Belief Propagation for Limb Pose Estimation

Fernando Flores-Mangas
University of Toronto
mangas@cs.toronto.edu

May 12, 2006

1 Motivation

BELIEF PROPAGATION (BP) is a widely used algorithm for making inference from graphical models. The key is its ability to incorporate not only local evidence (either from a single node or from a small neighborhood) but from the whole graphical model in a very efficient way. However, BP was initially thought for low dimensional, parametrical distributions (mainly Gaussian) which not commonly arise in vision applications. Later on, a message passing algorithm, called Nonparametric Belief Propagation (NBP) was proposed as an alternative to accommodate such needs, with which virtually any distribution could be estimated. The power of such algorithm and the broadness of problems in which it could be successfully applied, motivated me to choose it as the central topic for this project.

After reviewing a few papers where the algorithm is either described [4] or applied [3, 1, 2], I picked and implemented a series of components to build a test environment, as an experimental way of achieving a better understanding of the strengths and weaknesses of the algorithm. The main task in this environment is pose estimation of deformable objects, formed with connected limbs, as the phalanges of a finger or the actual limbs in a human body. This problem perfectly aligns with the needs of belief propagation algorithm since each limb can be treated as a node in the graphical model and its relationship with neighboring limbs define the edges of the graph.

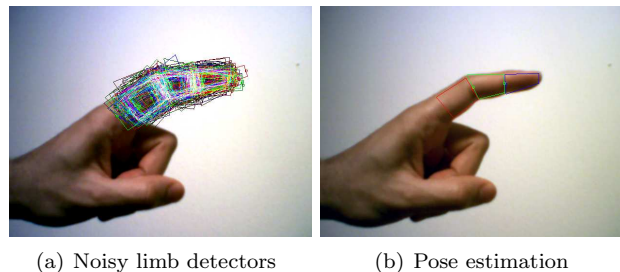


Figure 1: Finger Pose Estimation.

This report summarizes the results of such experimentation and analysis. Sections following this introduction are organized as follows: Section 2 presents the methodology used to implement, test and evaluate the performance of the NBP-based Pose Estimation algorithm. Section 3 shows the results of applying the algorithm, mainly to synthetically generated data, but a few examples from real data are shown as well. Finally, Section 4 presents some conclusions, such as minimum input requirements and also some failure modes, paired with what I interpret as the reasons, and some possible solutions.

2 Methodology

As mentioned earlier, in order to apply NBP to any (vision) problem, it is necessary to cast the problem as a graphical model. As done in [1], quadrangu-

lar shaped limbs are chosen to be the graph nodes and relationships between correspond to the edges. Each node is represented independently, for instance, a three limb model, such as a finger with a distal, central and proximal phalanges, would include three nodes and two (undirected) edges as shown in figure 2.

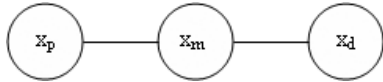


Figure 2: Three limb node model for a finger with three phalanges.

Each node in the graph is denoted as random variable X_i which corresponds to the pose parameters of limb i , e.g. X_m describes the pose of the middle phalange. On the other hand, edges represent constraints between adjacent limbs, and a potential function measuring the compatibility of neighboring limbs X_i and X_j can be described as $\psi_{ij}(X_i, X_j)$. The pose described by state X_i corresponds to a limb in the image, and the observation likelihood function can be denoted as $\phi_i(z_i|X_i)$. Finally, the whole object is a collection of all subparts, which in the finger example would be $X = \{X_p, X_m, X_d\}$.

2.1 Data Representation

Instead of representing a quadrangular limb with its set of four corners, features like shape, size, position and orientation parameters are used. To do so, the corners of a limb are first transformed to a new reference frame where the centroid is used as the origin, the y axis is parallel to the line that goes from the centroid to the center of the top two corners, and the x axis is perpendicular to y . Figure 3 shows an example.

