Three Dimensional Pose Estimation of Mouse from Monocular Images in Compact Systems

Ghadi Salem, Jonathan Krynitsky, Thomas Pohida
Center for Information Technology
National Institutes of Health
Bethesda, Maryland 20892
Email: ghadi.salem@nih.gov

Monson Hayes
Electrical and Computer Engineering
George Mason University
Fairfax, VA 22030

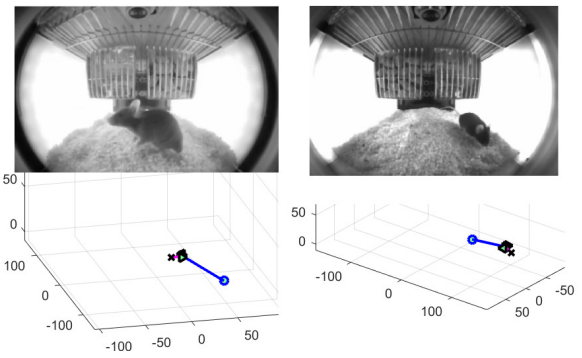
Xavier Burgos-Artizzu
Transmural Biotech
Barcelona, Spain

***Abstract*—Video-based activity and behavior analysis for mice has garnered wide attention in biomedical research. Animal facilities hold large numbers of mice housed in 'home-cages' densely stored within ventilated racks. Automated analysis of mice activity in their home-cages can provide a new set of sensitive measures for detecting abnormalities and time-resolved deviation from baseline behavior. Large scale monitoring in animal facilities requires minimal footprint hardware that integrates seamlessly with the ventilated racks. Compactness of hardware imposes use of fisheye lenses positioned in close proximity to the cage. In this paper, we estimate the 3D pose of a mouse from fisheye distorted monocular monochromatic images using a novel adaptation of a structured forests algorithm. The method utilizes classification decision trees leveraging their versatility to store arbitrary information in the leaf-nodes. During training, the samples arriving at each node are mapped from continuous pose space to discrete class labels such that similar poses are grouped in the same class. The node splitting function is trained by optimizing a classification objective function rather than a high-dimensional regression one. The leaf-nodes store the pose parameters for the set of samples reaching the node. A prediction model preserving the structural relationship of the pose is formed based on the samples in the leaf-nodes. We apply the method to what we believe is the first known training set for 3D recovery of mouse key points from monocular images. We compare the results of our approach to those obtained via standard regression techniques.**

I. Introduction

Mice are the most widely used mammalian model in biomedical research laboratories. It is common for animal facilities to hold in excess of $100,000$ mice in cages housed within ventilated racks. A recent trend in laboratory mice research is to capture activity and behavior in the naturalistic environment for long durations. Automated monitoring of activity can improve safety, reduce labor costs, and provide a new set of sensitive time-resolved biomarkers for quantifying the well-being of mice. The scientific demand for video-based automation of research-animal monitoring has led to development of several commercial and academic systems [1]. An impediment for large scale use, however, has been the lack of compact hardware design compatible with existing ventilated racks. Recently, a system was designed for integration with ventilated racks is that described by Salem et al. [2]. This system does not utilize an overhead camera as it would not meet mechanical constraints imposed by racks as noted by many [3]. Rather, cameras fitted with fisheye lenses positioned very close (i.e., 5mm) to the front and rear of the cage. The resulting images exhibit a large degree of nonlinearity due to the lens distortion exacerbated by very close proximity of the lens placement to the cage wall, as shown in Figure 1. This burden on downstream video analysis algorithms is further intensified by the high deformability of mice. We describe a novel method that extracts 3D pose of the mouse from the challenging images acquired using the hardware described in [2]. Rather than treating the output pose parameters as independent and estimating each parameter separately, as would be the case in standard single-variate regression, our method exploits the correlation between pose parameters

(a)



(b)

Figure 1. (a) The video acquisition system used for this work is designed to integrate into vivarium cage racks. The cameras are fitted with fisheye lenses and are in very close proximity to the cage. (b) Example images and the corresponding 3D pose estimated by our algorithm.

