

DNBP: Differentiable Nonparametric Belief Propagation

ANTHONY OPIPARI, University of Michigan, USA

JANA PAVLASEK, University of Michigan, USA

CHAO CHEN, University of Michigan, USA

SHOUTIAN WANG, University of Michigan, USA

KARTHIK DESINGH, University of Washington, USA

ODEST CHADWICKE JENKINS, University of Michigan, USA

1 INTRODUCTION

Perceiving the pose of objects in space is a critical capability for robots operating in human environments. Perception and tracking of articulated objects, such as kitchen tools and human figures, is particularly challenging due to their high-dimensional and continuous state spaces where self occlusions are ubiquitous. Probabilistic graphical model inference is an approach with potential to compute articulated pose under these conditions of uncertainty. Nonparametric belief propagation (NBP) algorithms [26, 55] are a form of generative probabilistic inference that have proven effective for inference in visual perception tasks such as human pose tracking [53] and articulated object tracking in robotic perception [13, 45]. In addition to accounting for uncertainty in partially observable environments, these algorithms show promising computational properties in practice [13, 41, 44]. By relying on local message passing, NBP algorithms are ammenable for implementation on distributed, heterogeneous computing platforms.

The adaptability of NBP algorithms to new applications, however, is limited by the need to define hand-crafted functions that describe the distinct statistical relationships in a particular dataset. Current methods that utilize NBP rely on extensive domain knowledge to parameterize these relationships. Reducing the domain knowledge required by NBP methods would enable their use in a broader range of applications.

The capacity of NBP algorithms to perform inference using arbitrary graphs sets them apart from other generative inference algorithms such as the recursive Bayes filter [59] (e.g. particle filter [20]) and has been shown to be important in computational perception because it allows for modeling of non-causal relationships [55]. Neural network-based approaches are an alternative for computational perception [10, 34, 38, 61, 65]. These methods generally avoid the need for extensive domain knowledge by learning from large amounts of labelled data. Data-driven approaches, however, are prone to noisy estimates and have limited capacity to represent uncertainty inherent in their output. In robotic applications, both of these limitations negatively impact the ability for a robot to operate effectively in unstructured environments.

In this paper, we present a differentiable nonparametric belief propagation (DNBP) method, a hybrid approach which leverages neural networks to parameterize the NBP algorithm. Inspired by the differentiable particle filter from Jonschkowski et al. [27] and the pull message passing for nonparametric belief propagation (PMPNBP) algorithm [13], we develop a differentiable nonparametric belief propagation algorithm. DNBP performs end-to-end learning of each probabilistic factor required for graphical model inference.

The effectiveness of DNBP is demonstrated on two simulated articulated tracking tasks and on a real-world hand pose tracking task in challenging, noisy environments. An analysis of the learned probabilistic factors and resulting tracking performance is used to validate the approach. Results show that our approach can leverage the graph structure to report uncertainty about its estimates

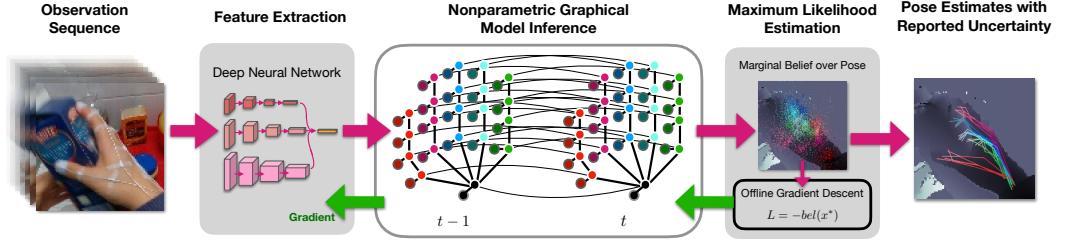


Fig. 1. Architecture diagram of differentiable nonparametric belief propagation. DNBP combines domain knowledge in the form of graphical models with differentiable neural networks for tractable inference in continuous spaces. Input features from a deep neural network and the probabilistic relationships encoded in a graphical model are learned jointly in an end-to-end fashion using backpropagation. Following offline training, DNBP can be applied to unseen data without hand-tuning.

while significantly reducing the need for prior domain knowledge required by previous NBP methods. DNBP performs competitively in comparison to traditional learning-based approaches on the tracking tasks. Collectively, these results indicate that DNBP has the potential to be successfully applied to robotic perception tasks, where maintaining a notion of uncertainty throughout the inference is beneficial.

1.1 Motivation and Societal Considerations

In this work, we seek methods that combine the robustness of generative probabilistic inference with the speed, recall power, and general adaptability of discriminative neural networks. Our aim is to find new *generative-discriminative* inference methods that can achieve the best of both worlds.

Our work on DNBP is motivated by a simple question: how can AI models be relied on in the face of uncertainty? As roboticists, we experience first-hand the uncertainty inherent in our interactions with the physical world, and the errors which result from it. It is not a matter of *if* mistakes occur during inference, but *when* mistakes will occur and *how* our systems can recover from such errors. There are profound questions as to *who* will be impacted when mistakes occur, and what costs will be imposed by these mistakes. *In other words, even if a neural network is accurate 99% of the time, how useful will it be if we do not know the 1% of time it is wrong?*

A notable example of the cost of mistakes in discriminative models is the 2020 wrongful arrest and detention of Mr. Robert Williams arising from a false positive facial recognition output by a neural network [64]. The potential for such harmful mistakes were foreseen in research from the algorithmic fairness community, such as work by Raji et al. [48] and Buolamwini and Gebru [8]. More broadly, the past decade has seen a transformative proliferation of discriminative neural networks [10, 34, 62] as well as extraordinary growth in the body of research focused on algorithmic fairness [15, 33].¹ In complement, the methods of inference must also improve if AI systems are to be useful for responsible decision making in an uncertain world. We posit that enabling AI models to maintain distributions over uncertain possibilities through the generation and evaluation of

¹In addition, the impact of artificial intelligence has also led to massive growth in the demand for computer science degrees [75] as the most lucrative pathway into the field of artificial intelligence. These growth trends have, unfortunately, not been seen for the participation of historically underrepresented minorities in professional pathways into computing and AI. This national trend is indicated by efforts such as the Michigan Computer Science and Engineering Climate, Diversity, Equity, and Inclusion Report, which reported 7.5% representation of students from underrepresented minority groups among its 2,586 declared undergraduate majors.

diagnosable hypotheses will empower human users seeking to diagnose when and why the models are unreliable.

An encouraging example of how generative inference can be reliable for robust AI comes from robotics and the foundational task of autonomous map building and localization [59]. Robots that use Bayes filtering for localization [12, 36] are able to overcome localization errors by recursively diagnosing which possible location for the robot best aligns with its observation of the environment. This generative approach to inference alternates between *hypothesising* possible states of the world and *evaluating* these hypotheses according to their agreement (or disagreement) with observation. Using this generative process enables robots to maintain probability mass over plausible states which are updated over time in response to perceived localization error. In contrast, a discriminative neural network approach to localization fuses together hypothesis and evaluation into a single opaque forward pass. The discriminative model implicitly reasons over the full state space. As such, any interpretation or diagnosis of the output of the neural network requires gathering meaning from the inner workings of the network, which remains an open area of research [5, 51]. Therefore, we underscore a critical feature of generative probabilistic inference is the *diagnosability* of mistakes in its estimates. In this work, we investigate a combined approach that explicitly generates hypothetical states and then weights them according to a learned neural network. With this approach, we aim to realize the best of both generative and discriminative approaches.

In addition to uncertainty in inference, we also know that decision making must often occur in situations dominated by partial observations. Our ability to reason in the physical world often relies on our beliefs about unseen objects just as much or perhaps more than it relies on what is directly in our field of view at any given moment. Reasoning in belief space offers one path to address such uncertainty by maintaining and reasoning over distributions, which allows for recovery from mistakes and recursive refinement as new observations are collected. However, purely probabilistic approaches to inference (arising from work by Kaelbling et al. [28]) have proven prohibitively slow computationally up to this point. New approaches to efficient generative probabilistic inference offer increasingly viable algorithms for object permanence [73] and belief space planning [2], although it remains unclear if these methods will be tractable for meaningful use. In this regard, we envision that robust robot systems will increasingly use generative-discriminative perception to dovetail with replanning algorithms [3, 19, 54, 72] for decision making. With this vision in mind, the current work sets out to address the challenge of modelling perceptual uncertainty using a generative-discriminative approach which could be used by a downstream belief-space replanning system. The replanning paradigm follows long-established practices used for recovery by mobile robots during autonomous navigation when localization errors occur [6, 12, 36]. We posit this approach to replanning will generalize across many scenarios where discriminative AI is now deployed, and ultimately lead to more accountable systems and responsible standards.

2 RELATED WORK

Belief Propagation: In the context of graphical models, inference refers to the process by which information about observed variables is used to derive the posterior distribution(s) of unobserved random variables. Belief propagation (BP) is a message passing algorithm for inferring the marginal distributions of graphical models [69]. BP computes exact marginal distributions on trees [46], and has demonstrated empirical success on graphs with cycles when applied in a loopy fashion [37, 39, 42, 57]. In order to apply inference techniques such as BP, the parameters of a graphical model (i.e. the graph structure and associated probabilistic factors) must be fully specified. The requirement that model parameters be specified limits BP’s adaptability to new applications. Thus, BP is a model-driven approach to inference with a high degree of introspection but limited adaptability (top left of Fig. 2). Furthermore, BP demands exact integral computations that constrain the algorithm’s

applicability to state spaces that are discrete. In contrast, this current study focuses on generative inference for continuous state spaces that robots are often faced with.

Nonparametric Belief Propagation: For continuous spaces, such as six degrees-of-freedom object pose, exact integrals called for in BP become intractable and approximate methods for inference have been considered. Nonparametric belief propagation (NBP) methods [26, 55], have been proposed which represent the inferred marginal distributions using mixtures of Gaussians and define efficient message passing approximations for inference. Isard [26] demonstrated the effectiveness of their proposed algorithm using a set of synthetic visual datasets each modeled with hand-crafted factors. Sudderth et al. [55] applied their NBP method successfully to a visual parts-based face localization task as well as a human hand tracking task [56]. In both applications, NBP relied on factor models which were chosen based on task-level domain knowledge (e.g. skin color statistics, valid hand configurations). Sigal et al. [53] extended these NBP methods to human pose estimation and tracking using factors which were each trained apart from the inference algorithm using independent training objectives.