Based on this new reference frame, scaling factors s_x and s_y are found for both x and y directions respectively. The value of s_x corresponds to the factor that scales the line formed with the top two corners into a fixed length l_x , (for example 40 pixels). The value of s_y is computed similarly but using the two

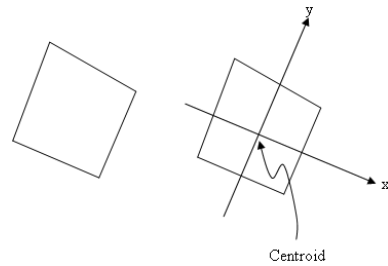


Figure 3: Reference frame.

leftmost corners. An example of this is shown in figure 4. This is called the normalization process.

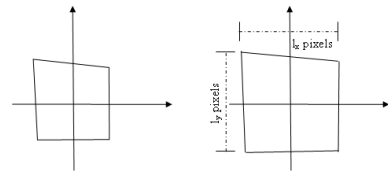


Figure 4: Normalization.

Once a quadrangular shape has been normalized (rotated towards a fixed reference frame and scaled to a fixed size) a Principal Component Analysis is run over the remaining corners (an 8 dimensional space). About one hundred different limbs descriptions were moused in and their eigenvalues and eigenvectors computed. I found that the three main components accounted for 98.36% of the variability of the data and therefore a three dimensional space was used for describing limbs' shape. Figure 5 shows a subset of the data on the top graph. The PCA eigenvalues are shown on the middle graph and the reconstruction on the bottom.

Figure 6 qualitatively shows what these three PCA components corresponds in real space. Each row shows the result of varying one of the components while keeping the other two unchanged. The first and last figures of the top and center row are the result of using extreme values (more than six standard deviations far from the mean), and are shown only to emphasize the deformation.

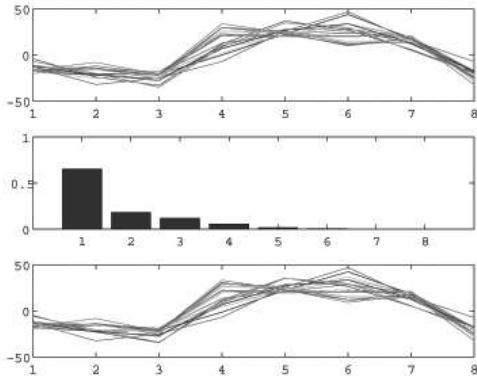


Figure 5: PCA Analysis. Top: data subset, Middle: eigenvalues, Bottom: reconstruction.

A full description of a node is then an eight dimensional variable accounting for the three PCA components, scaling in x and y , translation, (also in x and y) and orientation:

$$X_i = \{ps_1, ps_2, ps_3, s_x, s_y, p_x, p_y, \theta\} \quad (1)$$

2.2 Nonparametric Belief Propagation

The NBP algorithm is based on a message passing strategy, where each node i computes the message $m_{ij}(X_j)$ to be passed to a neighboring node j . Following intuition, the message considers three important factors as evidence towards a better definition of the posterior probability of a particular state X_j . These three factors are: local evidence: $\phi_i(z_i|X_i)$, compatibility of state X_j with state X_i of a neighboring limb: $\psi_{ji}(X_j, X_i)$, and a measurement of confidence provided by messages received at i from all its neighboring nodes: $\prod_{k \in \mathcal{N}(i) \setminus j} m_{ik}(X_k)$. The equation for computing the message is:

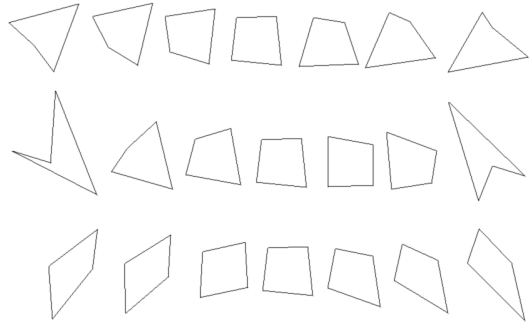


Figure 6: Sequences of limb reconstructions varying one PCA component at a time.

$$m_{ij}(X_j) = \int_{X_i} \phi_i(z_i|X_i) \psi_{ji}(X_j, X_i) \prod_{k \in \mathcal{N}(i) \setminus j} m_{ki}(X_k) dX_i \quad (2)$$

Not surprisingly, adequately selecting likelihood and compatibility functions is one of the key factors for the success (convergence to a meaningful posterior) of the NBP algorithm, as they provide with a big portion of the algorithm's available information, both bottom-up (image likelihood) and top down (model compatibility). Next sections describe what my choices are in this sense and the reasons for such preferences.