inherent in the training set. Our method is based on Dollár and Zitnick's [4] innovative approach to structured output space predictions. We adapt their method, which was applied to edge detection, to estimate continuous pose parameters. The method exploits pose structure present in the training set by mapping each set of pose parameters to a binary string. The encoded binary string is then further discretized into classes encoding the interdependencies between the parameters. Node splitting functions are trained based on the discrete labels, rather than the continuous parameters, employing standard classification optimization objectives. Leaf nodes store the whole set of poses reaching them or a representative subset. An aggregation model is then designed to generate predictions from the set of stored poses in a leaf. An ensemble model combines the predictions from multiple trees into a single 'best' prediction or, if desired, multiple pose proposals. Generating pose proposals is advantageous as it allows multiple hypotheses suited for the multimodal output pose space. The proposals can in turn seed algorithms aimed at merging poses by exploiting temporal consistency or further refinement via other regression methods. The main contributions of this work are:

1) A novel efficient structured pose estimation method applied to decision forests. The method assigns discrete class labels to continuous pose parameters grouping similar poses in the same class. Mapping the set of pose parameters (as opposed to an individual parameter) to discrete class labels has two benefits. First, the discretization accounts for the correlation between pose parameters, and hence the pose structure is maintained. Second, the discretization renders the training problem solvable by optimizing a classification objective (e.g., standard information criteria gain) as opposed to a regression objective function.
2) Evaluation of method on a unique and challenging dataset (made publicly available: scorhe.nih.gov) acquired via a specialized hardware setup designed to be suitable for wide-scale use in animal facilities. The dataset is, to our knowledge, the first known annotation set allowing recovery of mouse 3D pose from monocular images. The set should be of interest to vision researchers as, uncommon to many datasets, the target object lacks visible articulation and is highly deformable. The dataset should also be of interest to those involved in video-based activity and behavior analysis of mice, a topic that has lately received a lot of attention [1].

II. Related Work

Pose estimation continues to be an active field of research. A lot of the literature focuses on human pose estimation and has been reviewed in [5] among others. More recent approaches [6] estimate 3D position of human body joints from single depth images. Several works describe pose estimation for mice. One defining component of pose estimation is the pose model (i.e., pose parameters). An often used coarse pose model is the ellipse [7], [8]. Oriented ellipses (i.e., with one end of the ellipse aligned with the anterior of the mouse) are used in [9], [10]. Branson and Belongie [3] define an ellipse as a coarse pose which then guides detection for more refined pose based on twelve manually constructed deformable contour templates that are assumed to be representative of the mouse postures. de Chaumont et al. [11] define a physics engine based 2D articulated rigid body model consisting of head, belly, and neck linked with joints constraining relative motions between the connected bodies. In all described methods, pose parameters are defined in image domain. Translation to physical domain is done by fixed scaling as scaled orthography is assumed. For fisheye lens distorted images with the added complexity of large appearance variations due to proximity of lens to monitoring arena, defining pose in image domain might be uninformative. We instead define pose parameters as the 3D physical coordinates of the mouse key points. We adapt structured forests described in [4] to estimate the 3D pose parameters from monocular images.

III. Method

We first review standard decision forests, then outline the extension by Dollár and Zitnick [4] to structured output spaces, after which we describe our adaptation of the method to structured pose estimation.

