Opening the Black Box: Hierarchical Sampling Optimization for Estimating Human Hand Pose

Danhang Tang[†‡] Jonathan Taylor[†] Pushmeet Kohli[†]
Cem Keskin[†] Tae-Kyun Kim[‡] Jamie Shotton[†]

[†]Microsoft Research [‡]Imperial College London

Abstract

We address the problem of hand pose estimation, formulated as an inverse problem. Typical approaches optimize an energy function over pose parameters using a 'black box' image generation procedure. This procedure knows little about either the relationships between the parameters or the form of the energy function. In this paper, we show that we can significantly improve upon black box optimization by exploiting high-level knowledge of the parameter structure and using a local surrogate energy function. Our new framework, hierarchical sampling optimization*, consists of a sequence of predictors organized into a kinematic hierarchy. Each predictor is conditioned on its ancestors, and generates a set of samples over a subset of the pose parameters. The highly-efficient surrogate energy is used to select among samples. Having evaluated the full hierarchy, the partial pose samples are concatenated to generate a full-pose hypothesis. Several hypotheses are generated using the same procedure, and finally the original full energy function selects the best result. Experimental evaluation on three publically available datasets show that our method is particularly impressive in low-compute scenarios where it significantly outperforms all other state-of-the-art methods.*

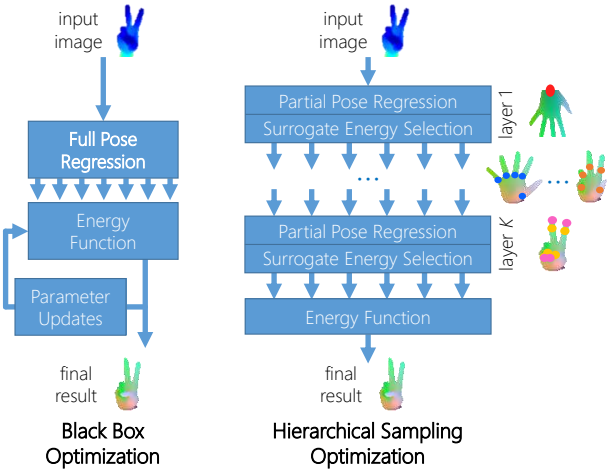


Figure 1: **Opening the black box.** A typical black box optimization approach (left) will regress candidate full poses and then iteratively update them based on the evaluation of the energy function. Our approach (right) instead regresses partial poses in a layered kinematic hierarchy, using a surrogate energy function to keep only the best partial poses. Our approach can achieve higher accuracy than black box optimization, even without iteration.

1. Introduction

Estimating the articulated pose of the human hand presents unique challenges for computer vision algorithms. The search space is high-dimensional, the hand can appear in essentially any 3D global orientation, there are frequent self-occlusions, and fingers have (locally) similar appearance. Early success [1, 20, 4] has been accelerated by the arrival of consumer depth cameras [13, 7, 12, 16, 25, 17, 21, 19], but the problem is far from solved.

Modern pose estimation algorithms attack the problem with a 'hybrid' discriminative/generative approach. First, a learned discriminative component makes a prediction directly from the input image about either the pose parameters themselves [17, 9] or some intermediates such as fingertip positions [16], hand joint positions [25], or hand parts [7, 19]. Second, a generative component combines discriminative and motion-based predictions in an optimization over the pose parameters.

Also known as 'analysis by synthesis' or 'inverse graphics', generative approaches aim to infer the hand pose parameters that allow one to *generate* or synthesize the input image. Given a model of the hand, its pose (and sometimes shape [8]) parameters are optimized to minimize an energy function. An ideal energy function is the reconstruction error: the distance between the observed image and a synthetic rendering of the model. Unfortunately, the recon-

struction error is hard to differentiate (though see *e.g.* [11]) and non-convex in the model parameters. One typically therefore employs 'black box' optimization strategies such as particle swarm optimization [14, 16, 17] that only require function evaluations, without gradients. While black box optimization can lead to high-quality results, it typically requires a large number of function evaluations and is therefore computationally expensive.

In this paper we propose to open up the black box and exploit our high-level knowledge about (i) the rendering process, and (ii) the relationship between the pose parameters. Our approach has a tight coupling between the generative and discriminative aspects, and results in high quality pose estimates at very high efficiency. Instead of directly regressing and optimizing the full hand pose (*e.g.* [17]), our new framework, termed *hierarchical sampling optimization*, builds up the pose parameters layer by layer, guided by the kinematic hierarchy. At each layer, a new set of candidate sample partial poses is regressed, conditioned on the result at the previous layer. A surrogate energy function, based on a high-level view of the rendering process, is introduced to quickly cull the bad samples. This process is repeated to generate multiple full pose hypotheses. Since they are rather accurate, instead of using PSO as in [17], we can simply rank and choose the best one as final output.

To demonstrate the efficacy of our approach, we perform an exhaustive evaluation of our method on three publicly available datasets. The experimental results show that our method outperforms six state-of-the-art methods on these datasets while considerably improving the efficiency compared to black box approaches.

To the best of our knowledge, this is the first discriminatively trained pipeline for inverting the graphics rendering procedures that is guided by the relationships between the arguments of the generative procedure. We expect that our work can be more broadly applied to other vision tasks that can be formulated as inverse problems.

