DOI:10.1145/2398356.2398381

Real-Time Human Pose Recognition in Parts from Single Depth Images

By Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore

Abstract

We propose a new method to quickly and accurately predict human *pose*—the 3D positions of body joints—from a single depth image, without depending on information from preceding frames. Our approach is strongly rooted in current object recognition strategies. By designing an intermediate representation in terms of body parts, the difficult pose estimation problem is transformed into a simpler per-pixel classification problem, for which efficient machine learning techniques exist. By using computer graphics to synthesize a very large dataset of training image pairs, one can train a classifier that estimates body part labels from test images invariant to pose, body shape, clothing, and other irrelevances. Finally, we generate confidence-scored 3D proposals of several body joints by reprojecting the classification result and finding local modes.

The system runs in under 5ms on the Xbox 360. Our evaluation shows high accuracy on both synthetic and real test sets, and investigates the effect of several training parameters. We achieve state-of-the-art accuracy in our comparison with related work and demonstrate improved generalization over exact whole-skeleton nearest neighbor matching.

1. INTRODUCTION

Robust interactive human body tracking has applications including gaming, human–computer interaction, security, telepresence, and health care. Human pose estimation from video has generated a vast literature (surveyed in Moeslund et al.[12] and Poppe[17]). Early work used standard video cameras, but the task has recently been greatly simplified by the introduction of real-time depth cameras.

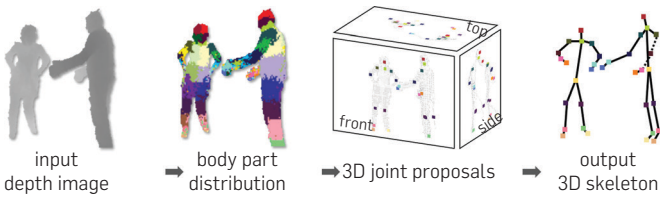
Depth imaging technology has advanced dramatically over the last few years, finally reaching a consumer price point with the launch of Kinect for Xbox 360. Pixels in a depth image record depth in the scene, rather than a measure of intensity or color. The Kinect camera gives a 640 × 480 image at 30 frames per second with depth resolution of a few centimeters. Depth cameras offer several advantages over traditional intensity sensors, which are working in low light levels, giving a calibrated scale estimate, and being color and texture invariant. They also greatly simplify the task of background subtraction, which we assume in this work. Importantly for our approach, it is rather easier to use computer graphics to synthesize realistic depth images of people than to synthesize color images, and thus to build a large training dataset cheaply.

However, even the best existing depth-based systems for human pose estimation[16, 22] still exhibit limitations. In particular, until the launch of Kinect for Xbox 360, of which the algorithm described in this paper is a key component, none ran at interactive rates on consumer hardware while handling a full range of human body shapes and sizes undergoing general body motions.

1.1. Problem overview

The overall problem we wish to solve is stated as follows. The input is a stream of depth images, that is, the image I_t at time t comprises a 2D array of N distance measurements from the camera to the scene. Specifically, an image I encodes a function $d_I(\mathbf{x})$ which maps 2D coordinates $\mathbf{x}$ to the distance to the first opaque surface along the pixel's viewing direction. The output of the system is a stream of 3D *skeletons*, each skeleton being a vector of about 30 numbers representing the body configuration (e.g., joint angles) of each person in the corresponding input image. Denoting the output skeleton(s) at frame t by $\boldsymbol{\theta}_t$, the goal is to define a function F such that $\boldsymbol{\theta}_t = F(I_t, I_{t-1}, \ldots)$. This is a standard formulation, in which the output at time t may depend on information from earlier images as well as from the current image. Our solution, illustrated in Figure 1, pipelines the function F into two

Figure 1. System overview. From a single input depth image, a per-pixel body part distribution is inferred. (Colors indicate the most likely part labels at each pixel and correspond in the joint proposals.) Local modes of this signal are estimated to give high-quality proposals for the 3D locations of body joints, even for multiple users. Finally, the joint proposals are input to skeleton fitting, which outputs the 3D skeleton for each user.



The original version of this paper appeared in the *Proceedings of the 2011 Conference on Computer Vision and Pattern Recognition*, 129–1304.

intermediate representations: a *body parts image* C_t is first computed, which stores a vector of 31 probabilities at every pixel, indicating the likelihood that the world point under that pixel is each of 31 standard body parts. The second intermediate representation is a list of *joint hypotheses* J_t, which contains triples of (body part, 3D position, confidence) hypotheses, with say five hypotheses per body part. Finally, the joint hypotheses are searched for kinematically consistent skeletons. With these intermediate representations, pseudocode for F may be written as the steps

$$C_t = \texttt{ComputeBodyParts}(I_t) \quad (1)$$

$$J_t = \texttt{ComputeJointHypotheses}(C_t, I_t) \quad (2)$$

$$\boldsymbol{\theta}_t = \texttt{FitSkeleton}(J_t, \boldsymbol{\theta}_{t-1}) \quad (3)$$

An important attribute of our solution is that only the final stage uses information from previous frames, so that much of the image interpretation is performed independently on every frame. This greatly enhances the system's ability to recover from tracking errors, which are inimical to almost all existing solutions. In this paper, we focus on the first two stages of the pipeline: the computation of body part labels and joint hypotheses from a single depth image. As we will be dealing with each input frame independently, the t subscripts will be elided in the subsequent exposition.

1.2. Related work

Previous work on the problem has, as noted above, been largely focussed on pose estimation from conventional intensity images (the 2D problem), but of course many of the ideas transfer to the 3D case, and, as we show below, our approach may also be applied to the 2D problem. Thus, we discuss both 2D- and 3D-based work in this section. Many 2D systems make use of a known background and use only the human silhouette as a basis for estimation, thereby gaining invariance to clothing, at the cost of increased ambiguity in the solution.