A. Standard decision forests

A quick review of decision forests helps set terminology and mathematical notation. Many references, such as [12] can be consulted for a more thorough treatment of the subject. For consistency, the review herein closely follows the work on which we are expanding [4]. For a training set $\mathcal{S} : \{\mathcal{X}, \mathcal{Y}\}$, a binary decision tree $f_t(x), x \in \mathcal{X}$ is a partition of the input feature space into (typically, but not necessarily) axis aligned cuboid regions, with each region storing an assigned output prediction $\hat{y}$. The prediction is $\hat{y} = g(\{y_k\})$ where $\{y_k\} \in \mathcal{Y}$ are the output labels falling in the cuboid regions having input feature vectors $\{x_k\} \in \mathcal{X}$. Each node in the tree defining a partition is referred to as a *split* or *internal* node, whereas the terminal node representing the region containing the prediction $\hat{y}$ is referred to as *leaf* node. The partitioning is done in a recursive manner. Starting with the *root* node, a *split function* partitions the training data into two sets. One set would be associated with the right node, and the other with the left. The split function for node j can be written mathematically as $h(x, \theta_j) \in \{0, 1\}$ where traversal is to the left if the output is 0 and otherwise is to the right. Typically, the parameters θ of the split function are simply (k, τ), an index of a feature element within the vector x, and a threshold, such that $h(x, \theta) = [x(k) < \tau]$, where $[\cdot]$ is the indicator function. The process is repeated until criteria on the information content of the node or the depth of the tree is met, thereby reaching a leaf node, wherein a prediction $g(\cdot)$ is stored. A decision forest is a collection of T trees $f_t(x), t \in \{1 \dots T\}$, each trained differently by injecting randomness in the choice of features used for split functions and/or the choice of subset

of the whole training set $\mathcal{S}$. Training a decision forest entails training each tree and then combining the resulting predictions by an ensemble model. The ensemble model can be stated mathematically as $\hat{y}_F = l(\hat{y}_t), t \in \{1 \dots T\}$ where the $\hat{y}_t$'s are the leaf node predictions from all trees in the forest. The main tasks in training a forest are training each individual tree and defining a suitable prediction aggregation model, i.e. defining $l(\cdot)$. Training each tree entails finding an optimal node split for each internal node and defining a prediction model for each leaf node, i.e., a suitable $\hat{y}$. The split node function parameters θ_j are obtained by maximizing an information gain criterion. The standard information gain relies on defining an information measure function H. For classification, it is common to define H as the Shannon entropy, Gini impurity, or the twoing criterion. For single-variate regression, a common H is one that minimizes variance of the child nodes.

B. Structured decision forests

Realizing the challenge of defining an information gain criterion for structured output spaces, Dollár and Zitnick [4] map the output space to discrete labels. The mapping would be defined such that similar structured labels are assigned the same discrete label. The information gain is then computed for the discrete classes rather than the structured labels themselves using the standard information gain criteria along with standard H used in classification training. The mapping is used to train the split functions at each internal node. The structured labels arriving at each node, however, are stored and propagated until a leaf node is reached. A prediction model for each leaf node, and an ensemble model for the whole forest would then be formulated based on the stored structured labels. The success of this elegant solution is contingent upon finding an effective mapping from structured labels space to discrete labels, i.e., $\mathcal{Y} \rightarrow \mathcal{C}$, such that each label $y \in \mathcal{Y}$ is mapped to a discrete label $c \in \mathcal{C}$, where $\mathcal{C} = \{1, \dots, k\}$. Given that measuring similarity for structured labels might not be well defined, [4] employed an intermediate mapping $\Pi : \mathcal{Y} \rightarrow \mathcal{Z}$ such that $\mathcal{Z}$ is a space on which similarity can be measured by computing Euclidean distance. The authors then applied their method to predicting edge maps in an image (i.e., classifying a pixel either as an edge pixel or not).

C. Structured pose-estimation

We leverage the framework developed by [4] and adapt it to a regression problem. Namely we estimate the 3D coordinates of key points for the mouse. Since the physical objects of interest have properties that constrain the relative location of key points, the pose output space is structured. The proposed method capitalizes on this underlying structure. In the subsections that follow, we detail the key components for setting up and training structured forests to accomplish the pose-estimation task. Namely, we define

- The structured output space $\mathcal{Y}$
- The mapping function Π.
- The discretization mapping $\mathcal{Z} \rightarrow \mathcal{C}$
- The leaf-node prediction model
- The ensemble model
- The input feature space $\mathcal{X}$