Related Work A number of methods have been proposed in the literature that approach hand pose estimation as an inverse problem. A popular strategy adopted by such algorithms is to use particle swarm optimization ('PSO') for solving the inverse problem [14]. PSO hypothesizes a 'swarm' of particles (pose hypotheses) that are iteratively perturbed based on their own and the other particles' energies. However, the large number of particles required will easily consume the entire rendering power of a high-end GPU to achieve interactive frame rates. Sharp *et al.* [17] use random forests to produce hand pose hypotheses that are then refined further using PSO. Similar to other hybrid approaches proposed in the literature, the work of [17] treats the mapping from the model parameters to the rendered image as a black box. In doing so, it ignores any available knowledge about the generation process or the relationship between different parameters of the model.

Some approaches break the problem into *independent* pieces, predicting joint locations using forests [7] or neural networks [25], and stitching these together using inverse kinematics. Tang *et al.* [22] approach the problem as a coarse-to-fine search, by leveraging the latent relation between joints. A more explicit way of breaking the problem is to exploit the kinematic structure, which has been studied in the field of human pose estimation [2, 6]. Sun *et al.* [21] decompose the output using a hierarchical kinematic structure. However, unlike our proposal, they do not refine the predictions and also do not reason about the likelihood of partial hypotheses as they are built. This prevents them from pruning bad hypotheses quickly.

2. The Pose Estimation Inverse Problem

This section formulates hand pose estimation as an inverse problem. We denote the input depth image, viewed as a function of pixel location u, as $Z(u) \in [0, \infty)$. Treating pose estimation as an inverse problem explicitly assumes that a hand with pose θ gives rise to a rendered depth image $R_\theta(u)$ with the same range as $Z(u)$. The goal is to then invert the process by finding the parameters θ such that $R_\theta \simeq Z$. As our model of the true process which gave rise to Z is imperfect, we cannot expect perfect equality, and instead aim to minimize an energy function, the 'golden energy', that measures the error in reconstructing Z with R_θ.

Before detailing the golden energy, we first describe our parameterization of the human hand. We will later highlight the inherent structure in the pose parameters that our optimization approach exploits. The full pose vector θ is arranged in a standard kinematic tree defined by the skeletal structure of the hand (see Fig. 2 top right for an illustration).[1] We use the wrist as the root of the tree, at which the global translation and rotation of the hand is specified. The kinematic tree structure then approximates the human anatomical hand skeleton, with each joint's parameters specifying its *rotation* relative to its parent. As is standard we assume a fixed hand shape that specifies the *translations* of each joint relative to its parent.

2.1. The golden energy

Given a full pose vector θ, how can we evaluate how well it 'fits' an observed test image? We exploit an energy function dubbed the 'golden' energy [17], which uses θ to render a synthetic depth image of the hand and then compares each pixel using a robust L1 term:

$$E_{\mathrm{Au}}(\theta) = \sum_u \min(|Z(u) - R_\theta(u)|, \tau) \,, \qquad (1)$$

[1] MCP, PIP, and DIP stand for the metacarpophalangeal, proximal interphalangeal, and distal interphalangeal joints, respectively.

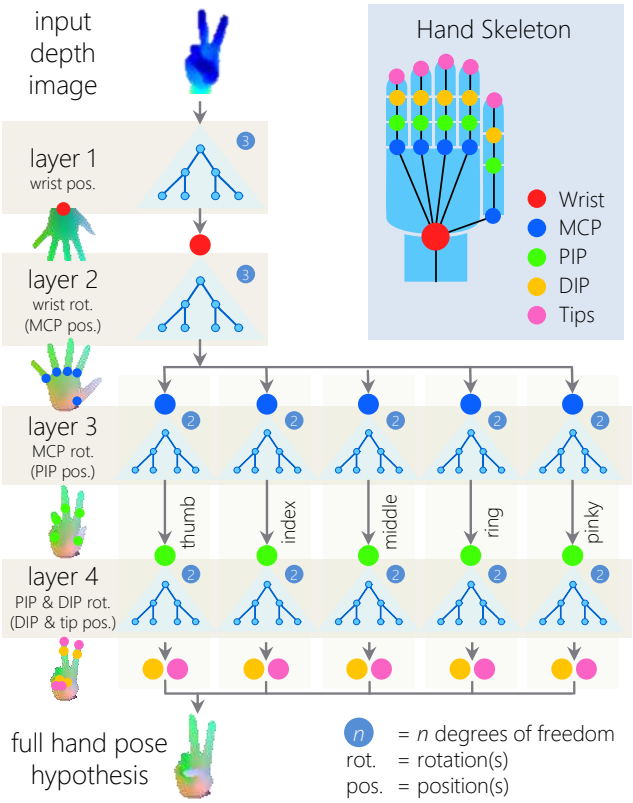


Figure 2: **Layers in the pose hierarchy.** Our approach regresses a full hand pose hypothesis in four layers. At each layer, we predict *particular subsets* of pose parameters, conditioned on all the previous layers' results. The layers are aligned with the kinematic tree, shown top right: layers 1 and 2 respectively predict wrist position and rotation; layer 3 contains one forest per finger, each of which predicts its respective MCP rotations (flexion and abduction); layer 4 also specializes per finger, and jointly predicts the (highly correlated) flexions for the PIP and DIP joints. Finally, we concatenate the partial poses resulting in a hypothesis of the full hand pose.

where τ is a truncation threshold, here set to 100mm.

The golden energy is effective at penalizing incorrect poses, including common problems such as 'model-over-background' and 'background-over-model'. However, computing E_{Au} is expensive: due to self-occlusions, a full rendering must be performed for each candidate value of θ to obtain R_θ. Black box strategies to optimize E_{Au} therefore tend to require high-end GPU compute to render the possibly thousands of hypothesized poses that are needed per frame [13, 17].