Of particular interest are systems that perform "one-shot" estimation, from a single image. Such systems are valuable not just as a way of avoiding the error accumulation of tracking-based systems, but also because testing and evaluation of a one-shot system is much simpler than testing a tracking-based solution. Agarwal and Triggs[1] treated pose estimation as a nonlinear regression problem, estimating pose directly from silhouette images. Given a *training set* $\mathcal{T}$ of $(I, \boldsymbol{\theta})$ pairs, and parameterizing the function F (writing it F_Φ), the parameters Φ are found that optimize accuracy on the training set, typically a function of the form $E(\Phi) = \sum_{(I,\boldsymbol{\theta})\in} d(\boldsymbol{\theta}, F_\Phi(I))$ for some accuracy measure $d(\cdot, \cdot)$. This approach forms the essence of the machine learning approach to pose estimation, which is also followed in our work. A large number of papers based on this approach improved the regression models, dealt with occlusion and ambiguity, and re-incorporated temporal information.[13, 25] Such models, however, by classifying whole-body pose in one monolithic function, would require enormous amounts of training data in order to build a fully general purpose tracker as required in the Kinect system. A second difficulty with the regression approach is that the existing methods are quite computationally expensive.

One approach to reducing the demand for training data is to divide the body into parts and attempt to combine the per-part estimates to produce a single pose estimate. Ramanan and Forsyth,[18] for example, find candidate body segments as pairs of parallel lines, clustering appearances across frames. Sigal *et al.*[24] use eigen-appearance template detectors for head, upper arms and lower legs, and nonparametric belief propagation to infer whole body pose. Wang and Popović[26] track a hand clothed in a colored glove. Our system could be seen as automatically inferring the colors of a virtual colored suit from a depth image. However, relatively little work has looked at recognizing parts of the human body. Zhu and Fujimura[28] build heuristic detectors for coarse upper body parts (head, torso, arms) using a linear programming relaxation but require the user to stand in a "T" pose to initialize the model. Most similar to our approach, Plagemann et al.[16] build a 3D mesh to find geodesic extrema interest points, which are classified into three parts: head, hand, and foot. Their method provides both location and orientation estimate of these parts, but does not distinguish left from right and the use of interest points limits the choice of parts.

1.3. Approach

Our approach builds on recent advances in object recognition,[7, 27] which can identify object classes such as "sheep," "building," and "road" in general 2D images. In particular, the use of randomized decision forests allows recognition from a 20-class lexicon to be performed in real time.[20]

The adaptation to the pose estimation problem is relatively straightforward. Starting with a 3D surface model of a generic human body, the surface is divided into 31 distinct body parts (Section 3.1). An object recognition algorithm is trained to recognize these parts, so that at run time, a single input depth image is segmented into a dense probabilistic body parts labeling.

Reprojecting the inferred parts into world space, we localize spatial modes of each part distribution and thus generate (possibly several) confidence-weighted proposals for the 3D locations of each skeletal joint. A combination of these joint locations comprises the output pose $\boldsymbol{\theta}$. Each proposal carries an inferred confidence value, which can be used by any downstream tracking algorithm Eq. (3) for robust initialization and recovery.

We treat the segmentation into body parts as a per-pixel classification task. (In contrast to classification tasks in object recognition or image segmentation, the machinery of Markov Random Fields has not proved necessary in our application.) Evaluating each pixel separately avoids a combinatorial search over the different body joints, although within a single part there are of course still

dramatic differences in the contextual appearance (see Figure 2). For training data, we generate realistic synthetic depth images of humans of many shapes and sizes in highly varied poses sampled from a large motion capture database. The classifier used is a deep randomized decision forest, which is well suited to our multi-class scenario and admits extremely high-speed implementation. The primary challenge imposed by this choice is the need for large amounts of training data, easily obtained given our use of synthetic imagery. The further challenge of building a distributed infrastructure for decision tree training was important to the success of our approach, but is beyond the scope of this paper.

An optimized implementation of our algorithm runs in under 5ms per frame on the Xbox 360 GPU, at least one order of magnitude faster than existing approaches. It works frame by frame across dramatically differing body shapes and sizes, and the learned discriminative approach naturally handles self-occlusions and poses cropped by the image frame. We evaluate both real and synthetic depth images, containing challenging poses of a varied set of subjects. Even without exploiting temporal or kinematic constraints, the 3D joint proposals are both accurate and stable. We investigate the effect of several training parameters and show how very deep trees can still avoid overfitting due to the large training set. Further, results on silhouette images suggest more general applicability of our approach.

1.4. Contributions

Our main contribution is to treat pose estimation as object recognition using a novel intermediate body parts representation designed to spatially localize joints of interest at low computational cost and high accuracy. Our experiments also carry several insights: (i) synthetic depth training data is an excellent proxy for real data; (ii) scaling up the learning problem with varied synthetic data is important for high accuracy; and (iii) our parts-based approach generalizes better than even an oracular whole-image nearest neighbor algorithm.

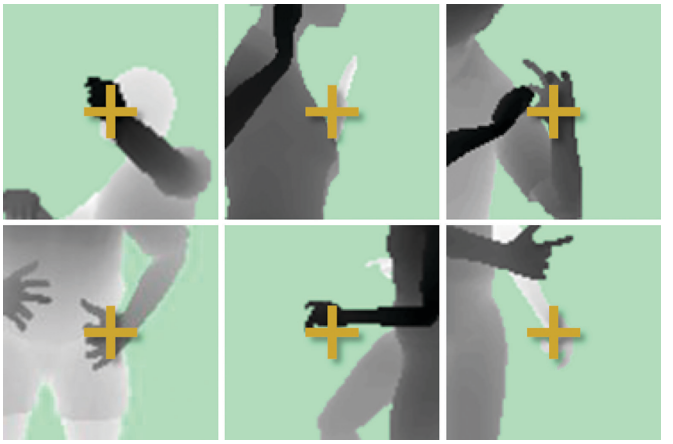
1.5. Sensor characteristics

Before describing our algorithm in detail, we describe the process by which we generate training data for human pose estimation. In order to do so, we first describe the characteristics of the depth sensor we employ, as those characteristics must be replicated in the synthetic data generation.