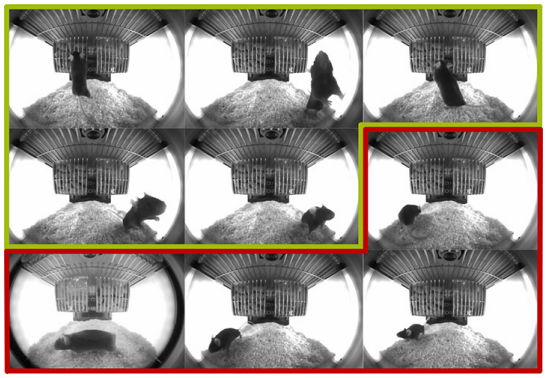


Figure 2. Example output of the discretization function. The function was applied to nine randomly chosen poses. The green and red outlines group the images for which the poses received the same class label by the node splitting function. The results are intuitive as all the poses in which the mouse nose was at greater elevation are grouped together.

1) Structured Output Pose Space: The output pose space is $\mathcal{Y} \in \mathbb{R}^D$, such that $y \in \mathcal{Y}$ is a D-dimensional vector containing the pose parameters. We start by defining a 3D Cartesian coordinate system with the origin at the center of the cage floor. The 3D coordinates of the landmarks in the said coordinate system are expressed as $\phi = \{\phi_i\}, i \in \{1, \dots, 4\}$, where the sequence of i's indexes tail, left-ear, right-ear, and nose respectively and each ϕ_i has the 3D coordinate of key point i. In order to generate a more translation-tolerant representation of ϕ suitable for use as y in the training framework, we define y as follows:

$$y = [\phi_1 - \mathcal{M}(\sigma_a), \phi_i - \phi_1], \qquad i \in \{2, \cdots, 4\} \qquad (1)$$

where $\mathcal{M}$ is a pre-assigned image-to-physical coordinate mapping that maps each pixel location σ in the image to a fixed but arbitrarily chosen 3D location. σ_a is then taken to be a point (e.g., binary silhouette ellipse-fit end point) on the detected foreground. Referencing the tail point to a fixed 3D point based on the detection window can be thought of as the 3D analog of referencing 2D landmarks relative to the corner point of the 2D detection window. The pose representation in Eq 1 achieves tolerance to translation by representing the key points relative to the tail point and the tail point relative to a 3D mapped position based on the 2D detection window.

2) Intermediate mapping function: One key component of the structured forest approach is the mapping Π from the structured labels to an intermediate space on which dissimilarity can be measured. The mapping is obtained by first converting each parameter $y(d), d \in 1, \dots, D$ to a binary sequence denoted $b_{L_d}(y(d))$, where L_d is the number of bits representing parameter $y(d)$. To generate the binary sequence for each $y(d)$, the full range of the parameter in the N training samples arriving at the node is computed as $r(d) = \max_i y_i(d) - \min_i y_i(d), i \in 1, \dots, N$. The range $r(d)$ is uniformly partitioned into L_d bins. For $y_i(d)$, Bit l in $b_{L_d}(y(d))$ is set if $y_i(d)$ falls in bin l. Second the binary

representations for each parameter are concatenated to form the initial z vector. Mathematically, the mapping generating z is then defined as $\Pi = \bigvee_{d=1}^{D} b_{L_d}(y(d))$, where $\bigvee$ denotes concatenation. Lastly, we further reduce dimensionality of z by Principal Component Analysis (PCA). PCA is used to limit the dimensions to at most 5 principal components. The reduced dimensionality also serves to denoise z. Two methods are used to assign the number of bits L_d to which parameter $y(d)$ is converted. The first method, hereinafter referred to as 'fixed', assigns to each parameter the same fixed number of bits b_L, i.e. $b_{L_d} = b_L \forall d$. The second method, hereinafter referred to as 'weighted', assigns more bits to parameters having larger ranges $r(\cdot)$. Namely, $b_{L_d} = B \times r(d) / \sum_{i=1}^{D} r(i)$, where B is the total number of bit, namely $B = b_L \times D$. b_L is a training parameter hereinafter referred to as 'number of bins'.