Ihler and McAllester [25] described a conceptual theory of particle belief propagation, where messages being sent to inform the marginal of a particular variable could be generated using a shared proposal distribution. Following the work of Ihler and McAllester, Desingh et al. [13] presented an efficient “pull” message passing algorithm (PMPNBP) which uses a weighted particle set to

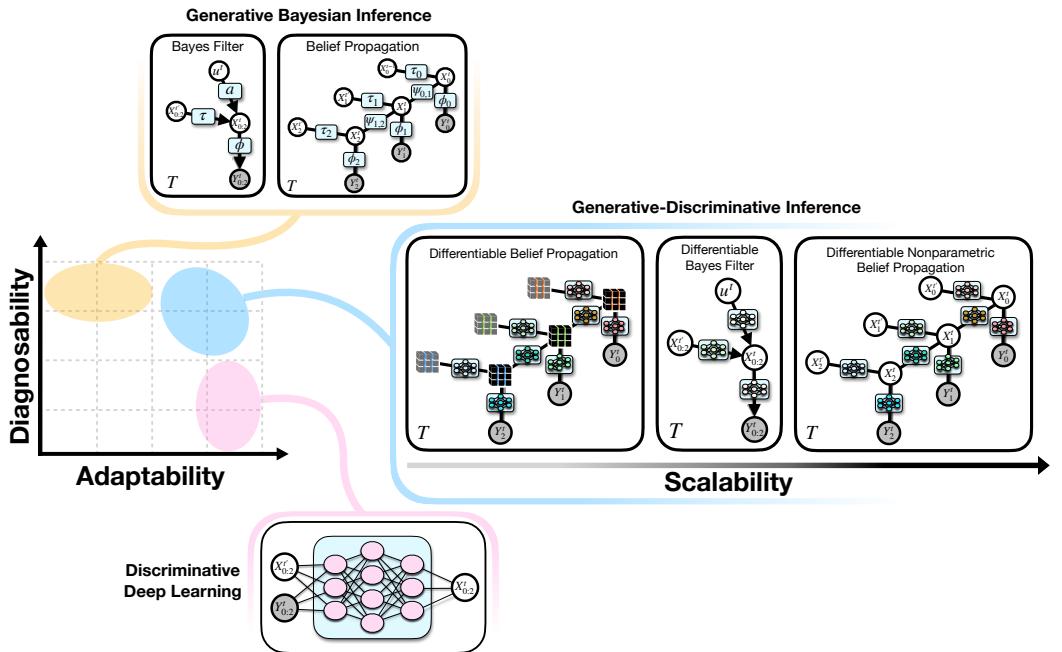


Fig. 2. A comparison of potential trade-offs among generative and discriminative inference approaches. Generative Bayesian inference (e.g. Bayes filtering [59] and belief propagation [46]) exhibits a high degree of diagnosability but low adaptability stemming from their reliance on brittle hand-crafting [50]. Discriminative deep learning [38] is a data-driven approach with high adaptability but can lack diagnosability in dynamic, partially-observable and uncertain environments. The aim of this work is to investigate a hybrid generative-discriminative inference approach, Differential Nonparametric Belief Propagation (DBNP), that can achieve the best of both model-driven and data-driven techniques; it should have a high degree of diagnosability and adaptability.

approximate messages between random variables. PMPNBP is effective on robot pose estimation tasks using hand-crafted factors. Using a similar approximation of belief propagation, Pavlasek et al. [45] took a step toward deep learning-based potential functions by introducing a pre-trained image segmentation network to the unary factors.

An important limitation of the existing NBP methods is they assume the probabilistic factors expressed in the graph are provided as input or rely on domain knowledge to separately model and train each function. However, the success of NBP methods in enabling inference that efficiently factors continuous state spaces motivates our work aimed at improving their adaptability.

Deep Learning: In recent years, neural network-driven deep learning has achieved state-of-the-art performance across a variety of perception tasks [22, 38, 49, 65]. The ability for deep learning models to reliably estimate their uncertainty has been identified as an important challenge for applying these techniques to robotic domains [58]. Bayesian deep learning approaches have been developed to address this challenge for domains where quantified uncertainty and the potential for introspection is expected [1]. Bayesian deep learning approaches for uncertainty quantification include Monte Carlo dropout [16, 47], variational inference [9, 31] and calibration [21]. The current study sets out to study a hybrid approach to enable uncertainty quantification by hybridizing deep neural networks with a nonparametric belief propagation algorithm.

Differentiable Belief Propagation: Deep learning architectures have been proposed that emulate the message passing operations of BP using tensor decompositions [14], invertible neural operators [35], convolutional neural networks [60], and graph neural networks [18, 71, 74]. These hybrid approaches were found to outperform non-hybrid models on a variety of inference datasets. However, they are either limited to discrete spaces [14, 18, 35, 60, 71] or provide only point estimates without a measure of uncertainty [74]. Xiong and Ruozzi [67] proposed a variational inference approach to approximate BP using learned neural network potentials and Gaussian quadrature. The variational approach demonstrated promising inference performance on discrete classification and a synthetic trivariate Gaussian mixture dataset. In contrast, the current work focuses on a NBP-deep learning hybrid for inference in continuous state spaces containing a larger number of degrees of freedom.

Differentiable Bayes Filtering: Variations of the Bayes filtering algorithm have been applied successfully to continuous state space inference tasks in robotics [12, 36, 59]. A new application of Bayesian filtering for robotics incorporates deep learning with end-to-end training. Haarnoja et al. [23] introduced a differentiable Kalman filter for mobile robot state estimation. Jonschkowski et al. [27] and Karkus et al. [29] both proposed differentiable particle filter algorithms for modeling continuous state spaces. Lee et al. [40] investigate how multimodal sensor information may be fused by deep learning and Bayesian filtering for rigid body pose estimation. Yi et al. [70] propose an end-to-end learning method for inference over factor-graph models in tracking and localization tasks. These studies all model a single object body using variants of the Bayes filter. In contrast, the current study focuses on modeling multi-part, articulated objects within the robotic context. The articulated object distinction motivates the use of NBP since its ability to factor high-dimensional continuous spaces is associated with improved performance in the face of increased dimensionality [13, 45].

3 BELIEF PROPAGATION

Consider a Markov Random Field (MRF) defined by the undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where \mathcal{V} denotes a set of nodes and \mathcal{E} denotes a set of edges. An example MRF model is shown in Fig. 3b. Each node in \mathcal{V} represents an observed (grey) or unobserved (white) random variable, while each edge in \mathcal{E} represents a pairwise relationship between two random variables in \mathcal{V} . The joint probability

distribution for \mathcal{G} is:

$$p(\mathcal{X}, \mathcal{Y}) = \frac{1}{Z} \prod_{(s,d) \in \mathcal{E}} \psi_{sd}(X_s, X_d) \prod_{d \in \mathcal{V}} \phi_d(X_d, Y_d) \quad (1)$$

where $\mathcal{X} = \{X_d \mid d \in \mathcal{V}\}$ is the set of unobserved variables and $\mathcal{Y} = \{Y_d \mid d \in \mathcal{V}\}$ is the set of corresponding observed variables. The scalar Z is a normalizing constant. For each node, the function $\phi_d(\cdot)$ is the *unary potential*, describing the compatibility of X_d with a corresponding observed variable Y_d . For each edge, the function $\psi_{sd}(\cdot)$ is the *pairwise potential*, describing the compatibility of neighboring variables X_s and X_d . This work considers MRF models limited to pairwise clique potentials.

Given the factorization of the joint distribution defined in Eq. (1), BP provides an algorithm for inference of the marginal posterior distributions, known as the beliefs, $bel_d(X_d)$. BP defines a message passing scheme for calculation of the beliefs as follows:

$$bel_d(X_d) \propto \phi_d(X_d, Y_d) \prod_{s \in \rho(d)} m_{s \rightarrow d}(X_d) \quad (2)$$

where $\rho(s)$ denotes the set of neighboring nodes of s . A message from node s to d is defined as:

$$m_{s \rightarrow d}(X_d) = \int_{X_s} \phi_s(X_s, Y_s) \psi_{sd}(X_s, X_d) \times \prod_{u \in \rho(s) \setminus d} m_{u \rightarrow s}(X_s) dX_s \quad (3)$$

Performing inference of random variables in continuous space causes the integral in Eq. (3) to become intractable. This motivates the use of efficient algorithms that approximate the message passing scheme of Eq. (2) and Eq. (3).

3.1 Nonparametric Belief Propagation

Nonparametric belief propagation (NBP) [55] uses Gaussian mixtures to represent the beliefs and messages for continuous random variables. Later works, including Ihler and McAllester [25] and Desingh et al. [13], further improve upon the tractability of approximate nonparametric inference by representing beliefs and messages with sets of weighted particles. These particle-based NBP methods infer an approximation of the beliefs using an iterative message passing algorithm, in which beliefs and messages are updated at each iteration t . In particular, Desingh et al. [13] avoid the expensive message generation of NBP by approximating Eq. (3) with a “pull” strategy. A message, $m_{s \rightarrow d}^t$, outgoing from s to d , is generated by first sampling M independent samples from $bel_d^{t-1}(X_d)$ then reweighting and resampling from this set.

4 DIFFERENTIABLE NONPARAMETRIC BELIEF PROPAGATION

We propose a differentiable nonparametric belief propagation (DNBP) method. DNBP maintains a representation of the uncertainty in the estimate by efficiently approximating the marginal posterior distributions encoded in an MRF. Our method avoids the need to define hand-crafted functions for each domain by modeling the potentials needed for the computation of the distributions with neural networks that are trained end-to-end. This hybrid generative-discriminative approach leverages the strengths of both NBP and neural networks.