2.3 Image Likelihood Function

The image likelihood function $\phi_i(z_i|X_i)$ is a measurement of how well a state X_i fits the image information z_i . The likelihood function that I finally used is a modified version of what was proposed in [1].

Generally speaking, this new likelihood function is based on the steered edge response for all the points lying over the border lines of the state X_i in which an edge is expected (discarding shared edges where typically no edge can be seen in the image, i.e. between two neighboring phalanges). Then, the this edge response is thresholded so that all detected edges con-

tribute equally to the result. Finally, the thresholded response is accumulated and normalized by the length of the accounted lines. Further details are provided next.

A Canny edge detector was used for computing the steered edge response, the σ for the gaussian low-pass filter was chosen depending on the size of the smallest limb (with σ proportional to this size), usually $\sigma = [1, 2]$.

When using the Canny edge detector, the magnitude $|\Delta I|$ and orientation α of the gradient at any image point is computed with equations 3 and 4 respectively:

$$|\Delta I| = \sqrt{\Delta I_x^2 + \Delta I_y^2} \quad (3)$$

$$\alpha = \tan^{-1} \left(\frac{\Delta I_y}{\Delta I_x} \right) \quad (4)$$

Where ΔI_x and ΔI_y are the gradients of the gaussian low pass filtered image in both directions. However, to take into account the direction of the limb edge, the steered image response ϵ_θ is rather computed with equation 5:

$$\epsilon_\theta = |\Delta I| \sin(\alpha - \theta)^2 \quad (5)$$

where θ is the orientation of the limb edge line. Consequently, this modified term downweights the response of image edges which are parallel to the limb edge line.

The value of ϵ_θ is then thresholded:

$$\hat{\epsilon} = \begin{cases} 0 & \epsilon_\theta < \tau \\ 1 & \epsilon_\theta > \tau \end{cases} \quad (6)$$

Finally, the last step is to accumulate all local responses $\hat{\epsilon}_i$ and normalize the result by the length $|L|$ of the accounted lines L . This gives shape to the actual likelihood function:

$$\phi_i(z_i|X_i) = \frac{1}{|L|} \sum_{i \in L} \hat{\epsilon}_i \quad (7)$$

2.4 Compatibility Function

Experimentation with a large set of neighboring limbs compatibility functions led to using a simple, yet effective gaussian distribution. Initial stages of experimentation included exponential distributions as to reduce the distance between limbs as much as possible. However, this model does not aligns with what would be desirable, as there may be the case where a significant distance between neighboring limbs is actually a better estimate, for example between the base of the head and the top of the torso (leaving room for the neck).

On the other hand, a gaussian distribution allows for specifying a mean for the optimal distance and the variance can be chosen to describe joints' typical variability. Results of gaussian parameter variability are shown in Section 3.

2.5 Implementation

A complete test environment was built upon the description given in previous sections. However, the NBP algorithm requires bottom up cues which in real applications would be provided by an external component. This cues or shouters are instead simulated using well known distributions.

Certainly, this fact poses a disadvantage into the testing environment, as we wont know (unless real shouters are implemented) how the real performance would look like, however, using synthetically generated shouters also increases the broadness of the possible tests and allows for result comparison, even as to decide possible accuracy requirements for the low level detectors.

2.6 Synthetic Bottom-up Shouters

There are mainly two ways of drawing shouters for a limb in the implemented test environment: they can be either from an accurate distribution or from an extremely noisy one. The former is based on moused in descriptions of all the limbs of a complete object. For each each limb, all parameters (PCA components, scaling factors, position and orientation) are computed and used as the mean for the shouters cor-

responding to that specific limb. The program’s GUI allows the user to specify variances for each parameter. The later (noisy distribution) is defined as uniformly distributed positions all over the image, and gaussian distributions with large variances for the rest of the parameters (PCA components, scaling and orientation).