3) Discretization mapping: The goal of the intermediate mapping function is to allow for discrete label assignments to the structured output labels, such that similar y's $\in \mathcal{Y}$ are assigned the same class label $c \in \mathcal{C}$, where $\mathcal{C} = \{1, \ldots, m\}$. We follow the same two approaches as [4] to obtain a discrete label assignments for the set of y's at each node. The first approach is clustering z's into m clusters by K-means. Each y is then assigned the cluster number in which its corresponding z falls. The second approach is to take the top $\log_2(m)$ principal components resulting for the earlier computed PCA. Each y is then assigned the a discrete label according to the orthant into which z falls. It is noted that the intermediate mapping and subsequent discrete label assignments are done for each node separately based on the subset of the training set arriving at the node. Discretization captures interdependencies between the pose parameters. Figure 2 shows the output of the discretizing function on a sample set of poses.

4) Leaf node prediction model: To completely specify a model for a single tree, a leaf node prediction model has to be specified. Decision trees are versatile in that arbitrarily complex information can be stored in their leaf nodes. In the case of structured output trees, the leaf nodes can store a single 'best' prediction from the output labels arriving at the node. In [4], the leaf node prediction is set to be the label y_m whose z_m is the medoid of all z_m's at the node. Although we experimented with different approaches, we chose to retain the same leaf-node prediction model as in [4]. Hence, our leaf-node prediction is the pose whose z_m is the medoid.

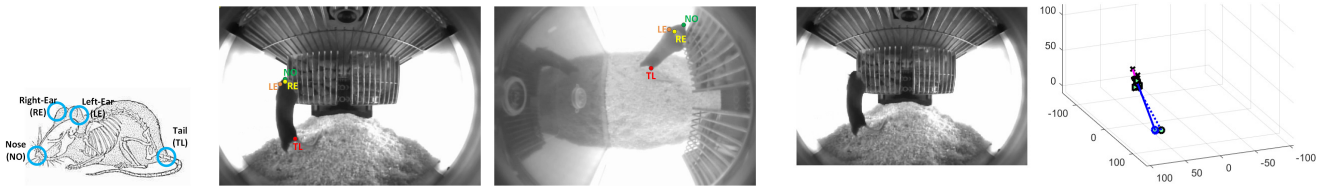
5) Ensemble model: An ensemble model is defined to aggregate the leaf node predictions. Here again, the result could be the single 'best' pose or a set of pose proposals (i.e., the set or subset of poses returned by all trees in the forest). Pose proposals are advantageous as they would account for multi-modality in the pose output space and can be useful if a temporal consistency model is applied to filter out implausible poses. Alternatively multiple proposals can be used to initialize another refinement method. In this work, we take advantage of the constrained environment and the available calibration mappings to rule out some improbable poses prior to selecting the 'best' pose. Namely, we project all the poses back onto the image and eliminate poses that fall out of range of the mouse detection window. We then chose a single pose by applying the same discretization mapping described in section III-C2 to the set of retained poses. Similar to the leaf-node prediction model, the 'best' pose is that which has the medoid binary string. Our method results in a prediction $\hat{y}$ which maintains the pose structure in the training data. Although we observed that ensemble models based on choosing the median of each parameter from the set of poses returned by the trees reduce the failure rates, it is noted that these models do not guarantee a physically plausible pose estimate.

6) Input space: Pose estimation is typically preceded by object detection. The detector could supply a window (e.g., bounding box) or a more refined segmentation map. In our case, we manually annotate mouse pixels in ~ 250 images for each camera. The annotations provide positive and negative samples to train a classifier to produce a segmentation mask. The features used for pose-estimation are derived both from the binary silhouette and the intensity image, more specifically the foreground bounding box region of the image. Two different types of features are extracted. Hence,e the feature vector $x = [x_1, x_2]$ is the concatenation of the two feature sets. The first feature set x_1 is the binary silhouette statistics. The statistics include area, image coordinates of the bounding box, ellipse fit major and minor axis lengths, ellipse fit orientation, image coordinates of major and minor axis end points, binary silhouette eccentricity, and major axis to minor axis ratio.