As described above, the camera produces a 640 × 480 array of depth values, with the following characteristics.

- Certain materials do not reflect infrared wavelengths of light effectively, and so 'drop out' pixels can be common. This particularly affects hair and shiny surfaces.
- In bright sunlight, the ambient infrared can swamp the active signal preventing any depth inference.
- The depth range is limited by the power of the emitter, and safety considerations result in a typical operating range of about 4 m.
- The depth noise level ranges from a few millimeters close up to a few centimeters for more distant pixels.

Figure 2. Example renderings focusing on one hand, showing the range of appearances a single point on the body may exhibit.



- The sensor operates on the principle of stereo matching between an emitter and camera, which must be offset by some baseline. Consequently, an occlusion shadow appears on one side of objects in the depth camera.
- The occluding contours of objects are not precisely delineated and can flicker between foreground and background.

2. TRAINING DATA

Pose estimation research has often focussed on techniques to overcome lack of training data,[13] because of two problems. First, generating realistic intensity images using computer graphics techniques[14, 15, 19] is hampered by the huge color and texture variability induced by clothing, hair, and skin, often meaning that the data is reduced to 2D silhouettes.[1] Although depth cameras significantly reduce this difficulty, considerable variation in body and clothing *shape* remains. The second limitation is that synthetic body pose images are of necessity fed by motion-capture ('mocap') data, which is expensive and time-consuming to obtain. Although techniques exist to simulate human motion (e.g., Sidenbladh et al.[23]), they do not yet produce the range of volitional motions of a human subject.

2.1. Motion capture data

The human body is capable of an enormous range of poses, which are difficult to simulate. Instead, we capture a large database of motion capture of human actions. Our aim was to span the wide variety of poses people would make in an entertainment scenario. The database consists of approximately 500,000 frames in a few hundred sequences including actions such as driving, dancing, kicking, running, and navigating menus.

We expect our semi-local body part classifier to *generalize* somewhat to unseen poses. In particular, we need not record all possible combinations of the different limbs; in practice, a wide range of poses prove sufficient. Further, we need not record mocap with variation in rotation about the

vertical axis, mirroring left–right, scene position, body shape and size, or camera pose, all of which can be added in post-hoc.

Since the classifier uses no temporal information, we are interested only in static *poses* and not motion. Often, changes in pose from one mocap frame to the next are so small as to be insignificant. We thus discard many similar, redundant poses from the initial mocap data using 'furthest neighbor' clustering[10] where the distance between poses $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ is defined as $\max_j \left\| \theta_1^j - \theta_2^j \right\|_2$, the maximum Euclidean distance over body joints j. We use a subset of 100,000 poses such that no two poses are closer than 5cm.

We have found it necessary to iterate the process of motion capture, sampling from our model, training the classifier, and testing joint prediction accuracy in order to refine the mocap database with regions of pose space that had been previously missed out. Our early experiments employed the CMU mocap database,[5] which gave acceptable results though covered far less of pose space.

2.2. Generating synthetic data

We have built a randomized rendering pipeline from which we can sample fully labeled training images. Our goals in building this pipeline were twofold: realism and variety. For the learned model to work well, the samples must closely resemble real camera images and contain good coverage of the appearance variations we hope to recognize at test time. While depth/scale and translation variations are handled explicitly in our features (see below), other invariances cannot be encoded efficiently. Instead, we learn invariances—to camera pose, body pose, and body size and shape—from the data.

The synthesis pipeline first randomly samples a pose from the mocap database, and then uses standard computer graphics techniques to render depth and (see below) body parts images from texture-mapped 3D meshes. The pose is retargeted to each of 15 base meshes (see Figure 3) spanning the range of body shapes and sizes. Further, slight random variation in height and weight gives extra coverage of body shapes. Other randomized parameters include camera pose, camera noise, clothing, and hairstyle. Figure 4 compares the varied output of the pipeline to hand-labeled real camera images.

In detail, the variations are as follows:

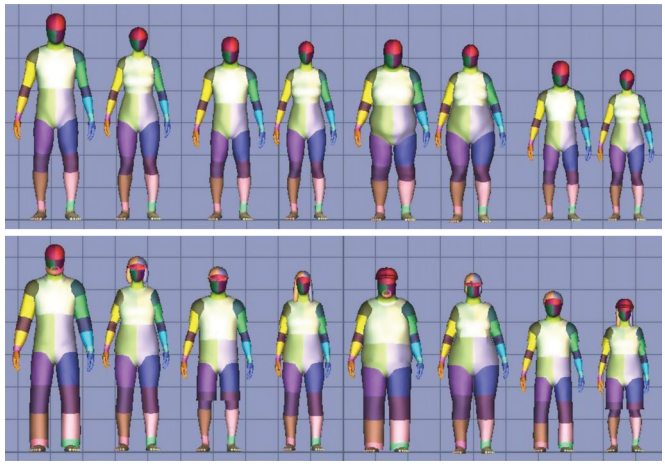
Base character. We use 3D models of 15 varied base characters, both male and female, from child to adult, short to tall, and thin to fat. Some examples are shown in Figure 3 (top row). A given render will pick uniformly at random from the characters.

Pose. Having discarded redundant poses from the mocap data, we retarget the remaining poses to each base character and choose uniformly at random. The pose is also mirrored left–right with probability $\frac{1}{2}$ to prevent a left or right bias.