DNBP uses an iterative, differentiable message passing scheme to infer the beliefs over hidden variables in an MRF. DNBP approximates the belief and messages in Eq. (2) and Eq. (3) at iteration

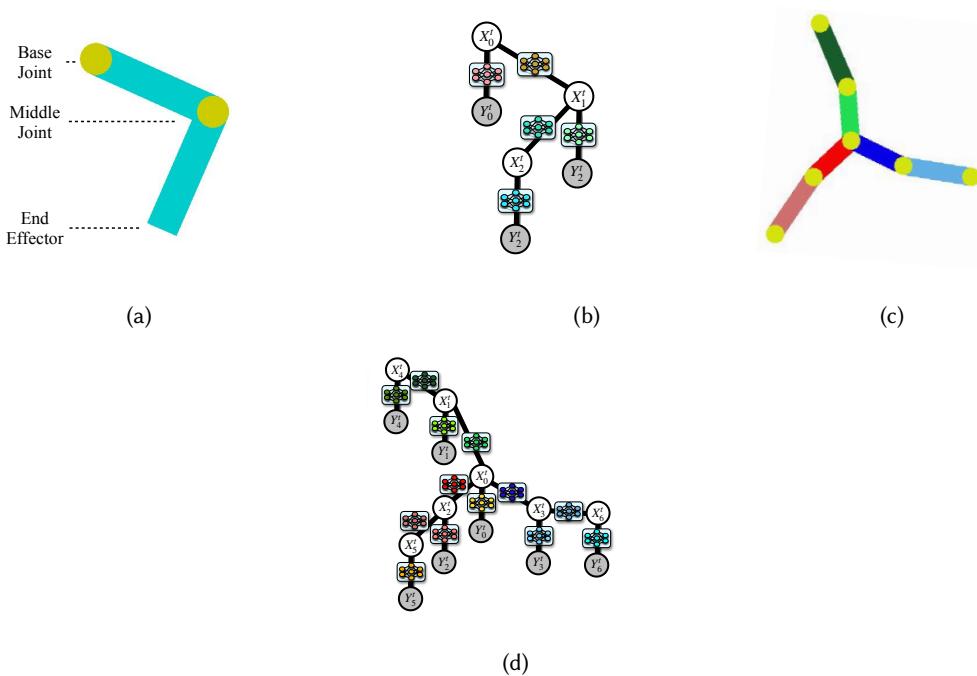


Fig. 3. a) Geometry and example configuration of the double pendulum. b) Graphical model used by DNBP for the double pendulum task. c) Geometry and an example configuration of the spider structure. d) Graphical model used by DNBP for the spider task.

t by sets of N and M weighted particles respectively:

$$bel_d^t(X_d) = \left\{ \left(\mu_d^{(i)}, w_d^{(i)} \right) \right\}_{i=1}^N \quad (4)$$

$$m_{s \rightarrow d}^t = \left\{ \left(\mu_{sd}^{(i)}, w_{sd}^{(i)} \right) \right\}_{i=1}^M \quad (5)$$

DNBP relies on a “pull” message passing strategy similar to the one presented by Desingh et al. [13]. In this strategy, each iteration of the algorithm is defined in terms of a message update step and a belief update step. The message update generates a new set of message particles as a reweighted set of samples from the previous iteration’s belief. Crucially, the weights associated with these updated message samples result from learned probabilistic factors as opposed to hand-crafted ones. Following a message update, the belief update combines information that is incoming to each node from the newly generated messages. Pseudocode of DNBP’s message and belief update schemes is included in Section 7.1. The following sections describe the networks used to compute the message and belief updates.

Unary Potential Functions: According to the factorization of the MRF joint distribution in Eq. (1), each unobserved variable X_d , for $d \in \mathcal{V}$, is related to a corresponding observed variable Y_d by the unary potential function $\phi_d(X_d, Y_d)$. DNBP models each unary function with a feedforward neural network. The unary potential for a particle, x_d , given an observed image, y_d , is:

$$\phi_d(X_d = x_d, Y_d = y_d) = l_d(x_d \oplus f_d(y_d)) \quad (6)$$

where f_d is a convolutional neural network, l_d is a fully connected neural network, and the symbol \oplus denotes concatenation of feature vectors. Details of network architectures are given in Section 7.2, Table 1.

Pairwise Potential Functions: For any pair of hidden variables, X_s and X_d , which are connected by an edge in \mathcal{E} , a pairwise potential function, $\psi_{sd}(X_s, X_d)$, represents the probabilistic relationship between the two variables. DNBP models each pairwise potential using a pair of feedforward, fully connected neural networks, $\psi_{sd}(X_s, X_d) = \{\psi_{sd}^\rho(\cdot), \psi_{sd}^\sim(\cdot)\}$. The pairwise *density* network, $\psi_{sd}^\rho(\cdot)$, evaluates the unnormalized potential for a pair of particles. The pairwise *sampling* network, $\psi_{sd}^\sim(\cdot)$, is used to form samples of node s conditioned on node d and vice versa. Details of network architectures are given in the Section 7.2, Table 1. The weight computation is detailed in the pseudocode in Section 7.1.

Particle Diffusion: DNBP uses a learned particle diffusion model for each hidden variable, modeled as distinct feedforward neural networks, $\tau_d^\sim(\cdot)$ for $d \in \mathcal{V}$. This diffusion model replaces the Gaussian diffusion models typically used by particle-based inference methods. At the outset of message generation at iteration t , DNBP’s belief particles from iteration $t - 1$ are resampled then passed through the diffusion model at the beginning of iteration t to form the messages used to update the distributions at iteration t .

Particle Resampling: The final operation of the belief update algorithm in NBP is a weighted resampling of belief particles. This resampling operation is non-differentiable [27, 29]. It follows that the iterative belief update algorithm is non-differentiable due to the resampling step. DNBP addresses the non-differentiability of the belief update algorithm by relocating the resampling and diffusion operations to the beginning of the message update algorithm. With this modification, the belief update returns a weighted set of particles approximating the marginal beliefs. The resulting belief density estimate is differentiable up to the beginning of the message update, when particles from the previous iteration were resampled. The resulting algorithm is differentiable through one belief update and message passing updates.

4.1 Supervised Training

DNBP’s training approach is inspired by the work of Jonschkowski et al. [27] with modifications to enable learning the potential functions distinct to DNBP. During training, DNBP uses a set of observation sequences, and a corresponding set of ground truth sequences. Using the observation sequences, DNBP estimates belief of each unobserved variable at each sequence step. Then, by maximizing estimated belief at the ground truth label of each unobserved variable, DNBP learns its network parameters by maximum likelihood estimation. Further details regarding the implementation of the training procedure are discussed in Section 7.2.

Objective Function: Given a set of weighted particles representing the belief of X_d produced by the inference procedure at iteration t , the density of the belief can be expressed as a mixture of Gaussians, with a component centered at each particle. The density of a sample x_d can be computed as follows:

$$\overline{bel}_d^t(x_d) = \sum_{i=1}^N w_d^{(i)} \cdot \mathcal{N}(x_d; \mu_d^{(i)}, \Sigma) \quad (7)$$

DNBP defines a loss function one each hidden node $d \in \mathcal{G}$ as:

$$L_d^t = -\log(\overline{bel}_d^t(x_d^{t,*})) \quad (8)$$

where $x_d^{t,*}$ denotes the ground truth label for node d at sequence step t . The loss for each hidden node is computed and optimized separately. At each sequence step during training, DNBP iterates

through the nodes of the graph, updating each node’s incoming messages and belief followed by a single optimization step of Eq. (8) using stochastic gradient descent.

5 RESULTS

The capability of DNBP is demonstrated on three challenging articulated tracking tasks. The first two tasks involve visually tracking the articulated joints of simulated articulated structures, as illustrated in Fig. 3. To increase the difficulty of these tasks, simulated clutter in the form of static and dynamic geometric shapes are rendered into the image sequences. In the second task, we evaluate DNBP on its ability to track the articulated pose of human hands. In both experiments, DNBP is directly compared to learned baseline approaches that are not NBP.

5.1 Datasets

Simulated Double Pendulum: To characterize DNBP’s tracking performance under chaotic motion, the double pendulum task was chosen as an initial evaluation. The double pendulum structure consists of two revolute joints connected to two rigid-body links in series (see Fig. 3a for illustration), which are acted on by gravity. The pose of the double pendulum is modeled by the 2-dimensional position of its two revolute joints, rendered as yellow circles, and one end effector. The training set on this task consists of 1024 total sequences with 20 frames per sequence while the validation set consists of 150 total sequences with 20 frames per sequence. Both training and validation sequences are split evenly among three bins of clutter ratio²: none, 0 to 0.04 and 0.04 to 0.1. Of the training and validation sequences with any amount of clutter, half contain static clutter and the other half contain dynamic clutter. The held-out test set is evenly split among clutter ratio deciles from 0 to 0.95, thus contains a shift in distribution from the training set, which was limited to clutter ratios below 0.1. Each decile contains 50 sequences with 100 frames per sequence. For test sequences with any amount of clutter, half contain static clutter and the other half contain dynamic clutter.

Simulated Articulated Spider: The spider task was chosen to further characterize DNBP’s performance using a structure with added articulations and a larger graphical model. As depicted in Fig. 3c, the spider is comprised of three revolute-prismatic joints, three purely revolute joints, and six rigid-body links. An example of the spider is shown in Fig. 3c, in which the joints are rendered as yellow circles and the rigid-body links are rendered as coloured rectangles. Unlike the double pendulum, which contained a stationary base joint, the spider is not tethered to any position and can move freely throughout the image under simulated joint control. The training, validation and test set for this task follow the same respective distributions of clutter as were used in the double pendulum datasets. The training set consists of 2,048 total sequences and the validation set consists of 300 sequences. The training and validation sequences are split evenly among five bins of clutter ratio: none, 0 to 0.04 and 0.04 to 0.1, 0.1 to 0.2 and 0.2 to 0.3. There are 20 frames per sequence in each of the spider datasets. Both simulated tasks use images of size 128 × 128 pixels. Ground truth keypoint locations are represented as continuous valued coordinates scaled to range of [−1, +1].

First-Person Hand Action Benchmark: The FPHAB dataset [17] consists of RGB-D image sequences taken from the first-person perspective. Thus, the dataset captures the pose and motion of human hands as they perform typical actions. This is a challenging dataset with extreme occlusions where complete observations of all the finger joints are rare. In total, there are 1175 distinct sequences and 105459 individual image frames. Each image is labeled with the 3D position of 21 hand joints (illustration of joint relations shown in center column of Fig. 1). The best-performing

²In this work, clutter ratio is defined as the ratio of pixels occluded by simulated clutter to the total number of image pixels and is averaged over a full sequence of images.

hand pose estimation baseline proposed by Garcia-Hernando et al. [17] is used for comparison in the current study. Just like Garcia-Hernando et al. [17], DNBP uses only depth observations. To ensure fair comparisons with the FPHAB baseline, this study follows the 1:1 cross-subject training protocol as described in FPHAB.

5.2 Implementation Details

On all three tasks, Adam [32] is used for network optimization with a batch size of 6 and models are trained until convergence of the validation loss. The graphs used by DNBP are shown in Figs. 1, 3b and 3d. While DNBP uses tree-structured graphs in these experiments, the inference strategy is compatible with graphs containing cycles since DNBP uses a loopy message passing scheme. DNBP is trained using 100 particles per message and tested using 200 particles per message. During training, one message update is performed at each sequence step, while two message updates are used at test time. The pairwise density, pairwise sampling and diffusion sampling processes of DNBP are defined over the relative translations between neighboring nodes. The maximum weighted particle from each marginal belief set of DNBP is used during evaluation for comparison with the ground truth.