2.2. The silver energy

One of the main contributions of this paper is to show that we can achieve state-of-the-art results that optimize the golden energy Eq. 1, while requiring orders of magnitude fewer full golden energy evaluations. The key insight is that by exploiting the special kinematic structure of the pose parameter vector, we can decompose the problem into subproblems which can be tackled independently using an efficient surrogate (or proxy) energy, which we dub the 'silver' energy. Our silver energy, E_{Ag}, is inspired by [16], but differs in that it is applied to each joint separately rather than to a full pose vector.

In our approach, we decompose θ into several *partial poses* that follow the four layers specified in Fig. 2. Layers 3 and 4 further decompose θ by finger f, giving the complete set of partial poses as: $\Theta = \{\theta_1, \theta_2\} \cup \{\theta_{3f}, \theta_{4f} : f \in \{\underline{\text{t}}\text{humb}, \underline{\text{i}}\text{ndex}, \underline{\text{m}}\text{iddle}, \underline{\text{r}}\text{ing}, \underline{\text{p}}\text{inky}\}\}$.

The representation of partial poses is different for different layers of the framework. For instance, layer 1 predicts a global translational offset (the vector from the image centroid to the wrist position), and thus $\theta_1 \in \mathbb{R}^3$ is a 3D Euclidean vector, while layer 2's parameter set θ_2 encodes the global (palm) rotation as a unit quaternion [13, 17], and so $\theta_2 \in \mathbb{R}^4$. For the rest of the bones, Euler angles are employed to represent bone rotations: the MCP joint has two DoF (flexion and abduction) so $\theta_{3f} \in \mathbb{R}^2$, whilst the PIP and DIP joints have only 1 DoF each but are lumped together so that θ_{4f} lies in $\mathbb{R}^2$ also. Note that while the above parameterization has been presented specifically for the human hand, our approach could straightforwardly be extended to arbitrary tree structures.

For notational convenience we will use l to uniquely index any of the 12 partial poses, and 'layer l' to refer to the layer that contains the partial pose with index l. Additionally, we will use $\bar{\theta}_l$ to denote partial pose θ_l concatenated with the partial poses of layer l's ancestors.

The silver energy $E_{\text{Ag}}(\bar{\theta}_l)$ is defined in the following manner for each partial pose θ_l. We first use standard forward kinematic transformations (see *e.g.* [24]) to compute the set of *positions* of the *children* bones C_l of partial pose θ_l (the colored circles below each layer in the left column of Fig. 2). We write $\rho_l(\bar{\theta}_l)$ to denote this set of positions.[2] The silver energy is then computed given layer l's predicted child positions $\rho_l(\bar{\theta}_l)$ as

$$E_{\text{Ag}}(\bar{\theta}_l) = \sum_{x \in \rho_l(\bar{\theta}_l)} [B(x) + D(x)] \ . \qquad (2)$$

The first term encourages each child to lie inside the ob-

[2] In more detail: Layer 1 predicts the wrist translation from the image centroid θ_1, which, added to the observed centroid $\text{CoM}(Z)$ gives the singleton set $\rho_1(\bar{\theta}_1) = \{\theta_1 + \text{CoM}(Z)\}$. Conditioned on $\bar{\theta}_1$, layer 2 predicts the wrist rotation θ_2, which uniquely determines the positions of the five MCP joints as the set $\rho_2(\bar{\theta}_2)$. Layer 3 predicts the MCP rotation for each finger f, which uniquely determines the position of the PIP joint as the singleton set $\rho_{3f}(\bar{\theta}_{3f})$. Finally, layer 4 jointly predicts PIP and DIP rotations, giving the DIP and fingertip positions as set $\rho_{4f}(\bar{\theta}_{4f})$.