Rotation and translation. The character is rotated about the vertical axis and translated in the scene, uniformly at random.

Hair and clothing. We add mesh models of several hairstyles and items of clothing chosen at random; some examples are shown in Figure 3 (bottom row).

Figure 3. Renders of several base character models. Top row: bare models. Bottom row: with random addition of hair and clothing.



Weight and height variation. The base characters already have a wide variety of weights and heights. To add further variety, we add an extra variation in height (vertical scale $\pm 10\%$) and weight (overall scale $\pm 10\%$).

Camera position and orientation. The camera height, pitch, and roll are chosen uniformly at random within a range believed to be representative of an entertainment scenario in a home living room.

Camera noise. Real depth cameras exhibit noise. We distort the clean computer graphics renders with dropped out pixels, depth shadows, spot noise, and disparity quantization to match the camera output as closely as possible. In practice however, we found that this noise addition had little effect on accuracy, perhaps due to the quality of the cameras or the more important appearance variations due to other factors such as pose.

We use a standard graphics rendering pipeline to generate the scene, consisting of a depth image paired with its body parts label image. Examples are given in Figure 4.

3. BODY PART INFERENCE AND JOINT PROPOSALS

In this section, we describe our intermediate body parts representation, detail the discriminative depth image features, review decision forests and their application to body part recognition, and finally discuss how a mode finding algorithm is used to generate joint position proposals.

3.1. Body part labeling

A key innovation of this work is the form of our intermediate body parts representation. We define several localized body part labels that densely cover the body, as color-coded in Figure 4. The parts are defined by assigning a label to each triangle of the mesh used for rendering of the synthetic data. Because each model is in vertex-to-vertex correspondence, each triangle is associated with the same part of the body in each rendered image. The precise definitions of the body parts are somewhat arbitrary: the number of parts was chosen at 31 after some initial experimentation with smaller numbers, and it is

Figure 4. Synthetic and real data. Pairs of depth image and ground truth body parts. Note wide variety in pose, shape, clothing, and crop.

convenient to fit the label in 5 bits. The definitions of some of the parts are in terms of particular skeletal joints of interest, for example, 'all triangles intersecting the sphere of radius 10 cm centered on the left hand.' Other parts fill the gaps between these parts. Despite these apparently arbitrary choices, later attempts to optimize the parts distribution have not proved significantly better than the set described in this paper.

For the experiments in this paper, the parts used are named as follows: LU/RU/LW/RW head, neck, L/R shoulder, LU/RU/LW/RW arm, L/R elbow, L/R wrist, L/R hand, LU/RU/LW/RW torso, LU/RU/LW/RW leg, L/R knee, L/R ankle, L/R foot (Left, Right, Upper, lower). Distinct parts for left and right allow the classifier to disambiguate the left and right sides of the body. Even though this distinction may be ambiguous, the probabilistic label we output can usefully use even ambiguous labels.

Of course, the precise definition of these parts could be changed to suit a particular application. For example, in an upper body tracking scenario, all the lower body parts could be merged. Parts should be sufficiently small to accurately localize body joints, but not too numerous as to waste capacity of the classifier.

3.2. Depth image features

We employ simple depth comparison features, inspired by those in Lepetit et al.[11] At a given pixel with 2D coordinates $\mathbf{x}$, the features compute

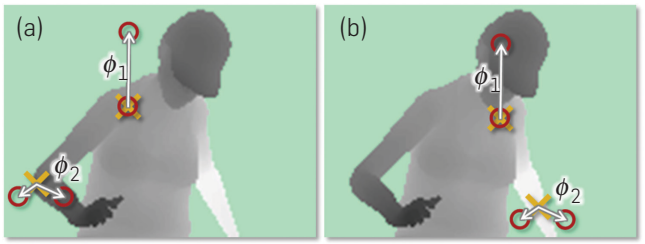
$$f_\phi(I,\mathbf{x}) = d_I\left(\mathbf{x}+\frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x}+\frac{\mathbf{v}}{d_I(\mathbf{x})}\right), \quad (4)$$

where $d_I(\mathbf{x})$ is the depth at pixel $\mathbf{x}$ in image I, and parameters $\phi = (\mathbf{u}, \mathbf{v})$ describe offsets $\mathbf{u}$ and $\mathbf{v}$. The normalization of the offsets by $\frac{1}{d_I(\mathbf{x})}$ ensures that the features are depth invariant: at a given point on the body, a fixed *world space* offset will result whether the pixel is close or far from the camera. The features are thus 3D translation invariant (modulo perspective effects). If an offset pixel lies on the background or outside the bounds of the image, the depth probe $d_I(\mathbf{x}')$ is given a large positive constant value.

Figure 5 illustrates two features at different pixel locations $\mathbf{x}$. Feature f_{ϕ_1} looks upward: Eq. (4) will give a large positive response for pixels $\mathbf{x}$ near the top of the body, but a value close to zero for pixels $\mathbf{x}$ lower down the body.

Feature f_{ϕ_2} may instead help find thin vertical structures such as the arm.

Figure 5. Depth image features. The yellow crosses indicate the pixel x being classified. The red circles indicate the offset pixels as defined in Eq. (4). (a) The two example features give a large depth difference response. (b) The same two features at new image locations give a much smaller response.