Finally an additional experimentation feature allows to draw shouters either from its own distribution, or from the distribution of any of the other limb, simulating the situation where a bottom up detector gets confused with similar limbs (i.e. middle phalanges with proximal phalanges).

2.7 Running NBP

Running NBP implies a series of steps, this section mentions the most important ones. First the number of “particles” needs to be specified. This number corresponds to the number N of shouters that will be used for each limb. As expected, the total number of shouters needed is N times the size (in limbs) of the object.

Then an full set shouters is drawn from the synthetic data distributions, (described in Section 2.6) with the selected distribution parameters. These shouters correspond to a variety of possible states of the search space, yet no information about its compatibility or image response is computed, only a constant value of 1.0 is used as initialization for the posterior. This initial set is stored as the “present” iteration set.

Steps 1 to 5 are then iterated for number of times T (until convergence is achieved). This number of iterations is specified in the GUI to allow for experimentation. Typical values are $T=[4,10]$.

1. Move “present” data set to “previous” data set.
2. Draw a new set of samples but keeping the top 10% of the existing samples, assuming that they are “accurate” states.
3. Compute the messages.
4. Compute individual posteriors for all states.

5. Sort states according to found posteriors for each limb.

Once the NBP iterations are done, there are two ways to make inference within the implementation, either by marginalization over the whole set, or using the MAP, which in this case would be the set states with the highest probability for each limb. Both approaches pose advantages and disadvantages which will be described in the results section.

3 Results

A very large set of test cases was run and its results analyzed both qualitatively and quantitatively. The GUI allows for easy modification of many important parameters for the algorithm and helps for better understanding.

Most of these tests were run over synthetically generated images although some real images were used as well. For the synthetic images, qualitative evaluation was made using the $F_1 - Measure$, which takes into account both sensitivity and precision using a harmonic mean. Qualitative evaluation was done by superimposing the resulting pose estimation over the input image. Results on real images are qualitatively evaluated only.

The first test set is a synthetic image simulating a three-limbed finger. A clean image was used first, 20 shouters are drawn for each limb and all correspond to the limb for which they were intended (as if bottom-up cues would perfectly differentiate between the tip of the finger, the middle and the base). I acknowledge this is not a realistic case but it is shown with the only purpose of setting a performance baseline. Figure 7 shows a set of superimposed samples over the test image on the top-left, the resulting estimated pose on the top right and the performance (using the $F_1 - Measure$) for 30 runs of the algorithm after 4 iterations is shown at the bottom.

Figure 8 shows the same kind of graphs but now limbs are drawn from any limb distribution (as if bottom up recognizers would not distinguish say, between the top of the finger and the base)

Clearly, the performance has dropped, but still the “finger” model information embedded in the NBP

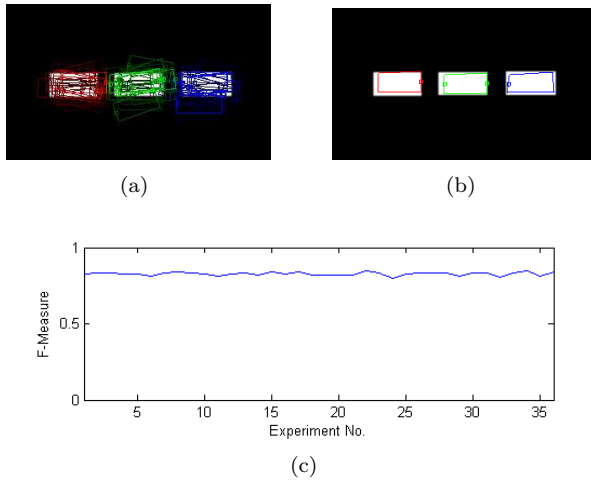


Figure 7: Synthetic finger, ideal case.