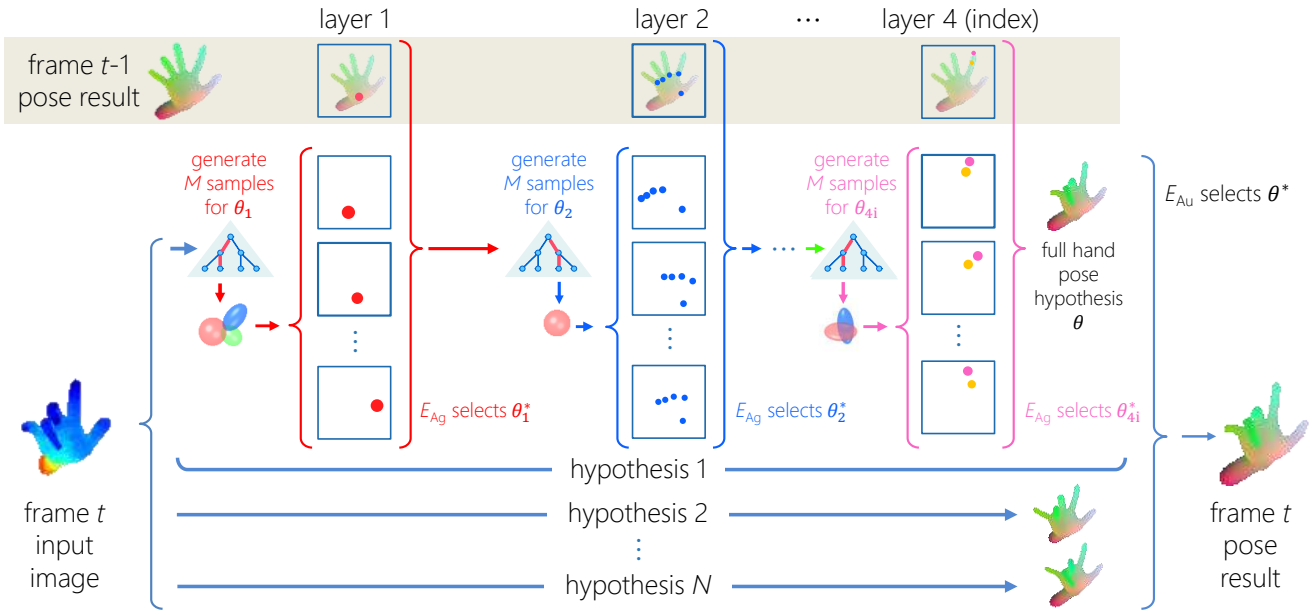


Figure 3: **Hierarchical sampling optimization.** See text for details. Note that for simplicity, we only show the optimization for the index finger at layer 4, which has parameters $\theta_{4\mathrm{i}}$.

served silhouette, and is defined as

$$B(x) = \mathrm{dist}(\pi(x); Z)\ , \tag{3}$$

where $\pi(x)$ projects a 3D position x into the image, and $\mathrm{dist}(u; Z)$ represents the value of the 2D truncated distance transform of the silhouette of hand input image Z at pixel u. The distance values are truncated to 10 pixels (outside), and then normalized to $[0, 1]$ outside and all 0 inside.

The second term in the silver energy aims to ensure the predicted depth at the child position roughly agrees with the observed depth. It is defined as

$$D(x) = \begin{cases} 0 & \text{if } B(x) > 0 \\ \Psi\left(\frac{Z(\pi(x)) - x_{\mathrm{z}} - \tau_1}{\alpha}\right) & \text{if } Z(\pi(x)) - x_{\mathrm{z}} > \tau_1 \\ \Psi\left(\frac{x_{\mathrm{z}} - Z(\pi(x)) - \tau_2}{\alpha}\right) & \text{if } x_{\mathrm{z}} - Z(\pi(x)) > \tau_2 \\ 0 & \text{otherwise} \end{cases}\ , \tag{4}$$

where x_{z} represents the z-coordinate of 3D position x, $\Psi(x) = \max(1, x)$. The thresholds τ_1 and τ_2 penalize the child positions living too far in front or behind the observed depth image. The whole silver energy can be considered as an efficient approximate of a 3D distance transform, forming a 'safe zone' within $[\tau_1, \tau_2]$ behind the point cloud. If a joint is outside this region, a penalty is given. We set τ_1 to 15mm for the wrist and 5mm for other joints, whereas τ_2 is simply set to 250mm. α is set to 10mm to allow a soft penalty around the 'safe zone' border. These values are chosen based on the general size of a human hand.

Note that in contrast to the golden energy, the silver energy can be computed extremely efficiently and can also be applied to partial poses $\bar{\theta}_l$. In the following sections we describe our algorithm for using the silver energy to quickly filter out bad partial pose hypotheses, greatly restricting the search space when we optimize the golden energy.

3. Hierarchical Sampling Optimization

We now describe our main contribution, hierarchical sampling optimization (HSO), a strategy for optimizing energies such as the golden energy. We start by providing a high-level overview of the method. We then describe how the method can efficiently sample partial poses using a regression forest, and finally move on to explain how the predictors used in the hypotheses generation phase are discriminatively trained.

3.1. Overview

The HSO approach is illustrated in Fig. 3. At a high-level, it looks much like a standard black box inverter: we generate N full hand pose hypotheses, and then select the pose θ^* with the lowest golden energy $E_{\mathrm{Au}}(\theta^*)$. However, unlike a standard approach, each hypothesis is built up by following the kinematic structure of human hand which was described in the previous section. Following many other approaches *e.g.* [17], we assume that the foreground pixels are pre-segmented.

Each layer l takes an *evaluation position* p_l and generates M sample values for the partial pose θ_l. Concatenating each partial pose sample with its ancestors' partial poses gives $\bar{\theta}_l$, allowing us to compute the sample's child positions $\rho_l(\bar{\theta}_l)$

and thereby the silver energy E_{Ag}. The sample with the minimal silver energy value is selected, giving θ_l^*. Temporal information can optionally be incorporated by additionally evaluating the best partial pose θ_l^* from the previous frame under E_{Ag}. The technique of multiple output with selection has been proved to be effective for reducing error [5]. In our method it is embedded into each level of hierarchy in order to reduce accumulated error, an inherent problem of all hierarchical methods.

We then proceed to each child l' at the next layer, conditioning on the result θ_l^* just obtained by taking the new evaluation position $p_{l'}$ from the relevant finger's output child position in $\rho_l(\bar{\theta}_l^*)$. For the first layer, we use the image centroid for the evaluation position, *i.e.* $p_1 = \text{CoM}(Z)$. After layer 4, the full pose vector θ is reconstructed by simply concatenating each layer's best result.