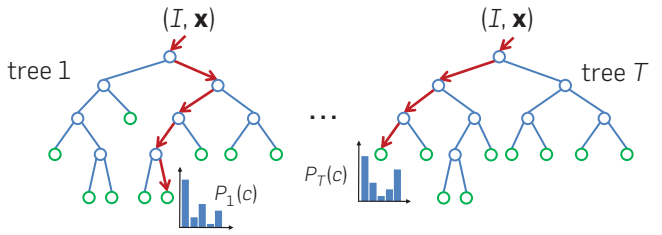
Individually, these features provide only a weak signal about which part of the body the pixel belongs to, but in combination in a decision forest they are sufficient to accurately disambiguate all trained parts. The design of these features was strongly motivated by their computational efficiency: no preprocessing is needed; each feature need read at most three image pixels and perform at most five arithmetic operations; and the features can be straightforwardly implemented on the GPU. Given a larger computational budget, one could employ potentially more powerful features based on, for example, depth integrals over regions, curvature, or local descriptors, for example, shape contexts.[3]

3.3. Randomized decision forests

Randomized decision trees and forests[2, 4] have proven fast and effective multi-class classifiers for many tasks, and can be implemented efficiently on the GPU.[20] As illustrated in Figure 6, a forest is an ensemble of T decision trees, each consisting of split and leaf nodes. Each split node consists of a feature ϕ and a threshold τ. To classify pixel $\mathbf{x}$ in image I, the current node is set to the root, and then Eq. (4) is evaluated. The current node is then updated to the left or right child according to the comparison $f_\phi(I,\mathbf{x}) < \tau$, and the process repeated until a leaf node is reached. At the leaf node reached in tree t, a learned distribution $P_t(c|I,\mathbf{x})$ over body part labels c is stored. The distributions are averaged together for all trees in the forest to give the final classification

$$P(c|I,\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} P_t(c|I,\mathbf{x}). \quad (5)$$

Figure 6. Randomized Decision Forests. A forest is an ensemble of trees. Each tree consists of split nodes (blue) and leaf nodes (green). The red arrows indicate the different paths that might be taken by different trees for a particular input.



Training. Each tree is trained on a different set of randomly synthesized images. A random subset of 2000 example pixels from each image is chosen to ensure a roughly even distribution across body parts. Each tree is trained using the algorithm in Lepetit et al.[11] To keep the training times down, we employ a distributed implementation. Training three trees to depth 20 from 1 million images takes about a day on a 1000 core cluster.

3.4. Joint position proposals

Body part recognition as described above infers per-pixel information. This information must now be pooled across pixels to generate reliable proposals for the positions of 3D skeletal joints. These proposals are the final output of our algorithm and could be used by a tracking algorithm to self-initialize and recover from failure.

A simple option is to accumulate the global 3D centers of probability mass for each part, using the known calibrated depth. However, outlying pixels severely degrade the quality of such a global estimate. We consider two algorithms: a fast algorithm based on simple bottom-up clustering and a more accurate algorithm based on mean shift, which shall now be described.

We employ a local mode-finding approach based on mean shift[6] with a weighted Gaussian kernel. We define a density estimator per body part as

$$f_c(\hat{\mathbf{x}}) \propto \sum_{i=1}^{N} w_{ic} \exp\left(-\left\| \frac{\hat{\mathbf{x}} - \hat{\mathbf{x}}_i}{b_c} \right\|^2 \right), \qquad (6)$$

where $\hat{\mathbf{x}}$ is a coordinate in 3D space, N is the number of image pixels, w_{ic} is a pixel weighting, $\hat{\mathbf{x}}_i$ is the reprojection of image pixel $\mathbf{x}_i$ into world space given depth $d_I(\mathbf{x}_i)$, and b_c is a learned per-part bandwidth. The pixel weighting w_{ic} considers both the inferred body part probability at the pixel and the world surface area of the pixel:

$$w_{ic} = P(c|I, \mathbf{x}_i) \cdot d_I(\mathbf{x}_i)^2. \qquad (7)$$

This ensures that density estimates are depth invariant and give a small but significant improvement in joint prediction accuracy. Depending on the definition of body parts, the posterior $P(c|I, \mathbf{x})$ can be pre-accumulated over a small set of parts. For example, in our experiments the four body parts covering the head are merged to localize the head joint.

Mean shift is used to find modes in this density efficiently. All pixels above a learned probability threshold λ_c are used as starting points for part c. A final confidence estimate is given as a sum of the pixel weights reaching each mode. This proved more reliable than taking the modal density estimate.

The detected modes lie on the *surface* of the body. Each mode is therefore pushed back into the scene by a learned z offset ζ_c to produce a final joint position proposal. This simple, efficient approach works well in practice. The bandwidths b_c, probability threshold λ_c, and surface-to-interior z offset ζ_c are optimized per-part on a hold-out validation set of 5000 images by grid search. (As an indication, this resulted in mean bandwidth 0.065m, probability threshold 0.14, and z offset 0.039m).

4. EXPERIMENTS

In this section, we describe the experiments performed to evaluate our method. We show both qualitative and quantitative results on several challenging datasets and compare with both nearest-neighbor approaches and the state of the art.[8] We provide further results in the supplementary material. Unless otherwise specified, parameters below were set as 3 trees, 20 deep, 300 k training images per tree, 2000 training example pixels per image, 2000 candidate features ϕ, and 50 candidate thresholds τ per feature.

Test data. We use challenging synthetic and real depth images to evaluate our approach. For our synthetic test set, we synthesize 5000 depth images, together with the ground truth body parts labels and joint positions. The original mocap *poses* used to generate these images are held out from the training data. Our real test set consists of 8808 frames of real depth images over 15 different subjects, hand-labeled with dense body parts and seven upper body joint positions. We also evaluate on the real depth data from Ganapathi et al.[8] The results suggest that effects seen on synthetic data are mirrored in the real data, and further that our synthetic test set is by far the 'hardest' due to the extreme variability in pose and body shape. For most experiments, we limit the rotation of the user to ±120° in both training and synthetic test data, since the user faces the camera (0°) in our main entertainment scenario, though we also evaluate the full 360° scenario.