The second feature set, x_2, is the raw intensity values at randomly pre-selected positions relative a normalized (i.e. width=1, length=1) foreground bounding box. To compute K such features, a set of offsets $(o_x^k, o_y^k), k = 1, \cdots, K, o \in [0, 1]$ are randomly generated. Each normalized position (o_x^k, o_y^k) is scaled to the actual size of the detected bounding box to yield an image pixel position $\sigma^k = (o_x^k \cdot b_w, o_y^k \cdot b_h)$ relative to the upper left corner of the bounding box, where b_w and b_h are the bounding box width and height respectively. The k^{th} entry in x_2 is the intensity value of the image at pixel location σ^k. Our implementation uses $K = 125$ such features.

IV. Dataset and Implementation Detail

The dataset used to evaluate the structured pose estimation method is that of mouse key point annotations. Pose parameters are based on four key-points on the mouse as shown in Figure 3a: tail (TL), left-ear (LE), right-ear (RE), and nose (NO). The video acquisition hardware was augmented with two additional overhead cameras, one at each end of the cage. These cameras are only used to acquire video for the training sets. Normal operation of the hardware unit does not utilize the overhead cameras as their placement is precluded by cage and rack mechanical constraints. The video acquisition for top and side cameras is synchronized. All cameras are calibrated so as to enable mapping image coordinates to physical space and vice versa. The key points (i.e., TL, LE, RE, NO) are marked in both orthogonal views as shown in Figure 3b. In situations where simple geometry can be used to compute position of some key-points given others, only a sufficient subset is annotated. Namely, if the left and right parts of the



(a) (b) (c)

Figure 3. (a) Skeletal structure for mouse labeled with key points defined for this work. (b) Example manual annotation of the key points from top and side camera. (c) Ground truth stick figure (solid) for the given annotations, and predicted pose stick figure (dashed)

mouse are symmetric along the straight line connecting tail (TL) to nose (NO), then only TL, NO, and the more visible of the two ears are annotated. Approximately 92,000 frames were annotated to account for the large variation in appearance due to the hardware configuration and mouse posture. A set of 6,500 frames was annotated by two different trained annotators to establish a meaningful measure for assessing estimation quality. The first step in the estimation pipeline is to produce a segmentation mask using the trained classifier described in III-C6. Two sets of features are then extracted from the binary silhouette and its corresponding bounding box in the intensity image. The first set is the binary silhouette statistics for the detected foreground (e.g., bounding box, centroid, area, ellipse fit parameters). The second is derived from intensity features as described in III-C6. The computed feature vectors along with the corresponding ground-truth pose parameters as defined in (1) are used to train the proposed structured forest. The leaf nodes store the single 'best' pose from the set arriving at the node. At runtime, the image is segmented, features are computed in the same fashion as was done in training, and the trees are traversed. Each tree contributes a pose estimate. A single 'best' pose is then selected from the ensemble. One implementation variation that yielded higher prediction accuracy was to utilize the calibration mappings and project the whole set of poses returned from all the trees back on the image. Only poses projecting to within empirically chosen tolerance of the binary silhouette of the mouse are retained. The single 'best' pose is then selected from the filtered set of poses. It is noted that the system used in [2] utilizes two cameras to overcome the cage-lid obstruction. However, only the image from the camera to which the mouse is closer (as determined by the area of the detected foreground) is used for training and prediction. Hence, the 3D construction is done from monocular images.