3.2. Sampling partial poses

The only remaining question is how to sample the partial poses θ_l. While in theory any sampling strategy could be used (*e.g.* sampling from a learned Gaussian, or from the prediction made by a convolutional neural network), we use a discriminatively trained regression forest. A forest F_l is trained to predict each partial pose θ_l. The samples for θ_l are generated by evaluating the forest *at just a single pixel*: the projection of 3D evaluation position p_l into the image. The forest itself is completely standard and uses well-known pixel-pair comparison features; see below for details. At each leaf node of forest F_l we store a learned Gaussian mixture model over the parameters θ_l. We can thus quickly draw M samples from this mixture model to generate the samples of θ_l required for the hierarchical sampling optimization. Note that the forest prediction, the mixture model sampling, and the silver energy evaluation are all extremely efficient.

3.3. Regression forests: testing and training

A regression forest [3] is simply an ensemble of decision trees, where at test time each internal tree node routes data to its left or right child-node by applying a threshold to a (possibly non-linear) projection of the input features. Each input ultimately ends up in a leaf node, where a distribution is stored for sampling as described above.

We employ an adaptation of the commonly used two-pixel difference function [18]:

$$\phi(p_l|\delta_1,\delta_2) = Z(\pi(p_l) + \frac{\delta_1}{p_l^z}) - Z(\pi(p_l) + \frac{\delta_2}{p_l^z}) \qquad (5)$$

where δ_1 and δ_2 are 2D offsets pointing to two neighboring pixels centered at the 2D projection π of p_l (which has z-coordinate p_l^z), and as previously $Z(u)$ retrieves a depth value given 2D image position u. Note that unlike [18], we use the *predicted* depth value p_l^z instead of the observed depth $Z(\pi(p_l))$ to scale-normalize the offsets, as it is possible that the observed depth is missing due to sensor noise.

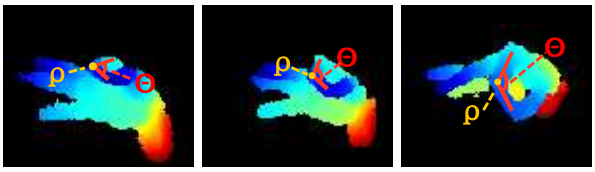


Figure 4: 3 frames from NYU dataset[25]. Due to global rotation, the same gesture looks rather different across these frames. Although the 3D location ρ changes, joint angle θ stays the same (in 3D) and trackable.

When training each tree, unlike previous methods that are applied to all foreground pixels [7, 23], in theory we only need 1 training examples corresponding to the evaluation position p_l per training image Z, paired with its output label parameters θ_l. However in practice, to be more robust to sensor noise and to minimize error accumulation at test time, we jitter each p_l with random offsets within the range of ± 5 pixels to generate 5 training examples per image.

As noted above, while θ_1 is a 3D offset vector (indicating the translation from the centroid to the wrist location), all other partial poses θ_l represent rotation angles. If one simply labels each training sample with angles and trains forest F_l accordingly there can be problems because the local joint rotations we are trying to estimate do not necessarily correlate with changes in appearance (and thus the feature responses of Eq. 5). This is illustrated in Fig. 4, where local appearance varies significantly in these 3 frames due to global rotation changes, but the index finger's joint angle that we are aiming to predict stays the same. Further, generating an output distribution G involves angle interpolation on a circular domain, which is known to have artifacts such as *gimbal lock*. Using quaternions as in [13, 17] can help to avoid this problem, but will also introduce new problems such as sign ambiguities in the interpolation. To avoid this we apply the following simple fix. In addition to the parameter set θ_l, we also record a 3D offset vector ζ_l to the children positions ρ_l. In particular, each data point is labeled with a tuple (ζ_l, θ_l), where $\zeta_l = \text{vec}(\{x - p_l : x \in \rho_l\})$ indicates a vector of 3D offsets from the current evaluation point p_l to each of its children output positions in ρ_l.

At training time, the 3D offset vectors are used to minimize the following reduction-in-spatial-variance objective,

$$Q(S_l) = \text{tr}(\Sigma^{S_l}) - \sum_k^{\{\text{l,r}\}} \frac{|S_l^k|}{|S_l|} \left(\text{tr}\left(\Sigma^{S_l^k}\right)\right), \qquad (6)$$

where $\text{tr}(\cdot)$ is the trace function and Σ is the sample covariance matrix of the offset vectors which is well-correlated with depth feature in Eq. 5. $k \in \{\text{l}, \text{r}\}$ indicates the left or right child node.

Leaf Model Likewise, we then use mean shift mode detection to cluster the 3D offsets (with a bandwidth 0.01m). After that data points of each cluster should be sufficiently close to each other so that interpolation on the angles in θ_l is not a problem. This allows us to directly fit a GMM G_l to θ_l. In the case of a quaternion parameter, the sign of every θ_l can be checked against the mean of its cluster, in order to avoid any sign ambiguities. As these 3D offset vectors were only used as a proxy for the true rotational parameters of interest during training, we can directly sample joint angles from G_l at test time without worrying about them.

4. Experiments

4.1. Dataset

We conduct our experiments on 3 publicly available datasets: Microsoft Research Synthetic Hand Dataset (MSHD) [17], ICVL [22] and NYU [25] , for they have spanned a wide range of perspectives (see Table 1). Note that the ICVL and NYU datasets only provide 3D joint locations as label, whereas our method require joint angles for training. To tackle this, inspired by the inverse kinematic step in [25], we use PSO to fit their groundtruth joint locations and recover angles. All experiments are conducted with Intel i7, 32GB RAM and an Nvidia Quadro K600.