Error metrics. We quantify both classification and joint prediction accuracy. For classification, we report the average per-class accuracy: the average of the diagonal of the confusion matrix between the ground truth part label and the most likely inferred part label. This metric weights each body part equally despite their varying sizes, though mislabelings on the part boundaries reduce the absolute numbers.

For joint proposals, we generate recall-precision curves as a function of confidence threshold. We quantify accuracy as average precision per joint, or mean average precision (mAP) over all joints. The first joint proposal within D meters of the ground truth position is taken as a true positive, while other proposals also within D meters count as false positives. This penalizes multiple spurious detections near the correct position, which might slow a downstream

tracking algorithm. Any joint proposals outside D meters also count as false positives. Note that *all* proposals (not just the most confident) are counted in this metric. Joints invisible in the image are not penalized as false negatives. Although final applications may well require these joints, it is assumed that their prediction is more the task of the sequential tracker Eq. (3). We set $D = 0.1$m below, approximately the accuracy of the hand-labeled real test data ground truth. The strong correlation of classification and joint prediction accuracy (the blue curves in Figures 8(a) and 10(a)) suggests that the trends observed below for one also apply for the other.

4.1. Qualitative results

Figure 7 shows example inferences of our algorithm. Note high accuracy of both classification and joint prediction across large variations in body and camera pose, depth in scene, cropping, and body size and shape (e.g., small child versus heavy adult). The bottom row shows some failure modes of the body part classification. The first example shows a failure to distinguish subtle changes in the depth image such as the crossed arms. Often (as with the second and third failure examples), the most likely body part is incorrect, but there is still sufficient correct probability mass in distribution $P(c|I, \mathbf{x})$ that an accurate proposal can be generated. The fourth example shows a failure to generalize well to an unseen pose, but the confidence gates bad proposals, maintaining high precision at the expense of recall.

Note that no temporal or kinematic constraints (other than those implicit in the training data) are used for any of our results. Despite this, per-frame results on video sequences in the supplementary material show almost every joint accurately predicted with remarkably little jitter.

4.2. Classification accuracy

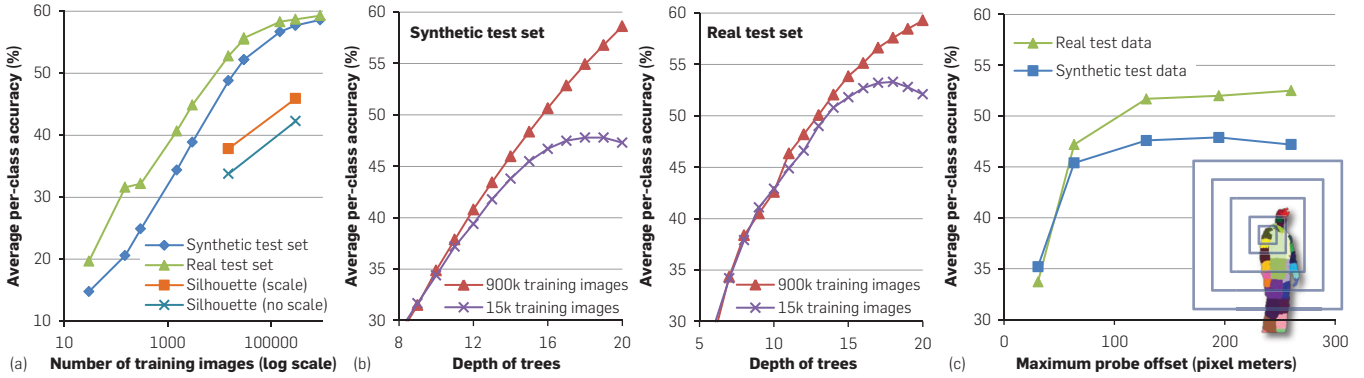
We investigate the effect of several training parameters on classification accuracy. The trends are highly correlated between the synthetic and real test sets, and the real test set appears consistently 'easier' than the synthetic test set, probably due to the less varied poses present.

Number of training images. In Figure 8(a), we show how test accuracy increases approximately logarithmically with the number of randomly generated training images, though starts to tail off around 100,000 images. As shown below, this saturation is likely due to the limited model capacity of a 3 tree, 20 deep decision forest.

Silhouette images. We also show in Figure 8(a) the quality of our approach on synthetic silhouette images, where the features in Eq. (4) are either given scale (as the mean depth) or not (a fixed constant depth). For the corresponding joint prediction using a 2D metric with a 10 pixel true positive threshold,

Figure 7. Example inferences. Synthetic (top row), real (middle), and failure modes (bottom). Left column: ground truth for a neutral pose as a reference. In each example, we see the depth image, the inferred most likely body part labels, and the joint proposals shown as front, right, and top views (overlaid on a depth point cloud). Only the most confident proposal for each joint above a fixed, shared threshold is shown.

Figure 8. Training parameters versus classification accuracy. (a) Number of training images. (b) Depth of trees. (c) Maximum probe offset.



we got 0.539 mAP with scale and 0.465 mAP without. While clearly a harder task due to depth ambiguities, these results suggest the applicability of our approach to other imaging modalities.

Depth of trees. Figure 8(b) shows how the depth of trees affects test accuracy using either 15k or 900k images. Of all the training parameters, depth appears to have the most significant effect as it directly impacts the model capacity of the classifier. Using only 15k images, we observe overfitting beginning around depth 17, but the enlarged 900k training set avoids this. The high accuracy gradient at depth 20 suggests that even better results can be achieved by training still deeper trees, at a small extra run-time computational cost and a large extra memory penalty. Of practical interest is that, until about depth 10, the training set size matters little, suggesting an efficient training strategy.