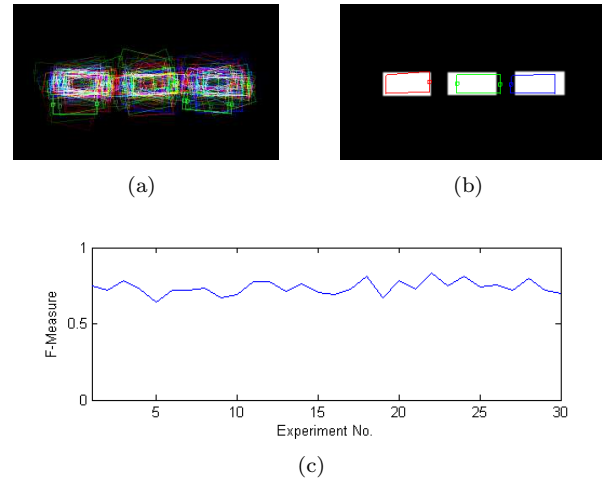


Figure 8: Synthetic finger, confused distributions.

compatibility function is enough to make an acceptable pose estimation. However, the number of shouters used in this experiment is substantially bigger ($N = 100$) in order to achieve this kind of results which would not be obtained with just 20 shouters per limb.

The last relevant test case with this image consists on a missing interior limb. Shouters for the interior missing limb were drawn from the noisy distribution described in Section 2.6. Same kind of graphs for this case are shown in Figure 9.

In an attempt to evaluate the performance of the algorithm in a graph that contains loops, two synthetic models with 4 and 5 limbs were used.

The 5 limb model was tested in a variety of ways, including the same used with the finger with very similar results. Figure 10 though, is shown because the performance in this test significantly decreased. The reason being that the best possible fit of the NBP algorithm depends on the accuracy of the shouters drawn from a noisy distribution. So, the performance decreases mainly because it is unlikely to have randomly placed shouters in the “right” pose (orientation, position, scale, etc).

The 4 limb model is a circular one, and it provides similar results with almost every test as the synthetic

finger did, except for one. When the shouters are selected from non-specific distributions, the posterior of the whole model has four peaks, corresponding to the four possible rotations of the model into the image. Therefore marginalizing over the samples for computing the posterior does not make any sense. In this case the MAP was used for displaying the result but still that was not enough, as every limb computes its own posterior and MAPs for different limb posteriors may coincide onto the same image limb. This happens often (not always) as shown in figure 11

Finally, the algorithm was tested in a few real images. The model of a finger was used and the pose of the finger in the image was estimated. Figure 1 shows a typical shouter set and pose estimation result. In this test shouters are not limb-specific.

This is certainly an ideal case with a white background and a clear edge finger. However I tried the algorithm in many other images, such as the “Bahen Sequence” or the (color images) of the CMU MoBo database. The results over the image where the shouters were placed was acceptable. However, when applied to the next image of the sequence, pose estimation was not accurate at all.

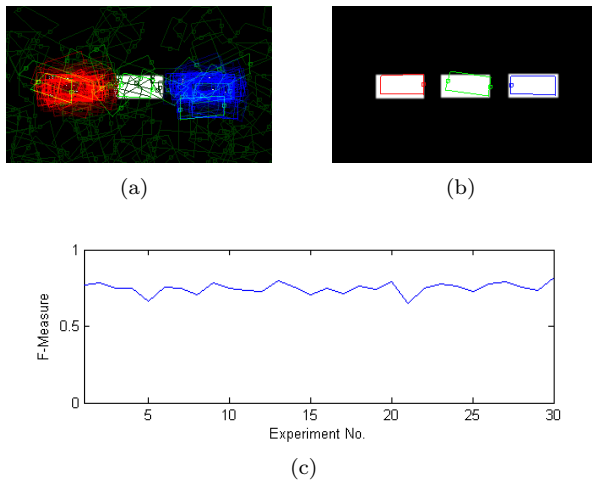


Figure 9: Synthetic finger, missing interior phalange.

4 Conclusions

In this project I analyzed, implemented, tested and evaluated a vision problem using NBP. The strength of the belief prop algorithm can be attributed to its ability to represent and combine both bottom-up (image cues) and top-down (object model) in a very natural way and therefore it shows to be applicable to a myriad of vision problems. However, the algorithm is certainly not the ultimate solution to any vision problem as a whole as it requires a big amount of information coming from other components (as limb detectors in the case of pose estimation) and its result is far from a final system response (interpretation of the posterior is needed). In the end, NBP is just a way of doing inference on a graphical model.