Table 1: Description of datasets

Dataset	Sensor	Description
MSHD [17]	Kinect2	synthetic data, full range of views, individual frames
ICVL [22]	Intel	real data, restricted range of views, fast movement
NYU [25]	PrimeSense	real data (high noise), medium range of views, slow movement

4.2. Self-comparison

To analyze the contributions of our method, a set of self-comparison experiments are conducted on the ICVL dataset [22] for its relatively compact size. Each forest is empirically trained with 3 trees and maximum depth of 15.

We first conduct an experiment to reveal the impact of of hierarchical optimization, by comparison between these two baselines: $M = 1$ (*i.e.* no per-joint optimization) and $M = 30$. To analyze the boost in both accuracy and efficiency, we vary N to obtain a curve showing the trade-off between them (Fig. 5(a)). For $M = 1$, N is varied from 100 to 1000, with a step size of 100; for $M = 30$, N can be much smaller. So we choose N from 5 to 20 with step size of 5. As Fig. 5(a) shows, the average error drops more drastically when $M = 30$, without increasing too much of time budget. From the accuracy prospective, to achieve the same error of 14.5mm, $M = 30$ only requires 17% time cost of $M = 1$.

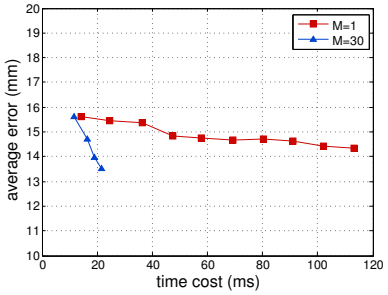
Our second attempt is to perform a parameter analysis on M and N. To uniformly sample from all trees, M is always set to multiple of $T = 3$. As in Fig. 5(b), the error reduces 2mm and 4mm for the first step of N and M respectively. However, once they are combined ($N = 10, M = 15$), accuracy is improved by 10mm. After $N = 40$ and $M = 30$, the error is still decreasing but converged.

4.3. Comparison with Prior Work

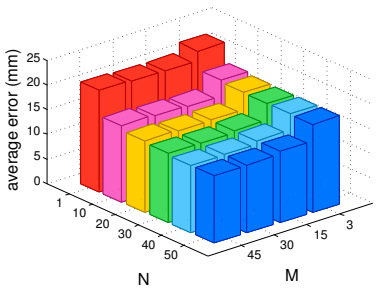
We then move on to compare HSO with six state of the art methods including Sharp *et al*. [17], Sun *et al*. [21], Tompson *et al*. [25], Latent Regression Forest (LRF) [22], Keskin *et al*. [7] and Intel® Perceptual Computing (PXC) [12]. These methods cover a wide spectrum of sensors, perspectives of challenges and different methods from decision forests to deep neural networks. All their results are either obtained from the authors or implementations that appear in previous literature. In all cases, each HSO is trained with 3 trees and maximum depth of 15 empirically. Note that in the experiments of ICVL and NYU, only one HSO is needed, whereas 128 HSOs are needed with MSHD dataset for the reason presented below.

The ICVL dataset consists of 2 sequences that with fast abrupt gestures, which often causes tracking to fail. We adopt the maximum allowed error as metric ϵ [24]. All methods except PXC are individual-frame based. PXC is tracking-based, but with an open hand pose detector as a reinitializer, which explains why its curve shows it is particularly good at some poses but performs poorly overall. Sun *et al*., due to its hierarchical structure and cascaded regressor, performs well, but HSO is able to further improve accuracy by about 15% of frames when $\epsilon = 20\text{mm}$.

On the NYU dataset we compare HSO with Tompson *et al*. [25] that is based on Convolutional Neural Networks. Following their setting, the evaluation is done in UV space. Their result contains 14 keypoint locations. To be able to compare fairly, we find a common subset of 12 joint locations to compare (remove palm center and the hamate bone position). The slow hand movements in this dataset favor tracking-based methods like [25]. Different from their approach, we incorporate temporal information hierarchically as described in Section 3.1. As shown in Fig. 7, our method performs better in most cases, and gives sometimes especially high accuracy predictions. However, due to the structured light sensor it uses, the sequences contain relatively high noise and sometimes significantly portion of missing



(a) Comparing with (M=30) and without (M=1) per-joint optimization



(b) Analysis on M and N

Figure 5: Self comparison on the ICVL dataset [22].

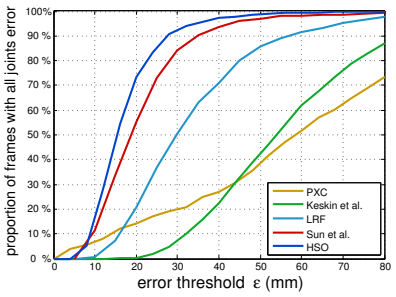
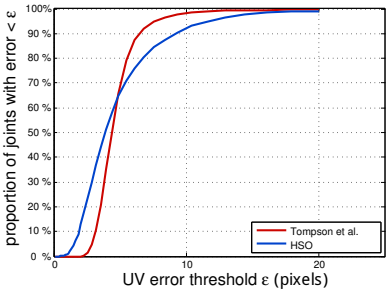
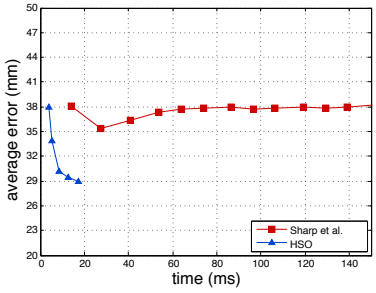
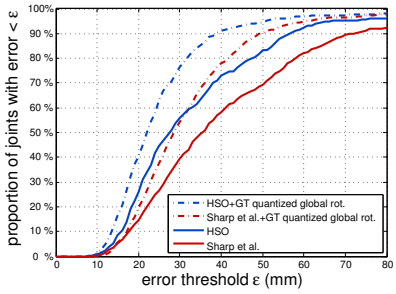


Figure 6: Comparing on the ICVL dataset [22].