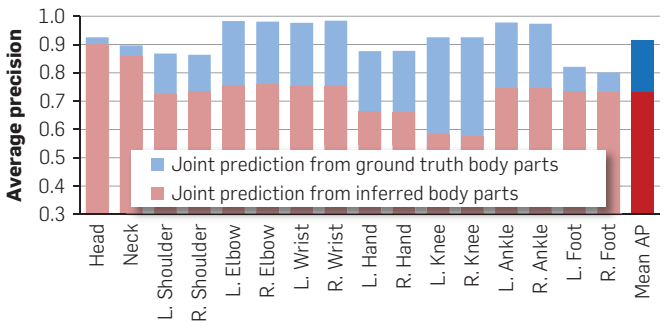
Maximum probe offset. The range of depth probe offsets allowed during training has a large effect on accuracy. We show this in Figure 8(c) for 5k training images, where 'maximum probe offset' means the max. absolute value proposed for both x and y coordinates of $\mathbf{u}$ and $\mathbf{v}$ in Eq. (4). The concentric boxes on the right show the five tested maximum offsets calibrated for a left shoulder pixel in that image; the largest offset covers almost all the body. (Recall that this maximum offset scales with world depth of the pixel.) As the maximum probe offset is increased, the classifier is able to use more spatial context to make its decisions, though without enough data it would eventually risk overfitting to this context. Accuracy increases with the maximum probe offset, though levels off around 129 pixel meters.

4.3. Joint prediction accuracy

In Figure 9, we show average precision results on the synthetic test set, achieving 0.731 mAP for the mean-shift clustering algorithm and 0.677 mAP using the fast clustering algorithm. Combined with body part classification, the fast clustering runs in under 5 ms on the Xbox GPU, while mean shift takes 20 ms on a modern 8 core desktop CPU.

In order to get an idea of the maximum achievable mAP, we compare the mean shift algorithm to an idealized setup that is given the *ground truth* body part labels. On the real test set, we have ground truth labels for head, shoulders, elbows, and hands. An mAP of 0.984 is achieved on those parts given the ground truth body part labels, while 0.914 mAP is achieved using the inferred body parts. As expected, these numbers are considerably higher on this easier test set. While we do pay a small penalty for using our intermediate body parts representation, for many joints the inferred results are both highly accurate and close to this upper bound.

Figure 9. Joint prediction accuracy. We compare the actual performance of our system (red) with the best achievable result (blue), given the ground truth body part labels.



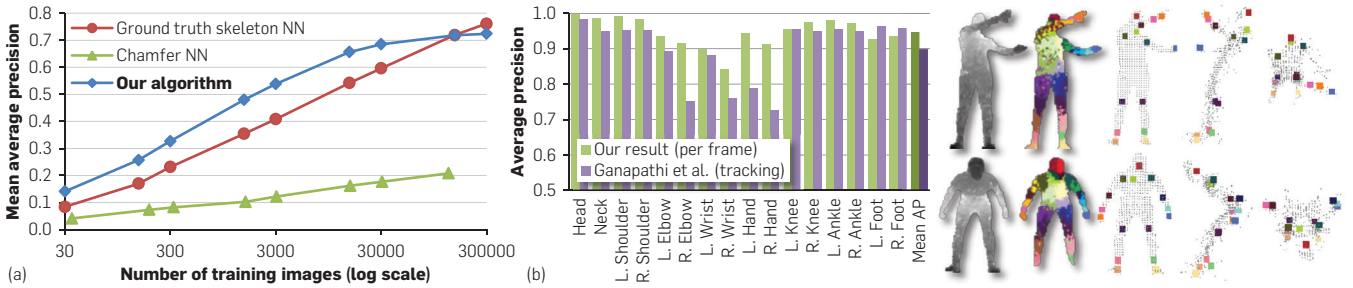
Comparison with nearest neighbor. To highlight the need to treat pose recognition in *parts*, and to calibrate the difficulty of our test set for the reader, we compare with two variants of exact nearest-neighbor whole-body matching in Figure 10(a). The first, idealized, variant matches the ground truth *test skeleton* to a set of training exemplar skeletons with optimal rigid translational alignment in 3D world space. Of course, in practice one has no access to the test skeleton. As an example of a realizable system, the second variant uses chamfer matching[9] to compare the test image to the training exemplars. This is computed using depth edges and 12 orientation bins. To make the chamfer task easier, we throw out any cropped training or test images. We aligned images using the 3D center of mass and found that further local rigid translation only reduced accuracy.

Our algorithm, recognizing in parts, generalizes better than even the idealized skeleton matching until about 150k training images are reached. As noted above, our results may get even better with deeper trees, but already we robustly infer 3D body joint positions and cope naturally with cropping and translation. The speed of nearest-neighbor chamfer matching is also considerably slower (2 fps) than our algorithm. While hierarchical matching[9] is faster, one would still need a massive exemplar set to achieve comparable accuracy.

Comparison with Ganapathi et al.[8] Ganapathi et al. provided their test data and results for direct comparison. Their algorithm uses body part proposals from Plagemann et al.[16] and further tracks the skeleton with kinematic and temporal information. Their data comes from a time-of-flight depth camera with rather different noise characteristics to our structured light sensor. Without any changes to our training data or algorithm, Figure 10(b) shows considerably improved joint prediction average precision. Our algorithm also runs at least 10× faster.

Full rotations and multiple people. To evaluate the full 360° rotation scenario, we trained a forest on 900k images containing full rotations and tested on 5k synthetic full rotation images (with held out poses). Despite the massive increase in left–right ambiguity, our system was still able to achieve an mAP of 0.655, indicating that our classifier can accurately learn the subtle visual cues that distinguish front and back facing poses. Residual left–right uncertainty after classification can naturally be propagated to a tracking algorithm through multiple hypotheses. Our approach can propose joint positions for multiple people in the image, since the per-pixel classifier generalizes well even without explicit training for this scenario.