V. Results

We assessed our approach on two separate sequences totaling 1,100 frames that were not used in training. The redundant annotations are used to define a distance metric as proposed in [9] which equalizes the error from each pose parameter by weighing it with the inverse of its variance in the redundant annotations. The distance measure between two poses y^1 and y^2 is

$$d(y^1, y^2) = \sqrt{\frac{1}{12}\sum_{i=1}^{12}\frac{1}{\sigma_i}(y^1(i) - y^2(i))^2} \qquad (2)$$

where 12 is the number of parameters (e.g., 3 per each of the 4 keypoints) in the pose representation 1. The σ_i's in (2) refer to the variance in each pose parameter between two human annotators for the whole set of redundantly annotated frames. Following [9], we define a normalized distance threshold d_{thr} for a successful estimate. d_{thr} is set to be such that the normalized distance for 99% of the redundantly annotated frames fall below d_{thr}. So if the estimation output has a normalized distance (i.e., that computed via equation (2)) exceeding d_{thr}, then it is considered a failed estimate. The value of d_{thr} was computed to be 3.77. We compared our approach to the standard regression forest where all the parameters are treated as uncorrelated and estimated separately. As shown in Figure 4, our approach reduces failure rates by 2%. Furthermore, our prediction maintains pose structure whereas there is no guarantee that the pose prediction from the standard regression forest is a physically plausible pose. While a failure rate of 35% seems high, we note that the seemingly easier task of

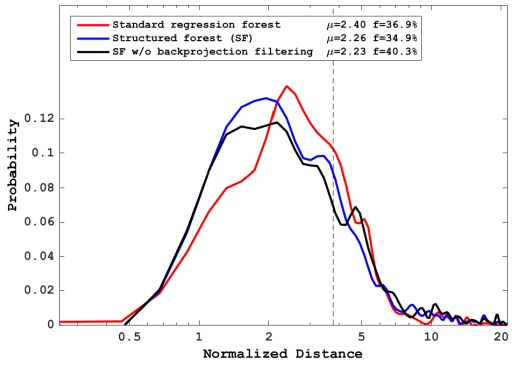


Figure 4. Probability distribution for pose distance as defined in 2. The vertical line denotes the threshold for what is deemed to be a failure in estimation. The reported means μ for distance error is computed for cases where estimation was deemed successful. The best performance was obtained with our structured forest approach with filtering out poses that do not project back onto the image within prespecified proximity of the detected foreground. For the same number of trees, the structured forest implementation is superior to the standard regression forest.

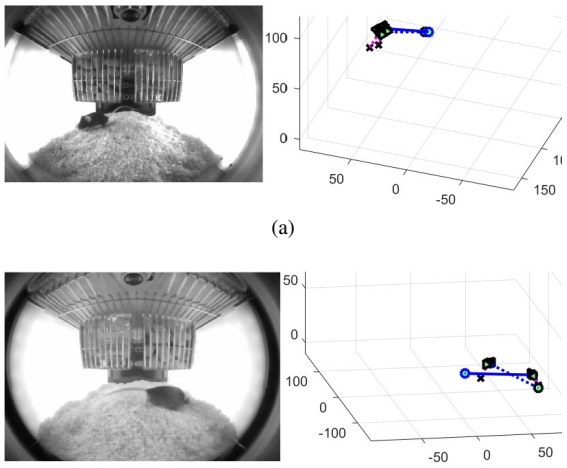


Figure 5. Ground truth (solid) versus estimate (dashed). (a) An example of a successful prediction (i.e. distance error lower than threshold computed by assessing annotator consistency). (b) An example of a failed prediction. The main error is due to 180 degree rotation. The lack of distinct image features to properly orient the pose is evident in the example.

Table I
RESULTS OF ALGORITHM TRAINING PARAMETERS SWEEPS.

	Value	% fail	d mean
No of Bins Fixed	3	37.6	3.31
	4	37.3	3.34
	5	38.4	3.58
No of Bins Weighted	3	36.4	3.31
	4	39.3	3.54
	5	34.9	3.41
No of Trees	4	44.7	3.40
	8	37.7	3.26
	16	34.9	3.41
Fraction of training set	0.4	40.9	3.61
	0.5	39.8	3.44
	0.9	38.0	3.51

estimating 2D pose from monocular images of mice [9], [10] using state of the art method reported similar results. The high deformability of the mouse and lack of visible features make it difficult to discriminate slight pose differences in many cases as shown in Figure 5.