On both simulated tasks, DNBP is compared to an LSTM recurrent neural network [24]. Both models use image inputs that are normalized channel-wise based on training set statistics. The total number of trainable parameters between LSTM and DNBP were chosen to be similar. For hand tracking, the preprocessing protocol of Xiong et al. [66], is followed. Notably, preprocessing on the hand tracking task assumes ground truth bounding boxes to ensure fair comparison with the baseline method published by Garcia-Hernando et al. [17]. Similarly, the feature extractor used by DNBP in the following experiments was designed to emulate the feature extractor of compared baseline. Details of network parameters and inspection of learned relationships are included in the Appendices 7.2 and 7.6.

5.3 Performance metrics

As a quantitative measure of tracking error, average Euclidean error is used. On the simulated tasks, Euclidean error is averaged over all images in the test set. On the hand tracking task, Euclidean error is averaged over all joints per frame then used to calculate the percent of frames satisfying variable error thresholds as used by Garcia-Hernando et al. [17].

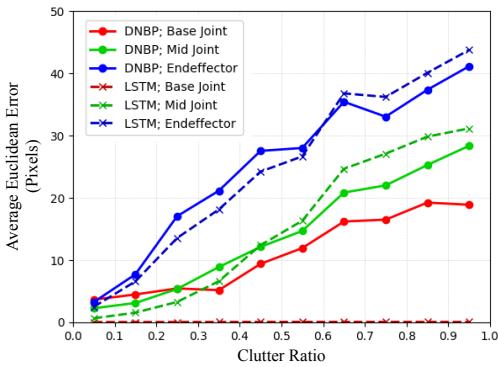


Fig. 4. Average error of DNBP and LSTM predictions as a function of clutter ratio and keypoint type for double pendulum tracking.

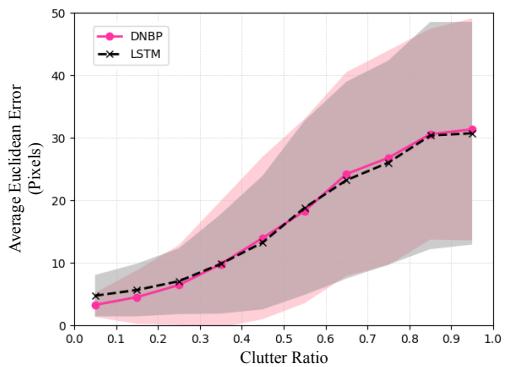


Fig. 5. Average error of DNBP and LSTM predictions as a function of clutter ratio for articulated ‘spider’ tracking.

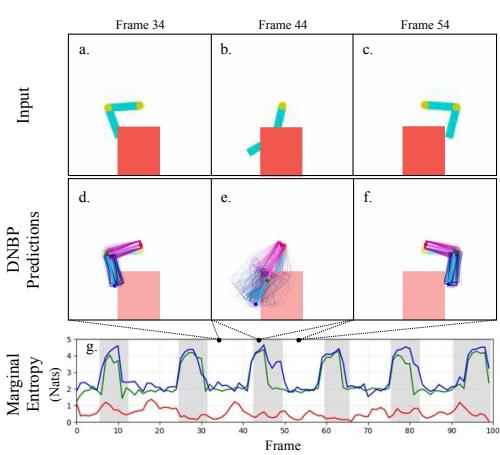


Fig. 6. Tracking of double pendulum by DNBP under partial occlusion (orange block). Uncertainty associated with predictions is shown as samples from the joint distribution in pink and blue (d,e,f). (g) Marginal entropy for each keypoint across test sequence; base keypoint (red), middle keypoint (green), end-effector keypoint (blue). Sequence steps highlighted by gray correspond to images in which $> 25\%$ of the pendulum is occluded.

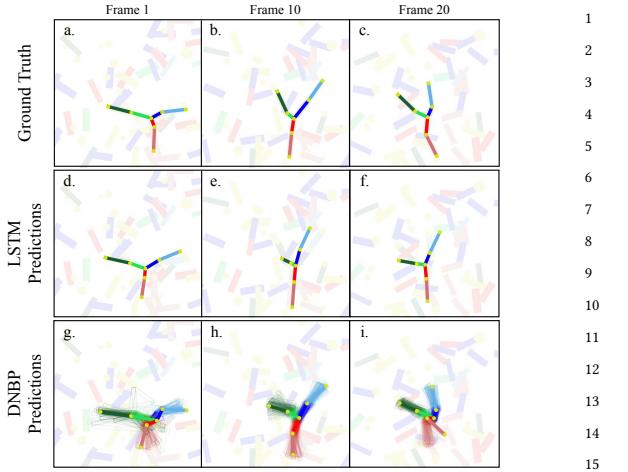


Fig. 7. Comparison of articulated ‘spider’ tracking by LSTM (d,e,f) and DNBP (g,h,i) under cluttered conditions. Predicted and ground truth keypoints shown as yellow circles. Clutter shown as faded shapes for illustration to highlight predictions.

Discrete entropy [52] is used as a quantitative measure of uncertainty estimated by DNBP. Discrete entropy is calculated by binning samples from each marginal belief set. For qualitative analysis of the uncertainty estimated by DNBP, samples from an approximation of the joint posterior distribution (i.e. for collection of all unobserved variables) are formed using a sequential Monte Carlo sampling approach [43]. Visualization of these samples are formed by plotting a rendered link between each pair of keypoint samples.

5.4 Double Pendulum Tracking Results

As shown in Fig. 4, the keypoint tracking error of DNBP is directly compared to that of the LSTM baseline on the held-out test set for each keypoint type (base, middle and end effector) across the full range of clutter ratios. Results from this comparison show that DNBP’s average keypoint tracking error is comparable to the LSTM’s corresponding error for both the mid joint and end effector keypoints, independent of clutter ratio. For the base joint keypoint, which is stationary at the center position of every image, the LSTM was able to memorize the correct position. DNBP, which diffuses particles based on the message passing scheme, does not memorize the base joint position and registers a consistently larger error which increased with clutter ratio.

DNBP provides measures of uncertainty associated with its predictions, which are generated according to the algorithmic prior of belief propagation. Next tested was the hypothesis that the DNBP model would generate increased uncertainty under conditions in which an occluding object is placed into the input images such that it covers portions of the double pendulum. This test was performed by rendering an occluding block onto a test sequence as shown in Fig. 6a-c. Under optimal conditions, in which the pendulum is minimally occluded ($< 25\%$ by surface area), the

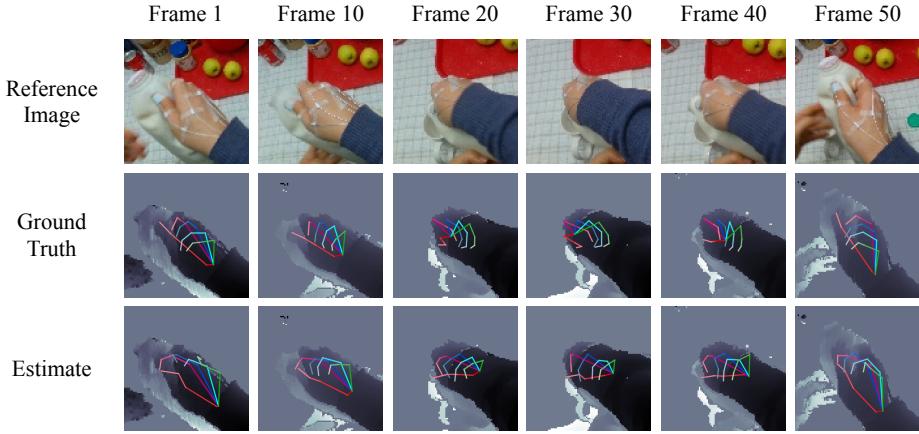


Fig. 8. Output from DNBP throughout a chosen sequence of hand tracking. DNBP maintains plausible estimates of the hand pose in cases of occlusion (Frames 20, 30, 40) and recovers with improved observability (Frame 50).

model’s output indicates a low level of uncertainty (see Fig. 6d,f,g.) for each keypoint and each frame. In contrast, under conditions in which the pendulum is occluded by the superimposed object, the model’s output indicates relatively high levels of uncertainty precisely at frames in which the superimposed object occludes a portion ($> 25\%$) of the double pendulum (see Fig. 6e,g.). These results demonstrate that the estimate of uncertainty produced by DNBP can identify predictions which are unreliable.

5.5 Articulated Spider Tracking Results

After having established the performance characteristics of DNBP on the relatively straightforward double pendulum task, we next set out to determine DNBP’s capability for tracking more complex structures. To this end, the 3-arm spider structure is used as a more challenging articulated pose tracking task. Each model’s performance is quantitatively assessed on the held-out test set of the articulated spider tracking task using the same approach as described for the double pendulum experiment by varying clutter ratio (Fig. 5). Similar to the results of the double pendulum experiment, average error on the spider task increases as a function of clutter ratio for both the LSTM and for DNBP. For clutter ratios between 0 and 0.25, average error for both models remains near 6 pixels then increases consistently with clutter ratio, reaching above 30 pixels of average error for clutter ratios above 0.85. As in the case of the double pendulum experiment, these results demonstrate comparable performance between LSTM and DNBP on an articulated pose tracking task.

Next, a qualitative example of tracking performance under conditions of clutter is shown in Fig. 7. In Fig. 7(a-c), the ground truth pose is shown amidst distracting shapes across selected frames of a test sequence with clutter ratio of 0.25. Pose predictions generated by LSTM are shown in Fig. 7(d-f) and by DNBP in (g-i). Qualitative assessment of the images indicates both the LSTM and DNBP place their predictions in the correct region of the image. Additionally, each model is shown to correctly predict the relative positions of the three arms. Over the sequence, both models track the motion of each keypoint, however appear to struggle with certain keypoint predictions.

5.6 Human Hand Tracking Results

To evaluate DNBP’s capability for application to real-world tasks, the algorithm’s state estimation and tracking performance was evaluated on the FPHAB dataset. This is a challenging dataset with

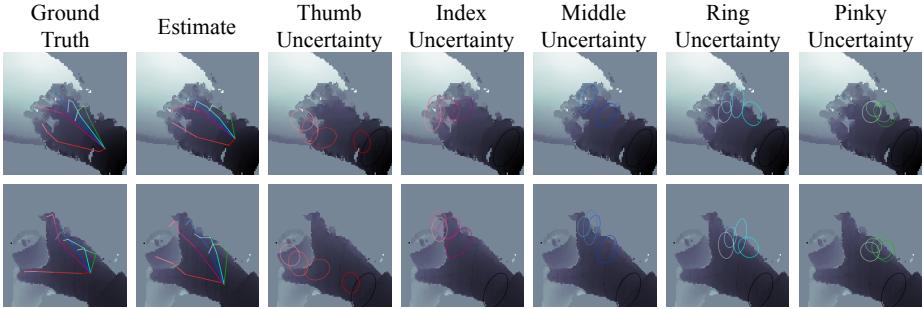


Fig. 9. Output from DNBP on randomly sampled frames. See Section 7.7 for more examples.

extreme occlusions where complete observations of all the finger joints are rare. Firstly, Euclidean error between the estimated and ground truth pose is measured for every frame in the test set. For this first evaluation, DNBP is applied as a frame-by-frame estimator without maintaining its belief over time. The quantitative results from this experiment, are included in Fig. 10 with direct comparison to a pure neural network baseline. The results from this experiment indicate that for error thresholds below 50mm, DNBP will consistently have an accuracy of 95% and above.