Figure 10. Comparisons. (a) Comparison with nearest neighbor matching. (b) Comparison with Ganapathi et al.[8] Even without the kinematic and temporal constraints exploited by Ganapathi et al.,[8] our algorithm is able to more accurately localize body joints.



5. DISCUSSION

We have seen how accurate proposals for the 3D locations of body joints can be estimated in super real-time from single depth images. We introduced body part recognition as an intermediate representation for human pose estimation. Use of a highly varied synthetic training set allowed us to train very deep decision forests using simple depth-invariant features without overfitting, learning invariance to both pose and shape. Detecting modes in a density function gives the final set of confidence-weighted 3D joint proposals. Our results show high correlation between real and synthetic data, and between the intermediate classification and the final joint proposal accuracy. We have highlighted the importance of breaking the whole skeleton into parts, and show state-of-the-art accuracy on a competitive test set.

As future work, we plan further study of the variability in the source mocap data, the properties of the generative model underlying the synthesis pipeline, and the particular part definitions. Whether a similarly efficient approach can directly regress joint positions is also an open question. Perhaps a global estimate of latent variables such as coarse person orientation could be used to condition the body part inference and remove ambiguities in local pose estimates.

Acknowledgments

We thank the many skilled engineers in Xbox, particularly Robert Craig, Matt Bronder, Craig Peeper, Momin Al-Ghosien, and Ryan Geiss, who built the Kinect tracking system on top of this research. We also thank John Winn, Duncan Robertson, Antonio Criminisi, Shahram Izadi, Ollie Williams, and Mihai Budiu for help and valuable discussions, and Varun Ganapathi and Christian Plagemann for providing their test data. C

References

1. Agarwal, A., Triggs, B. 3D human pose from silhouettes by relevance vector regression. In *Proceedings of CVPR* (2004).
2. Amit, Y., Geman, D. Shape quantization and recognition with randomized trees. *Neural Computation, 9*, 7 (1997), 1545–1588.
3. Belongie, S., Malik, J., Puzicha, J. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI 24*, 4 (2002), 509–522.
4. Breiman, L. Random forests. *Mach. Learn. 45*, 1 (2001), 5–32.
5. CMU Mocap Database. http://mocap.cs.cmu.edu.
6. Comaniciu, D., Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. PAMI 24*, 5 (2002).
7. Fergus, R., Perona, P., Zisserman, A. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of CVPR* (2003).
8. Ganapathi, V., Plagemann, C., Koller, D., Thrun, S. Real time motion capture using a single time-of-flight camera. In *Proceedings of CVPR* (2010).
9. Gavrila, D. Pedestrian detection from a moving vehicle. In *Proceedings of ECCV* (June 2000).
10. Gonzalez, T. Clustering to minimize the maximum intercluster distance. *Theor. Comp. Sci. 38* (1985).
11. Lepetit, V., Lagger, P., Fua, P. Randomized trees for real-time keypoint recognition. In *Proceedings of CVPR* (2005).
12. Moeslund, T., Hilton, A., Krüger, V. A survey of advances in vision-based human motion capture and analysis. *CVIU 104*(2–3) (2006), 90–126.
13. Navaratnam, R., Fitzgibbon, A.W., Cipolla, R. The joint manifold model for semi-supervised multi-valued regression. In *Proceedings of ICCV* (2007).
14. Ning, H., Xu, W., Gong, Y., Huang, T.S. Discriminative learning of visual words for 3D human pose estimation. In *Proceedings of CVPR* (2008).
15. Okada, R., Soatto, S. Relevant feature selection for human pose estimation and localization in cluttered images. In *Proceedings of ECCV* (2008).
16. Plagemann, C., Ganapathi, V., Koller, D., Thrun, S. Real-time identification and localization of body parts from depth images. In *Proceedings of ICRA* (2010).
17. Poppe, R. Vision-based human motion analysis: An overview. *CVIU 108*(1–2) (2007), 4–18.
18. Ramanan, D., Forsyth, D. Finding and tracking people from the bottom up. In *Proceedings of CVPR* (2003).
19. Shakhnarovich, G., Viola, P., Darrell, T. Fast pose estimation with parameter sensitive hashing. In *Proceedings of ICCV* (2003).
20. Sharp, T. Implementing decision trees and forests on a GPU. In *Proceedings of ECCV* (2008).
21. Shotton, J., Winn, J., Rother, C., Criminisi, A. *TextonBoost*: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proceedings of ECCV* (2006).
22. Siddiqui, M., Medioni, G. Human pose estimation from a single view point, real-time range sensor. In *IEEE International Workshop on Computer Vision for Computer Games* (2010).
23. Sidenbladh, H., Black, M., Sigal, L. Implicit probabilistic models of human motion for synthesis and tracking. In *Proceedings of ECCV* (2002).
24. Sigal, L., Bhatia, S., Roth, S., Black, M., Isard, M. Tracking loose-limbed people. In *Proceedings of CVPR* (2004).
25. Urtasun, R., Darrell, T. Local probabilistic regression for activity-independent human pose inference. In *Proceedings of CVPR* (2008).
26. Wang, R., Popović, J. Real-time hand-tracking with a color glove. In *Proceedings of ACM SIGGRAPH* (2009).
27. Winn, J., Shotton, J. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Proceedings of CVPR* (2006).
28. Zhu, Y., Fujimura, K. Constrained optimization for human pose estimation from depth sequences. In *Proceedings of ACCV* (2007).

Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, Andrew Blake, and Mat Cook ({jamiesho, tsharp, awf, ablake, and a-macook}@microsoft.com), Microsoft Research, Cambridge, UK.

Alex Kipman and Mark Finocchio ({akipman and markfi}@microsoft.com), Xbox Incubation.

Richard Moore ST-Ericsson.