We have also investigated the effect of training parameters on the accuracy of the estimations. Table I shows a subset of parameter sweeps that were done to validate the approach. We observed that filtering out pose-proposals that do not project to close proximity to the image detection window consistently reduced failure rates by 5-7%. Excluding segmentation, the frame rate for pose estimation with backprojection filtering is 60 fps.

VI. CONCLUSION

We've presented a novel solution for the practical problem of estimating mouse pose in video systems well-suited for scalable use in animal vivaria. Our solution leverages a unique and challenging dataset of mouse key points. We presented a novel approach to structured pose estimation and applied the approach to the dataset. We were able to accurately construct 3D pose estimates from monocular images of mice. We benchmarked our approach against standard regression forest techniques of predicting pose assuming no correlation between the parameters. Our evaluations indicate this new approach offers many advantages compared to standard regression forest methods. First, we showed that our approach reduces failure rates. Second, the predictions preserve the pose structure observed in the training sets. Third, the approach yields a set of valid pose proposals rather than a single prediction. These proposals can seed other pose refinement algorithms. One drawback however, is that the method would be unable to predict a pose sufficiently different from any encountered in training. Therefore, as shown in the results section, a rich training set is required for the method to work as described. The dataset is available online (scorhe.nih.gov).

ACKNOWLEDGMENT

GS wishes to thank Piotr Dollár for guidance and input.

REFERENCES

[1] A. I. Dell, J. A. Bender, K. Branson, I. D. Couzin, G. G. de Polavieja, L. P. Noldus, A. Perez-Escudero, P. Perona, A. D. Straw, M. Wikelski, and U. Brose, "Automated image-based tracking and its application in ecology," *Trends in Ecology and Evolution*, vol. 29, no. 7, pp. 417 – 428, 2014.

[2] G. H. Salem, J. U. Dennis, J. Krynitsky, M. Garmendia-Cedillos, K. Swaroop, J. D. Malley, S. Pajevic, L. Abuhatzira, M. Bustin, J.-P. Gillet *et al.*, "Scorhe: A novel and practical approach to video monitoring of laboratory mice housed in vivarium cage racks," *Behavior research methods*, vol. 47, no. 1, pp. 235–250, 2015.

[3] K. Branson and S. Belongie, "Tracking multiple mouse contours (without too many samples)," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 1039–1046 vol. 1.

[4] P. Dollar and C. Zitnick, "Fast edge detection using structured forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 8, pp. 1558–1570, Aug 2015.

[5] R. Poppe, "Vision-based human motion analysis: An overview," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 4–18, 2007.

[6] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman *et al.*, "Efficient human pose estimation from single depth images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 12, pp. 2821–2840, 2013.

[7] H. Pistori, V. V. V. A. Odakura, J. B. O. Monteiro, W. N. Gonçalves, A. R. Roel, J. de Andrade Silva, and B. B. Machado, "Mice and larvae tracking using a particle filter with an auto-adjustable observation model," *Pattern Recognition Letters*, vol. 31, no. 4, pp. 337–346, 2010.

[8] K. Branson, V. Rabaud, and S. J. Belongie, "Three brown mice: See how they run," in *VS-PETS Workshop at ICCV*, 2003.

[9] P. Dollar, P. Welinder, and P. Perona, "Cascaded pose regression," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 1078–1085.

[10] X. P. Burgos-Artizzu, D. C. Hall, P. Perona, and P. Dollár, "Merging pose estimates across space and time," in *BMVC*, 2013.

[11] F. de Chaumont, R. D.-S. Coura, P. Serreau, A. Cressant, J. Chabout, S. Granon, and J.-C. Olivo-Marin, "Computerized video analysis of social interactions in mice," *Nature Methods*, vol. 9, no. 4, pp. 410–417, 2012.

[12] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Foundations and Trends® in Computer Graphics and Vision*, vol. 7, no. 2–3, pp. 81–227, 2012.