(a) Comparing the accuracy and efficiency trade-off with [17]



(b) Comparing with the discriminative part of [17]

Figure 7: Results on NYU dataset [25].

Figure 8: Results on MSHD dataset [17].

pixels. This makes the silver energy fail and degrades our accuracy on the difficult cases.

Finally we compare with [17] on their MSHD dataset. Despite the use of synthetic data, this dataset is particularly challenging, since it exhibits a full range of global rotations. As described in [17], such a diverse data distribution requires very deep decision trees to have satisfying accuracy, which leads to considerable memory consumption. Following their solution, we utilize the same rotation space classifier (obtained from the authors) to quantize the space into 128 clusters, and train an HSO for each. Moreover, on some global rotations, for instance, egocentric view, the hand can be occluded by the forearm. Thus we use an extra step to randomize forearm positions and use the golden energy to choose the best one.

Firstly, conditioned on ground-truth global rotation results, we compare HSO and Sharp *et al.* [17]. In order to vary time cost, we change N of HSO, and the number of PSO generations in [17]. Note that to compare fairly, the starting pose in [17] is discarded, so that both methods do not utilize any temporal information. As a result, PSO finds it very difficult to converge just with particles from their discriminative part. HSO outperforms in both accuracy and efficiency, as shown in Fig. 8(a).

We next replace the PSO part in [17] with the same golden energy selector as in our case, just to compare the discriminative parts. Here we consider two cases: using the groundtruth global rotation cluster, or being probabilistically conditioned on the results of the global rotation classifier, as in [17]. In both cases HSO substantially outperforms [17], as shown in Fig. 8(b). Furthermore, the fact that the curves using the groundtruth global rotation cluster is much better, indicates that their global rotation classifier (random ferns in this case), has hindered the accuracy of the whole pipeline.

In terms of efficiency, from Fig. 5(a) and Fig. 8(a) we can gather that HSO can achieve a satisfying accuracy while running at a speed of 50fps.

Qualitative results are demonstrated in Fig. 9. Despite the quantitative proof of reduction in error accumulation, the 5$^{\text{th}}$ row of MSHD results gives an example of failed palm orientation estimation (groundtruth is facing up whilst prediction is facing down), and inevitably all the rest of the joint predictions are wrong. Also, failed cases with the NYU dataset shows how missing pixels affects the predictions.

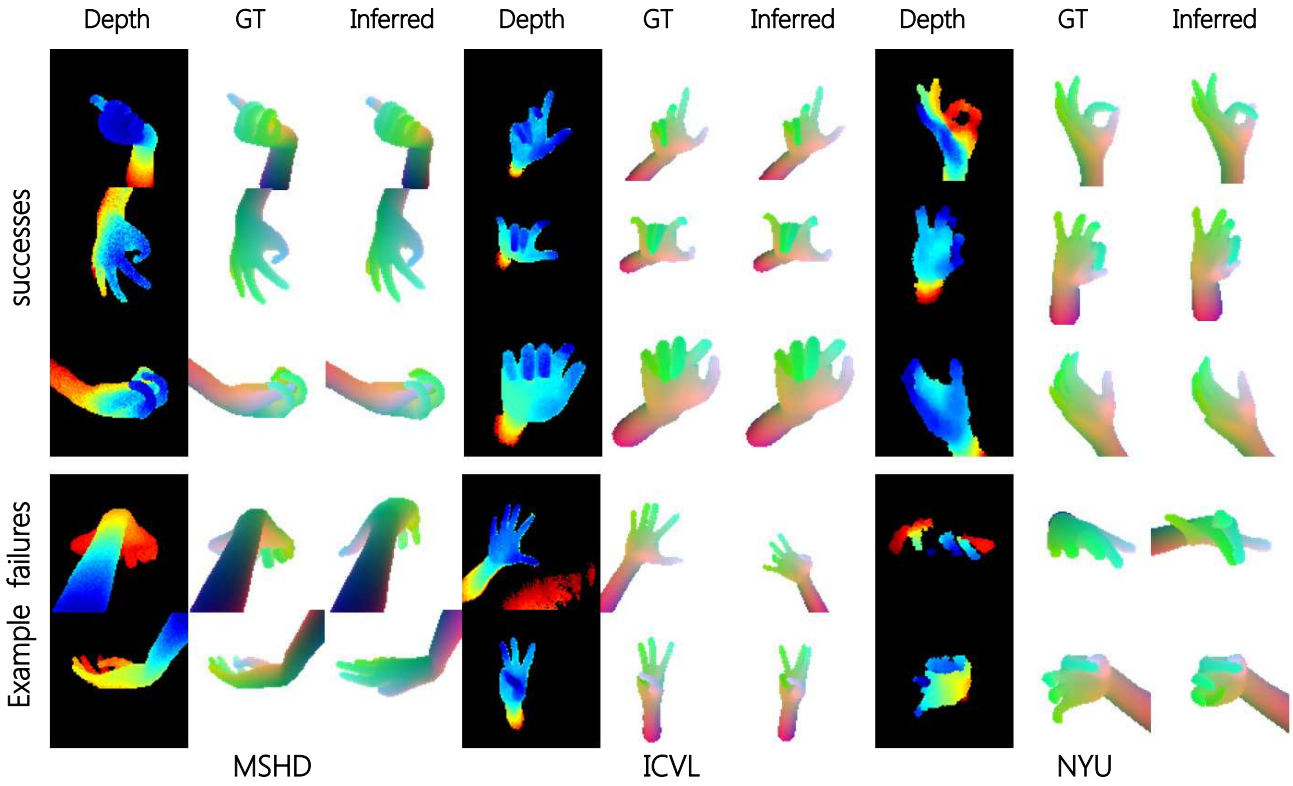


Figure 9: Example successes and failures (due to difficult angle, unclean segmentation and noise, etc.) across three datasets.