Regarding its application to pose estimation, some details about its components must be handled with care, as for example the image likelihood. It is unlikely that edge-based image likelihoods alone will reach acceptable results. A big amount of information is lost when using an edge detector and probably this kind of information is simply not enough. On the other hand, several features are consistently visible on limbs other than just edges, for instance color. If the model is initialized (or other priors are

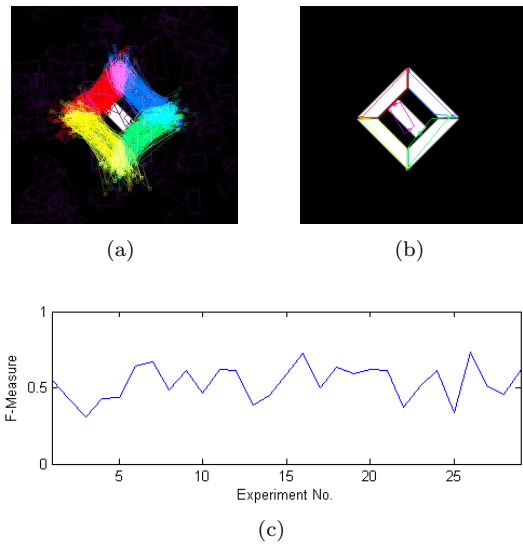


Figure 10: Synthetic 5-limbed body, missing interior limb.

known), a color histogram could be computed and used as an additional likelihood feature. Histograms of gradients may be used as well. The goal would be to increase the descriptiveness of the image likelihood with the least assumptions.

The same thing happens with the compatibility function. Along with distance, angles could be used as well, and furthermore, specific compatibility functions could be defined for each joint choosing only those features that better relate a pair of neighboring limbs. Also, virtual edges (edges not present in the original model) may link non-neighboring nodes (such as the face and the hands of a detailed human model) with further constraints (such as color in this face-hands example).

Finally, the performance of the whole algorithm is also extremely dependant on the accuracy and specificity of the shouters, but NBP makes an excellent job in prioritizing those which in fact satisfy the model and down-weights those who don't.

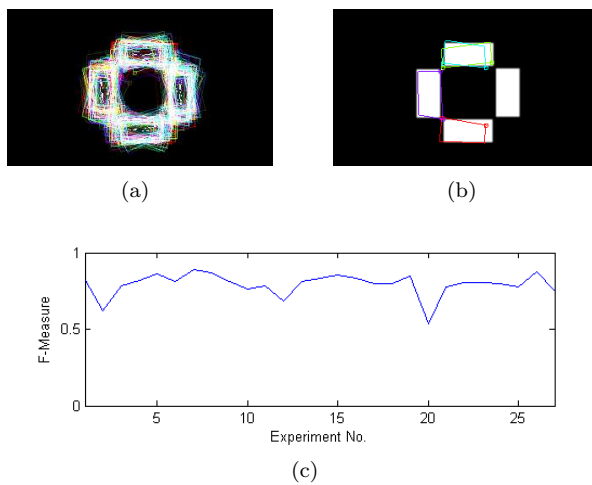


Figure 11: Synthetic 5-limbed body, missing interior limb.

belief propagation. In *CVPR (1)*, pages 605–612, 2003.

Acknowledges

I want to take this opportunity to thank both course professors David Fleet and Allan Jepson for the great effort of putting together this amazing course and for their experienced comments on this project.

References

- [1] Gang Hua, Ming-Hsuan Yang, and Ying Wu. Learning to estimate human pose with data driven belief propagation. In *CVPR*, volume 2, pages 747–754, 2005.
- [2] Leonid Sigal, Sidharth Bhatia, Stefan Roth, Michael J. Black, and Michael Isard. Tracking loose-limbed people. In *CVPR (1)*, pages 421–428, 2004.
- [3] Leonid Sigal, Michael Isard, Benjamin H. Sigelman, and Michael J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *NIPS*, 2003.
- [4] Erik B. Sudderth, Alexander T. Ihler, William T. Freeman, and Alan S. Willsky. Nonparametric