Following the comparison against a state of the art baseline, it was hypothesized that DNBP's performance would improve when applied as a tracking method which maintains belief over time. To perform this test, DNBP was applied sequentially to each test sequence and evaluated under the same error metric. The result from this test, as shown in Fig. 10, demonstrates that DNBP does improve in terms of frame error when allowed to track its uncertainty over time. Qualitative examples (on frames from randomly chosen sequences) showing DNBP's tracking performance are shown in Figs. 8 and 9 and Appendix 7.7. The tracking videos showing the

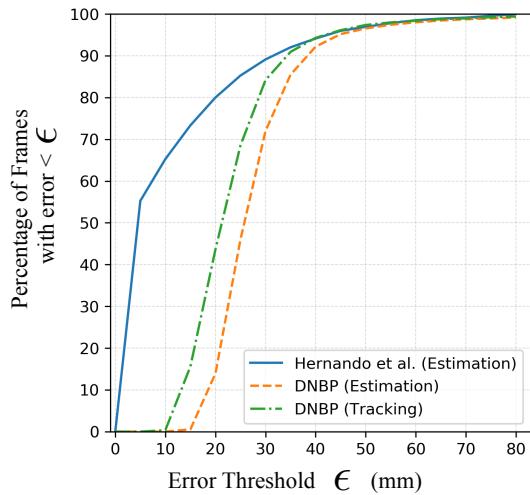


Fig. 10. Quantitative comparison between DNBP and neural network baseline on hand pose tracking task of the FPHAB dataset. For each model the percent of frames with predicted pose less than a set threshold is calculated as the threshold is varied from 0mm to 80mm.

DNBP's estimates and belief are included in the supplementary material and project webpage: <https://progress.eecs.umich.edu/projects/dnbp/>.

6 CONCLUSION

In this work, we proposed a novel formulation of belief propagation which is differentiable and uses a nonparametric representation of belief. It was hypothesized that combining maximum likelihood estimation with the nonparametric inference approach would enable end-to-end learning of the probabilistic factors needed for inference. Results on both qualitative and quantitative experiments demonstrate successful application of this approach and highlight the capability of DNBP to estimate useful measures of uncertainty, which are crucial for applications where incorrect estimates lead to catastrophic decisions, such as robotics. The current approach is limited by its use of non-differentiable resampling and its demand for a graph model as input. Exploration of methods to overcome these limitations, such as by incorporating a soft-resampling strategy [29], are left as future work. Inspired by recent work that has extended differentiable state estimation algorithms into the planning domain [4, 30, 63], we see the potential to embed DNBP within a differentiable planning system as an exciting direction for future work.

REFERENCES

- [1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khorsavi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion* 76 (2021), 243–297. <https://doi.org/10.1016/j.inffus.2021.05.008>
- [2] Alphonsus Adu-Bredu, Nikhil Devraj, Pin-Han Lin, Zhen Zeng, and Odest Chadwicke Jenkins. 2021. Probabilistic Inference in Planning for Partially Observable Long Horizon Problems. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 3154–3161. <https://doi.org/10.1109/IROS51168.2021.9636685>
- [3] Alphonsus Adu-Bredu, Zhen Zeng, Neha Pusalkar, and Odest Chadwicke Jenkins. 2022. Elephants Don't Pack Groceries: Robot Task Planning for Low Entropy Belief States. *IEEE Robotics and Automation Letters* 7, 1 (2022), 25–32. <https://doi.org/10.1109/LRA.2021.3116327>
- [4] Peter Anderson, Ayush Shrivastava, Devi Parikh, Dhruv Batra, and Stefan Lee. 2019. Chasing Ghosts: Instruction Following as Bayesian State Tracking. In *Advances in Neural Information Processing Systems*. 369–379.
- [5] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115. <https://doi.org/10.1016/j.inffus.2019.12.012>
- [6] Joydeep Biswas and Manuela M. Veloso. 2013. Localization and navigation of the CoBots over long-term deployments. *The International Journal of Robotics Research* 32, 14 (2013), 1679–1694. <https://doi.org/10.1177/0278364913503892>
- [7] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540
- [8] Joy Buolamwini and Timnit Gebru. 2018. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency (Proceedings of Machine Learning Research, Vol. 81)*, Sorelle A. Friedler and Christo Wilson (Eds.). PMLR, 77–91. <https://proceedings.mlr.press/v81/buolamwini18a.html>
- [9] Razvan Caramalau, Binod Bhattachari, and Tae-Kyun Kim. 2021. Active Learning for Bayesian 3D Hand Pose Estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 3419–3428.
- [10] Dan Cireşan, Ueli Meier, and Juergen Schmidhuber. 2012. Multi-column Deep Neural Networks for Image Classification. arXiv:1202.2745 [cs.CV]
- [11] Alex Clark. 2015. Pillow (PIL fork) documentation.
- [12] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. 1999. Monte carlo localization for mobile robots. In *International Conference on Robotics and Automation (ICRA)*, Vol. 2. IEEE, 1322–1328.
- [13] Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. 2019. Efficient nonparametric belief propagation for pose estimation and manipulation of articulated objects. *Science Robotics* 4, 30 (2019). <https://doi.org/10.1126/scirobotics.aaw4523>
- [14] Mohammed Haroon Dupty and Wee Sun Lee. 2020. Neuralizing Efficient Higher-order Belief Propagation. *CoRR* abs/2010.09283 (2020). arXiv:2010.09283 <https://arxiv.org/abs/2010.09283>

- [15] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through Awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (Cambridge, Massachusetts) (*ITCS '12*). Association for Computing Machinery, New York, NY, USA, 214–226. <https://doi.org/10.1145/2090236.2090255>
- [16] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 48)*, Maria Florina Balcan and Kilian Q. Weinberger (Eds.). PMLR, New York, New York, USA, 1050–1059. <https://proceedings.mlr.press/v48/gal16.html>
- [17] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. 2018. First-Person Hand Action Benchmark with RGB-D Videos and 3D Hand Pose Annotations. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [18] Víctor García-Satorras and Max Welling. 2021. Neural Enhanced Belief Propagation on Factor Graphs. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 130)*, Arindam Banerjee and Kenji Fukumizu (Eds.). PMLR, 685–693. <https://proceedings.mlr.press/v130/garciasatorras21a.html>
- [19] Caelan Reed Garrett, Chris Paxton, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Dieter Fox. 2020. Online Replanning in Belief Space for Partially Observable Task and Motion Problems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 5678–5684. <https://doi.org/10.1109/ICRA40945.2020.9196681>
- [20] S. Godsill. 2019. Particle Filtering: the First 25 Years and beyond. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7760–7764. <https://doi.org/10.1109/ICASSP.2019.8683411>
- [21] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1321–1330. <https://proceedings.mlr.press/v70/guo17a.html>
- [22] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. 2018. DensePose: Dense Human Pose Estimation in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [23] Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. 2016. Backprop KF: Learning Discriminative Deterministic State Estimators. In *Advances in Neural Information Processing Systems*. 4376–4384.
- [24] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (1997), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- [25] Alexander Ihler and David McAllester. 2009. Particle belief propagation. In *Artificial Intelligence and Statistics*. 256–263.
- [26] Michael Isard. 2003. PAMPAS: Real-valued graphical models for computer vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society.
- [27] Rico Jonschkowski, Divyam Rastogi, and Oliver Brock. 2018. Differentiable Particle Filters: End-to-End Learning with Algorithmic Priors. In *Robotics: Science and Systems (RSS)*. <https://doi.org/10.15607/RSS.2018.XIV.001>
- [28] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 1 (1998), 99–134. [https://doi.org/10.1016/S0004-3702\(98\)00023-X](https://doi.org/10.1016/S0004-3702(98)00023-X)
- [29] Péter Karkus, David Hsu, and Wee Sun Lee. 2018. Particle Filter Networks with Application to Visual Localization. In *Conference on Robot Learning (CoRL)*, Vol. 87. PMLR, 169–178.
- [30] Péter Karkus, Xiao Ma, David Hsu, Leslie Pack Kaelbling, Wee Sun Lee, and Tomás Lozano-Pérez. 2019. Differentiable Algorithm Networks for Composable Robot Learning. In *Robotics: Science and Systems*. <https://doi.org/10.15607/RSS.2019.XV.039>
- [31] Diederik Pieter Kingma. 2017. Variational inference & deep learning: A new synthesis. (2017).
- [32] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).
- [33] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2017. Inherent Trade-Offs in the Fair Determination of Risk Scores. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA (LIPIcs, Vol. 67)*, Christos H. Papadimitriou (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 43:1–43:23. <https://doi.org/10.4230/LIPIcs.ITCS.2017.43>
- [34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 25. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>
- [35] Jonathan Kuck, Shuvam Chakraborty, Hao Tang, Rachel Luo, Jiaming Song, Ashish Sabharwal, and Stefano Ermon. 2020. Belief Propagation Neural Networks. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 667–678. <https://proceedings.neurips.cc/paper/2020/file/07217414eb3fbe24d4e5b6caf91ca18-Paper.pdf>