5. Conclusion

The golden energy, as introduced by [17], is in many senses the ideal energy to evaluate the fit of a model to input depth data. The idea is simple: just render and compare. Unfortunately, this energy is notoriously hard to optimize, with elusive gradients and many local minima, making traditional black-box optimization techniques extremely brittle and inefficient. Indeed, Sharp *et al*. go to considerable lengths engineering a system at a very low level in order to make the black-box optimization work at real-time, albeit at the expense of complete saturation of a top of the line GPU.

In contrast, our work recognizes the fact that we are solving similar instances of the same problem repeatedly, and thus some knowledge of the problem domain can, and almost surely should be applied to the optimization procedure. In particular, we exploit the fact that subsets of the parameters have subtle relations that can be tested for, via our silver energy, and exploited to drastically reduce the number of hypotheses tested by the more expensive golden energy. This allows us to achieve state of the art results on markedly reduced computational budgets.

Regardless, we recognize the fact that there is more work to be done, and are optimistic by the fact that this method can be so easily modified by replacing the energies and regressors used. For example, although the golden energy is extremely well motivated from a modelling perspective, there continues to be a model mismatch due to inaccuracies in our hand model and non-gaussian sensor noise. As we ultimately want to minimize the prediction of joint angles, which we might call the platinum energy, while being robust to these model defects we might consider actually learning a mapping from the depth image, rendered image pair to the platinum energy value.

In addition, there are many complimentary and promising future directions. From a graphical model point of view, our algorithm utilizes only one kinematic structure, going from the wrist to finger tips, which was decided intuitively. This could be combined with other structures to have more robust results. Also considering contextual information from already predicted joints would encode an implicit collision term, which is absent in the current Markov structure. Moreover, when training the forest, we could try to explicitly encourage the samples from each layer to be diverse, as is done in [5, 10, 15], in order to increase the chances of finding a good sample. Additionally the cascaded regressors in Sun *et al*. [21] can be naturally incorporated into our hierarchical framework and thus might further improve the accuracy.

References

[1] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *CVPR*, 2003. 1

[2] J. Charles, T. Pfister, D. Magee, D. Hogg, and A. Zisserman. Upper body pose estimation with temporal sequential forests. In *BMVC*, 2014. 2

[3] A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer, 2013. 5

[4] M. de La Gorce, D. Fleet, and N. Paragios. Model-based 3D hand pose estimation from monocular video. *TPAMI*, 33(9):1793–1805, 2011. 1

[5] A. Guzman-Rivera, P. Kohli, B. Glocker, J. Shotton, T. Sharp, A. Fitzgibbon, and S. Izadi. Multi-output learning for camera relocalization. In *CVPR*, 2014. 5, 8

[6] Y. Ho, S. Lee, S. Yong, and D. Il. Random tree walk toward instantaneous 3D human pose estimation. In *CVPR*, 2015. 2

[7] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *ECCV*, 2012. 1, 2, 5, 6

[8] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *CVPR*, 2015. 1

[9] P. Krejov, A. Gilbert, and R. Bowden. Combining discriminative and model based approaches for hand pose estimation. In *FG*, 2015. 1

[10] A. Kulesza and B. Taskar. Structured determinantal point processes. In *NIPS*. 2010. 8

[11] M. M. Loper and M. J. Black. OpenDR: An approximate differentiable renderer. In *ECCV*, 2014. 2

[12] S. Melax, L. Keselman, and S. Orsten. Dynamics based 3D skeletal hand tracking. In *i3D*, 2013. 1, 6

[13] I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3D tracking of hand articulations using Kinect. In *BMVC*, 2011. 1, 3, 5

[14] I. Oikonomidis, N. Kyriazis, and A. Argyros. Tracking the articulated motion of two strongly interacting hands. In *CVPR*, 2012. 2

[15] D. Park and D. Ramanan. N-best maximal decoders for part models. In *ICCV*, 2011. 8

[16] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *CVPR*, 2014. 1, 2, 3

[17] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi. Accurate, robust, and flexible real-time hand tracking. In *CHI*, 2015. 1, 2, 3, 4, 5, 6, 7, 8

[18] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011. 5

[19] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In *CVPR*, 2015. 1

[20] B. Stenger, A. Thayananthan, P. H. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *TPAMI*, 28(9):1372–1384, 2006. 1

[21] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *CVPR*, 2015. 1, 2, 6, 8

[22] D. Tang, H.-J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3D hand posture. In *CVPR*, 2014. 2, 6, 7

[23] D. Tang, T.-H. Yu, and T.-K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *ICCV*, 2013. 5

[24] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The Vitruvian Manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, 2012. 3, 6

[25] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *TOG*, 33(5):169, 2014. 1, 2, 5, 6, 7