- [36] Cody Kwok, Dieter Fox, and Marina Meila. 2003. Real-Time Particle Filters. In *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer (Eds.), Vol. 15. MIT Press. <https://proceedings.neurips.cc/paper/2002/file/2d2ca7eedf739ef4c3800713ec482e1a-Paper.pdf>
- [37] Xiangyang Lan, Stefan Roth, Daniel P. Huttenlocher, and Michael J. Black. 2006. Efficient Belief Propagation with Learned Higher-Order Markov Random Fields. In *European Conference on Computer Vision (ECCV) (Lecture Notes in Computer Science, Vol. 3952)*. Springer, 269–282. https://doi.org/10.1007/11744047_21
- [38] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.
- [39] Kuang-chih Lee, Dragomir Anguelov, Baris Sumengen, and Salih Burak Göktürk. 2008. Markov random field models for hair and face segmentation. In *International Conference on Automatic Face and Gesture Recognition (FG 2008)*. IEEE Computer Society, 1–6. <https://doi.org/10.1109/AFGR.2008.4813431>
- [40] Michelle A. Lee, Brent Yi, Roberto Martín-Martín, Silvio Savarese, and Jeannette Bohg. 2020. Multimodal Sensor Fusion with Differentiable Filters. *CoRR* abs/2010.13021 (2020). arXiv:2010.13021
- [41] Yanqi Liu, Anthony Oipipari, Odest Chadwicke Jenkins, and R. Iris Bahar. 2022. A Reconfigurable Hardware Library for Robot Scene Perception. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design (San Diego, California) (ICCAD '22)*. Association for Computing Machinery, New York, NY, USA, Article 101, 9 pages. <https://doi.org/10.1145/3508352.3561110>
- [42] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *Conference on Uncertainty in Artificial Intelligence (UAI)*. Morgan Kaufmann, 467–475.
- [43] Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. 2014. Sequential Monte Carlo for Graphical Models. In *Advances in Neural Information Processing Systems*. 1862–1870.
- [44] Joseph Ortiz, Talfan Evans, and Andrew J. Davison. 2021. A visual introduction to Gaussian Belief Propagation. *arXiv preprint arXiv:2107.02308* (2021).
- [45] Jana Pavlasek, Stanley Lewis, Karthik Desingh, and Odest Chadwicke Jenkins. 2020. Parts-Based Articulated Object Localization in Clutter Using Belief Propagation. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE.
- [46] Judea Pearl. 1988. Chapter 4 - BELIEF UPDATING BY NETWORK PROPAGATION. In *Probabilistic Reasoning in Intelligent Systems*, Judea Pearl (Ed.). Morgan Kaufmann, San Francisco (CA), 143 – 237. <https://doi.org/10.1016/B978-0-08-051489-5.50010-2>
- [47] Lorena Qendro, Sangwon Ha, René de Jong, and Partha Maji. 2021. Stochastic-Shield: A Probabilistic Approach Towards Training-Free Adversarial Defense in Quantized CNNs. In *Proceedings of the 1st Workshop on Security and Privacy for Mobile AI (Virtual, WI, USA) (MAISP'21)*. Association for Computing Machinery, New York, NY, USA, 1–6. <https://doi.org/10.1145/3469261.3469404>
- [48] Inioluwa Deborah Raji, Timnit Gebru, Margaret Mitchell, Joy Buolamwini, Joonseok Lee, and Emily Denton. 2020. *Saving Face: Investigating the Ethical Concerns of Facial Recognition Auditing*. Association for Computing Machinery, New York, NY, USA, 145–151. <https://doi.org/10.1145/3375627.3375820>
- [49] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.
- [50] Nicholas Roy, Ingmar Posner, Tim Barfoot, Philippe Beaudoin, Yoshua Bengio, Jeannette Bohg, Oliver Brock, Isabelle Depatie, Dieter Fox, Dan Koditschek, et al. 2021. From Machine Learning to Robotics: Challenges and Opportunities for Embodied Intelligence. *arXiv preprint arXiv:2110.15245* (2021).
- [51] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- [52] Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.* 27, 3 (1948), 379–423. <https://doi.org/10.1002/j.1538-7305.1948.tb01338.x>
- [53] Leonid Sigal, Sidharth Bhatia, Stefan Roth, Michael J. Black, and Michael Isard. 2004. Tracking Loose-Limbed People. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 421–428. <https://doi.org/10.1109/CVPR.2004.252>
- [54] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. 2013. DESPOT: Online POMDP Planning with Regularization. In *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2013/file/c2aee86157b4a40b78132f1e71a9e6f1-Paper.pdf>
- [55] Erik B. Sudderth, Alexander T. Ihler, William T. Freeman, and Alan S. Willsky. 2003. Nonparametric Belief Propagation. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 605–612. <https://doi.org/10.1109/CVPR.2003.1211409>
- [56] Erik B Sudderth, Michael I Mandel, William T Freeman, and Alan S Willsky. 2004. Visual hand tracking using nonparametric belief propagation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04)*.

189–189.

- [57] Jian Sun, Nanning Zheng, and Heung-Yeung Shum. 2003. Stereo Matching Using Belief Propagation. *IEEE Trans. Pattern Anal. Mach. Intell.* 25, 7 (2003), 787–800. <https://doi.org/10.1109/TPAMI.2003.1206509>
- [58] Niko Sünderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. 2018. The limits and potentials of deep learning for robotics. *The International Journal of Robotics Research* 37, 4-5 (2018), 405–420. <https://doi.org/10.1177/0278364918770733> arXiv:<https://doi.org/10.1177/0278364918770733>
- [59] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. 2005. *Probabilistic Robotics*. MIT Press.
- [60] Jonathan Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. 2014. Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation. In *Advances in Neural Information Processing Systems*. 1799–1807.
- [61] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. 2018. Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects. In *Conference on Robot Learning (CoRL)*.
- [62] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37, 3 (1989), 328–339. <https://doi.org/10.1109/29.21701>
- [63] Yunbo Wang, Bo Liu, Jiajun Wu, Yuke Zhu, Simon S. Du, Fei-Fei Li, and Joshua B. Tenenbaum. 2020. DualSMC: Tunneling Differentiable Filtering and Planning under Continuous POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*. ijcai.org, 4190–4198. <https://doi.org/10.24963/ijcai.2020/579>
- [64] Robert Williams. 2020. Opinion: I was wrongfully arrested because of facial recognition. Why are police allowed to use it? (Jun 2020). <https://www.washingtonpost.com/opinions/2020/06/24/i-was-wrongfully-arrested-because-facial-recognition-why-are-police-allowed-use-this-technology/>
- [65] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. 2018. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In *Robotics: Science and Systems (RSS)*.
- [66] Fu Xiong, Boshen Zhang, Yang Xiao, Zhiguo Cao, Taidong Yu, Joey Zhou Tianyi, and Junsong Yuan. 2019. A2J: Anchor-to-Joint Regression Network for 3D Articulated Pose Estimation from a Single Depth Image. In *International Conference on Computer Vision (ICCV)*.
- [67] Hao Xiong and Nicholas Ruozzi. 2020. General Purpose MRF Learning with Neural Network Potentials. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, Christian Bessiere (Ed.). ijcai.org, 2769–2776. <https://doi.org/10.24963/ijcai.2020/384>
- [68] Qianru Ye, Shanxin Yuan, and Tae-Kyun Kim. 2016. Spatial Attention Deep Net with Partial PSO for Hierarchical Hybrid Hand Pose Estimation. In *ECCV*.
- [69] Jonathan S Yedidia, William T Freeman, Yair Weiss, et al. 2003. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium* 8 (2003), 236–239.
- [70] Brent Yi, Michelle Lee, Alina Kloss, Roberto Martín-Martín, and Jeannette Bohg. 2021. Differentiable Factor Graph Optimization for Learning Smoothers. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE.
- [71] Kijung Yoon, Renjie Liao, Yuwen Xiong, Lisa Zhang, Ethan Fetaya, Raquel Urtasun, Richard Zemel, and Xaq Pitkow. 2019. Inference in Probabilistic Graphical Models by Graph Neural Networks. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. 868–875. <https://doi.org/10.1109/IEEECONF44664.2019.9048920>
- [72] Sung Wook Yoon, Alan Fern, and Robert Givan. 2007. FF-Replan: A Baseline for Probabilistic Planning.. In *ICAPS*, Vol. 7. 352–359.
- [73] Zhen Zeng, Adrian Röfer, and Odest Chadwicke Jenkins. 2020. Semantic Linking Maps for Active Visual Object Search. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 1984–1990. <https://doi.org/10.1109/ICRA40945.2020.9196830>
- [74] Zhen Zhang, Fan Wu, and Wee Sun Lee. 2020. Factor Graph Neural Networks. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 8577–8587. <https://proceedings.neurips.cc/paper/2020/file/61c66a2f4e6e10dc9c16ddf9d19745d6-Paper.pdf>
- [75] Stuart Zweben, Jodi Tims, and Yan Timanovsky. 2020. ACM-NDC study 2019–2020: eighth annual study of non-doctoral-granting departments in computing. *ACM Inroads* 11, 3 (2020), 26–37.

7 APPENDIX

7.1 Algorithm Pseudocode

In this section, pseudo code of the DNBP message passing algorithm is given for reference. As discussed in Section 4, this algorithm is a differentiable variant of the PMPNBP algorithm [13].

Results in this work were generated with U set to 10, while past related work [13] used $U = 1$. It was observed that this modification improved training stability during preliminary development. Note that γ is a hyperparameter that controls the resampling strategy and is set to 0.9 in our experiments. γ is used only during training; during evaluation, all M samples are drawn from $bel_d^{n-1}(X_d)$.

Algorithm 1: Message update

input :Belief set $bel_d^{n-1}(X_d) = \{(w_d^{(i)}, \mu_d^{(i)})\}_{i=1}^T$
 Incoming messages $m_{u \rightarrow s}^{n-1}(X_s) = \{(w_{us}^{(i)}, \mu_{us}^{(i)})\}_{i=1}^M$ for each node $u \in \rho(s) \setminus d$
output:Outgoing messages, $m_{s \rightarrow d}^n(X_d) = \{(\mu_{sd}^{(i)}, w_{sd}^{(i)})\}_{i=1}^M$

```

1 Draw  $(1 - \gamma^{n-1}) \cdot M$  independent samples from  $bel_d^{n-1}(X_d)$ 
2    $\{\mu_{sd}^{(i)} \leftarrow bel_d^{n-1}(X_d)\}_{i=1}^{(1-\gamma^{n-1}) \cdot M};$ 
3 Apply particle diffusion to each sampled particle
4    $\mu_{sd}^{(i)} = \mu_{sd}^{(i)} + \tau_d(\epsilon);$ 
5 Draw remaining  $\gamma^{n-1} \cdot M$  samples independently from uniform proposal distribution;
6 foreach  $\{\mu_{sd}^{(i)}\}_{i=1}^M$  do
7   for  $\ell = [1 : U]$  do
8     Sample  $\hat{X}_s^{(i)} \sim \psi_{sd}(X_s, X_d = \mu_{sd}^{(i)});$ 
9      $w_{unary}^{(i)} = w_{unary}^{(i)} + \phi_s(X_s = \hat{X}_s^{(i)}, Y_s);$ 
10   end
11    $w_{unary}^{(i)} = \frac{w_{unary}^{(i)}}{U};$ 
12   foreach  $u \in \rho(s) \setminus d$  do
13      $W_u^{(i)} = \sum_{j=1}^M w_{us}^{(j)} \times w_u^{(ij)}$  where  $w_u^{(ij)} = \psi_{sd}(X_s = \mu_{us}^{(j)}, X_d = \mu_{sd}^{(i)});$ 
14   end
15    $w_{neigh}^{(i)} = \prod_{u \in \rho(s) \setminus d} W_u^{(i)};$ 
16    $w_{sd}^{(i)} = w_{unary}^{(i)} \times w_{neigh}^{(i)};$ 
17 end
18 Associate  $\{w_{sd}^{(i)}\}_{i=1}^M$  with  $\{\mu_{sd}^{(i)}\}_{i=1}^M$  to form outgoing  $m_{s \rightarrow d}^n(X_d);$ 

```

Algorithm 2: Belief update

1 **input** : Incoming messages, $m_{s \rightarrow d}^n(X_d) = \{(w_{sd}^{(i)}, \mu_{sd}^{(i)})\}_{i=1}^M$, for each node $s \in \rho(d)$
 2 **output**: Belief set $bel_d^n(X_d) = \{(w_d^{(i)}, \mu_d^{(i)})\}_{i=1}^T$
 3
 4 **foreach** $s \in \rho(d)$ **do**
 5 Update message weights
 6 $w_{sd}^{(i)} = w_{sd}^{(i)} \times \phi_d(X_d = \mu_{sd}^{(i)}, Y_d)$ for $i \in [1 : M]$;
 7
 8 Normalize message weights
 9 $w_{sd}^{(i)} = \frac{w_{sd}^{(i)}}{\sum_{j=1}^M w_{sd}^{(j)}}$ for $i \in [1 : M]$;
 10
 11
 12 **end**
 13 Form belief set $bel_d^n(X_d) = \bigcup_{s \in \rho(d)} m_{s \rightarrow d}^n(X_d)$;
 14
 15 Normalize belief weights
 16 $w_d^{(i)} = \frac{w_d^{(i)}}{\sum_{j=1}^T w_d^{(j)}}$ for $i \in [1 : T]$;

7.2 Network Architecture & Training

For both simulated articulated tracking tasks, the network architecture for each sub-network described in Section 4 is summarized in Table 1. For the hand tracking task, each network follows the same structure as those in Table 1, with two exceptions: (1) the feature extractor, $f_s(\cdot)$, used for hand tracking is based on the architecture used by the FPHAB baseline that was introduced by Ye et al. [68]. (2) each node likelihood network, $l_s(\cdot)$, has one additional feature reduction layer of [fc(64, BatchNorm, ReLU)] preceding the layers of the corresponding network in Table 1.

The following sections describe specific implementation details used in the supervised training of DNBP. To ensure independence from spatial location, the pairwise density, pairwise sampling and diffusion sampling processes of DNBP are defined over the space of transformations between variables. Specifically, each of these networks takes as input or produces as output a translation between samples of their corresponding random variables.

Gradient Decoupling: The belief weight, $w_d^{t,(i)}$, of particle i is proportional to the product of component weights, $w_{unary_d}^{t,(i)} \times w_{unary_s}^{t,(i)} \times w_{neigh_s}^{t,(i)}$, where s is the neighbor of node d from which particle i originated (see Algorithm 2). Since each of these component weights is produced by a

NETWORK	UNIT LAYERS
f_s	5 x [conv(3x3, 10, stride=2, ReLU), maxpool(2x2, 2)]
l_s	2 x fc(64, ReLU), fc(1, Sigmoid scaled to [0.005, 1])
ψ_{sd}^ρ	4 x fc(32, ReLU), fc(1, Sigmoid scaled to [0.005, 1])
ψ_{sd}^\sim	2 x fc(64, ReLU), fc(2)
τ_s^\sim	2 x fc(64, ReLU), fc(2)

Table 1. Network parameters of learned DNBP potential functions used on both simulated articulated tracking tasks. Note $s, d \in \mathcal{V}$, and $(s, d) \in \mathcal{E}$. Unary potentials: $l_s(f_s(\cdot))$. Pairwise potentials: $\{\psi_{sd}^\rho, \psi_{sd}^\sim\}$. Particle diffusion: τ_s^\sim .

separate potential network (either ϕ_d , ψ_{sd} , or ψ_{sd}^ρ respectively), direct optimization of the belief density will lead to interdependence of the potential network gradients during training. In the context of DNBP, interdependence between different potential functions is inconsistent with the factorization given in \mathcal{G} . Tompson et al. [60] describe a similar phenomenon they refer to as gradient coupling which was addressed by expressing a product of features in log-space which “decouples” the gradients.

To avoid interdependence between potential functions during training, we consider the *partial*-belief densities which are defined for each node $d \in \mathcal{V}$ as mixtures of Gaussian density functions:

$$\overline{\text{bel}}_{d,\text{unary}_d}^t(X_d) = \sum_{i=1}^N w_{\text{unary}_d}^{t,(i)} \cdot \mathcal{N}(X_d; \mu_d^{(i)}, \Sigma) \quad (9)$$

$$\overline{\text{bel}}_{d,\text{unary}_{\rho(d)}}^t(X_d) = \sum_{i=1}^N w_{\text{unary}_s}^{t,(i)} \cdot \mathcal{N}(X_d; \mu_d^{(i)}, \Sigma) \quad (10)$$

$$\overline{\text{bel}}_{d,\text{neigh}_{\rho(d)}}^t(X_d) = \sum_{i=1}^N w_{\text{neigh}_s}^{t,(i)} \cdot \mathcal{N}(X_d; \mu_d^{(i)}, \Sigma) \quad (11)$$

Using these definitions, direct interaction between the potential networks’ gradients is avoided by maximizing the product of partial-beliefs at the ground truth of each node in log space. The product of partial-beliefs is defined:

$$\overline{\text{bel}}_d^t(X_d) = \overline{\text{bel}}_{d,\text{unary}_d}^t(X_d) \times \overline{\text{bel}}_{d,\text{unary}_{\rho(d)}}^t(X_d) \times \overline{\text{bel}}_{d,\text{neigh}_{\rho(d)}}^t(X_d) \quad (12)$$

Unary Potentials: During training of DNBP, only those gradients derived from the belief update of each node are used to update the corresponding node’s unary potential network parameters. Any gradients derived from the outgoing messages of a particular node are manually stopped from propagating to that node’s unary network. This is done to avoid confounding the objective functions of neighboring nodes, which each rely on the others’ unary network during message passing. This approach can be implemented with standard deep learning frameworks by dynamically stopping the parameter update of each unary network depending on where in the algorithm its forward pass was registered.

Pairwise Density Networks: To speed up and stabilize the training of pairwise density potential networks, the following substitution is made during training. While calculating $w_{sd}^{(i)}$ for outgoing message i from node s to d , the summation over incoming messages from $u \in \rho(s)$ to s is replaced by a single evaluation of:

$$W_u^{(i)} = \psi_{sd}(X_s = x_s^*, X_d = \mu_{sd}^{(i)}) \quad (13)$$

where x_s^* is the ground truth label of sender node s . This change improves inference time and reduces memory demands by removing a summation over M particles while also providing more stable training feedback to the network. This substitution is removed at test time after training is complete.

Pairwise Sampling Networks: The pairwise sampling networks, $\psi_{s,d}$, take a random sample of Gaussian noise as input and generate conditional samples using the following rule:

$$\epsilon \sim \mathcal{N}(0, 1) \quad (14)$$

$$x_{s|d} = x_d + \psi_{s,d}(\epsilon) \quad (15)$$

where $x_{s|d}$ is the sample of variable X_s conditioned on neighboring sample x_d and where ϵ is a noise vector sampled from a zero-mean, unit variance multivariate Gaussian distribution with

$\dim(\epsilon) = 64$. Similarly, for sampling in the opposite conditioning direction (node d conditioned on s), memory efficiency is gained by reusing the $\psi_{s,d}^{\sim}$ network but negating the sampled translation.

7.3 Double Pendulum Clutter

As summarized in Section 5.1, the double pendulum dataset was generated using a modified version of the OpenAI [7] Acrobot environment. Synthetic geometric shapes are rendered into each image of the dataset to simulate noisy, cluttered environments. All simulated clutter on the double pendulum task is generated according to the following parameters: 50% of clutter is rendered visually beneath the pendulum while the remaining 50% is rendered on top of the pendulum. For dynamic clutter, each geometry simulates motion using a random, constant position velocity (\dot{x}, \dot{y}) and orientation velocity ($\dot{\theta}$). Position velocities are sampled from $\mathcal{N}(0, 0.025)$. Orientation velocities are sampled from $\mathcal{N}(0, 0.05)$. Clutter is simulated as either rectangles with 80% probability or circles with 20% probability. Clutter rectangles are sized randomly with length of $\max(0, l \sim \mathcal{N}(0.2, 0.05))$ and height of $\max(0, h \sim \mathcal{N}(0.8, 0.2))$. Color of clutter rectangles is randomly chosen with RGB of (0, 204, 204) or (245, 87, 77). Clutter circles are sized randomly with radius of $\max(0, r \sim \mathcal{N}(0.1, 0.1))$ and colored randomly with RGB of (204, 204, 0) or (96, 217, 63). Size and color distributions were chosen to ensure clutter visually resembles the double pendulum parts. The position of each clutter geometry was randomly initialized within 1.5x the extent of the image boundary.

The training and validation datasets were distributed evenly among clutter ratios of $[0, 0 - 0.04,$ and $0.04 - 0.1]$. For the training/validation sequences that included clutter, the number of clutters rendered beneath and on top of the double pendulum was individually randomly sampled from independent Binomial distributions using $n = 15, p = 0.3$. To generate the test set, which was uniformly distributed among clutter ratios as described in Section 5.1, rejection sampling was used with variable numbers of rendered geometries.

7.4 Articulated Spider Model

Data for the articulated spider tracking task of Section 5.5 was simulated using the Pillow [11] image processing library. Three revolute-prismatic joints are all centrally located and treated as the root of the spider’s kinematic tree. The remaining three revolute joints are attached to pairs of links, forming three distinct ‘arms’ of the spider. Each joint is rendered as a yellow circle while the six rigid-body links are rendered as distinct red, green or blue rectangles respectively. Size parameters that follow are with respect to rendered image size of 500x500px. The three inner revolute-prismatic joints include rotational constraints limiting each to a non-overlapping 120° range of articulation as well as prismatic constraints limiting the extension to within $[20, 80]$ pixels of translation. The three purely revolute joints are constrained to rotations between $\pm 35^\circ$ with respect to their local origins. Each rigid-body link has width of 20px and height of 80px pixels while each joint has radius of 10px.

For every simulated sequence, the spider is initialized with uniformly random root position within the central 180x180px window and uniformly random root orientation from $[0, 2\pi]$. Furthermore, each joint state is initialized uniformly at random within its particular articulation constraints. The spider is simulated with dynamics using randomized, constant root, and joint velocities with respect to a time step (dt) of 0.01. The root’s position velocities (\dot{x}, \dot{y}) are each sampled from an equally weighted 2-component Gaussian mixture with means (+24, -24) and standard deviations (15, 15). Whereas, the root’s orientation velocity ($\dot{\theta}$) is sampled from an equally weighted 2-component Gaussian mixture with means (+0.3, -0.3) and standard deviations (0.1, 0.1). Each rotational joint’s velocity is sampled from an equally weighted 2-component Gaussian mixture with means (+0.3, -0.3) and standard deviations (0.1, 0.1). Similarly, each prismatic joint’s velocity is sampled from an

equally weighted 2-component Gaussian mixture with means $(+500, -500)$ and standard deviations $(60, 60)$. Note that if any joint reaches an articulation limit during simulation, the direction of its velocity is reversed.

7.5 Articulated Spider Clutter

Clutter generation for the articulated spider tracking task follows a similar generation process as was used for the double pendulum task. Clutter parameters that follow are with respect to rendered image size of 500x500px and time step (dt) of 0.01. 50% of clutter is rendered beneath and 50% is rendered on top of the spider. For dynamic clutter, each geometry simulates motion using a random, constant position velocity (\dot{x}, \dot{y}) and orientation velocity $(\dot{\theta})$. Position velocities are sampled from $\mathcal{N}(0, 3)$ while orientation velocities are sampled from $\mathcal{N}(0, 0.05)$. Clutter is simulated as either a rectangle with 70% probability or a circle with 30% probability. Clutter rectangles are sized randomly with length of $\max(0, l \sim \mathcal{N}(20, 3))$ and height of $\max(0, h \sim \mathcal{N}(80, 5))$. The color of clutter rectangles is chosen uniformly at random from the same colors as were used for the spider arms. Clutter circles are sized randomly with a radius of $\max(0, r \sim \mathcal{N}(10, 3))$ and colored yellow to match the color of the spider's joints. The position of each clutter geometry was randomly initialized within the image boundary.

For the training/validation sequences that included clutter, the number of clutter shapes rendered beneath and on top of the double pendulum was each randomly sampled from independent Binomial distributions using $n = 10$, $p = 0.5$. The test set was generated with uniformly distributed clutter ratios, as described in Section 5.1, using rejection sampling with variable numbers of rendered geometries.

7.6 Learned Pairwise Inspection

As further validation of DNBP, the learned pairwise potentials are inspected in Fig. 11. The normalized histogram of pairwise translations computed from the training set for $X_1 - X_0$ (top) and $X_2 - X_1$ (bottom) are shown in the left column of Fig. 11. The middle column shows the normalized histogram of samples from learned pairwise sampler networks, $\psi_{0,1}^{\sim}(\cdot)$ and $\psi_{1,2}^{\sim}(\cdot)$. Finally, the right column shows output from the learned pairwise density networks, $\psi_{0,1}^{\rho}(\cdot)$ and $\psi_{1,2}^{\rho}(\cdot)$, generated with 100x100 uniform samples across pairwise translation space. The qualitative similarity between

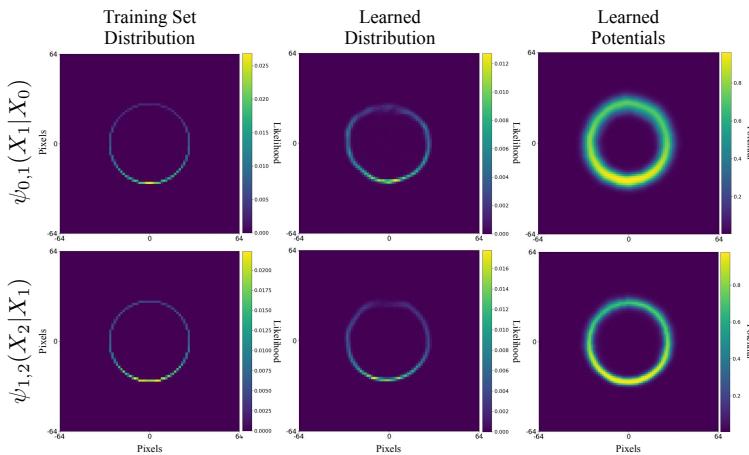


Fig. 11. Inspection of learned pairwise potentials from double pendulum tracking.

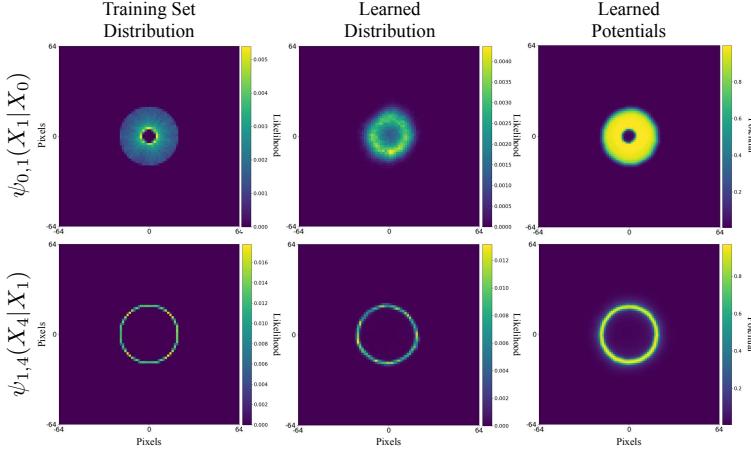


Fig. 12. Inspection of DNBP’s learned pairwise potentials from spider tracking. Only two of the six are shown to avoid redundancy, remaining four show very similar output.

each learned potential model and the corresponding true distribution of pairwise translations indicates that DNBP is successful in learning to model each pairwise potential factor. The circular pairwise relationships are explained by the fact that each pair of double pendulum keypoints is related by a revolute joint. The effect of simulated gravity in the double pendulum experiment can be observed by the bias of each pairwise potential in favor of the lower half of each plot as indicated by increased likelihood.

The pairwise potential functions learned by DNBP in the spider tracking task are visualized as was done in the double pendulum task. Fig. 12 shows qualitative output from two of the six models. Only two are shown to avoid redundancy; chosen results are representative of remaining four potential functions. The left column of Fig. 12 shows the normalized histogram of pairwise translations as computed from the training set for $X_1 - X_0$ (top) and $X_4 - X_1$ (bottom). The middle column of Fig. 12 shows the normalized histogram of samples from learned pairwise sampler networks, $\psi_{0,1}(\cdot)$ and $\psi_{1,4}(\cdot)$. Finally in the right column of Fig. 12, uniformly sampled output (100x100 samples across pixel space) of the learned pairwise density networks is shown. Once again, the visual similarity between each learned potential function and the corresponding true distribution of pairwise translations is an indicator that DNBP is successful in learning to model each pairwise factor. Observe that the learned potential functions for $\psi_{1,4}(\cdot)$, which correspond to a revolute articulation, show no bias in favor of the downward configuration. This result is notably different from the potential functions learned on the double pendulum task and can be explained by the absence of gravity in the spider simulation. Similarly, the learned models for $\psi_{0,1}(\cdot)$ on the spider task exhibit a torus shape due to the effect of prismatic motion associated with the corresponding joint’s articulation type and constraint.

7.7 Hand Tracking Results

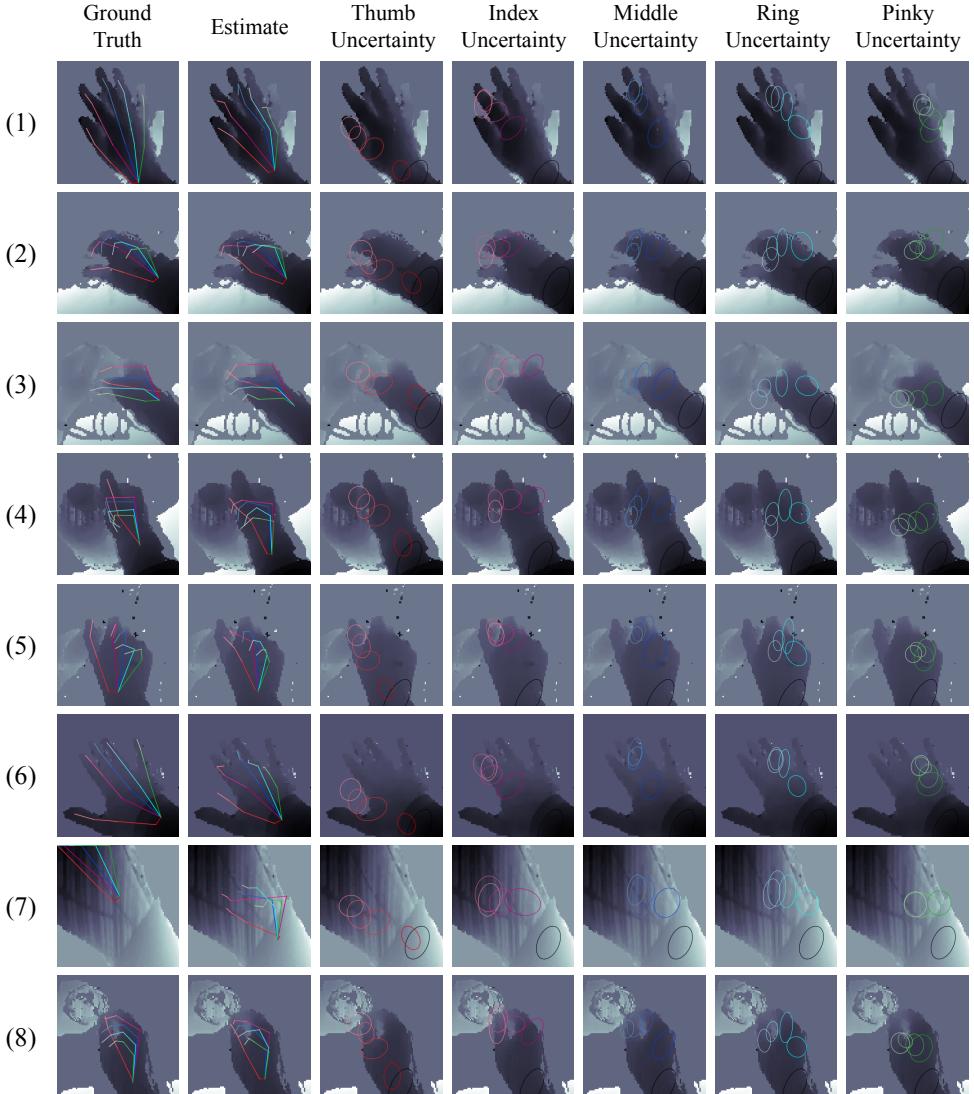


Fig. 13. Output from DNBP on randomly sampled frames of the hand pose tracking experiment. Visualized model uncertainty is calculated from the marginal belief estimates of DNBP as 1 standard deviation in the horizontal and vertical dimensions respectively as calculated by estimated covariance of belief particles. Uncertainty in depth dimension is not visualized.