Parameterizing Object Detectors in the Continuous Pose Space

Kun He[1], Leonid Sigal[2], and Stan Sclaroff[1]
hekun@cs.bu.edu, lsigal@disneyresearch.com, sclaroff@cs.bu.edu

[1] Computer Science Department, Boston University
[2] Disney Research Pittsburgh

Abstract. Object detection and pose estimation are interdependent problems in computer vision. Many past works decouple these problems, either by discretizing the continuous pose and training pose-specific object detectors, or by building pose estimators on top of detector outputs. In this paper, we propose a structured kernel machine approach to treat object detection and pose estimation jointly in a mutually benificial way. In our formulation, a unified, continuously parameterized, discriminative appearance model is learned over the entire pose space. We propose a cascaded discrete-continuous algorithm for efficient inference, and give effective online constraint generation strategies for learning our model using structural SVMs. On three standard benchmarks, our method performs better than, or on par with, state-of-the-art methods in the combined task of object detection and pose estimation.

Keywords: object detection, continuous pose estimation

1 Introduction

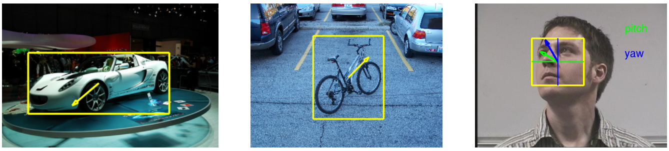


Fig. 1: Three examples of joint object detection and pose estimation.

We focus on the combined problems of object detection and pose estimation. Given an image x containing some object, we seek to localize the object in x, while estimating its pose at the same time. We can encode the prediction output as $y = (B, \theta)$, where B is a structured output indicating the object's location, and θ is a real-valued vector indicating the object's pose. Fig. 1 shows three examples; in these examples, B is a rectangular bounding box for the detected object, while θ gives 1D or 2D angles specifying the object's orientation.

Object detection and pose estimation are interdependent problems, and it is challenging to simultaneously infer both the object's location and pose in uncontrolled images. Many past works have broken this problem into two stages to simplify the situation. Some approaches, *e.g.* [1–3], discretize the pose space and then learn pose-specific detectors. While having considerable success, the complexity of such approaches scales with the granularity of the discretization of the pose space, and generalization to continuous pose estimation is difficult. On the other hand, regression methods, *e.g.* [4–6], produce continuous pose estimates given detection results as input. However, the outputs from actual object detectors in practice are often not optimized for successive stages, resulting in suboptimal pose estimation performance.

In contrast, we argue that object detection and pose estimation should be solved jointly in a mutually beneficial way. We also argue that object pose should not be treated as a discrete variable: the continuous pose space usually has a smooth underlying structure that can be lost in discretization. We thus solve the combined problem within a unified structured prediction framework, simultaneously estimating the object's location and pose.

We take a kernel machine approach, where localization and pose are jointly modeled using a product of two kernels: a structural kernel for localization, and a pose kernel for continuous parameterization. In order to solve the associated nonconvex inference problem, we devise a cascaded inference algorithm that efficiently generates diverse proposals to explore the search space. For learning our model using the structural SVM, we propose a mini-batch online learning algorithm with simple but effective constraint generation strategies, which significantly decreases training time. To summarize, our contributions are:

1. We formulate object detection and continuous pose estimation jointly as a structured prediction problem. Our method learns a *single, continuously parameterized*, object appearance model over the entire pose space.
2. We design a cascaded discrete-continuous inference algorithm to effectively optimize a nonconvex objective involving a complicated search space.
3. We give an online mini-batch constraint generation strategy that can significantly speed up the training of structural SVMs.

In experiments with three standard benchmarks in the combined task of object detection and pose estimation, our method performs better than, or on par with, state-of-the-art methods that are typically more complicated.

2 Related Work

Multi-view object detection is an extensively studied problem. Representative works include [1, 7, 8]. These works predominantly treat the object pose or viewpoint estimation problem as a multiclass classification problem, by discretizing the viewsphere and learning view-specific object detectors.

More recently, in light of the success of the Deformable Part Model (DPM) [9] in generic object detection, view-based DPM mixture models have been introduced [2, 3, 10] to train a collection of view-specific DPMs, or even provide limited 3D reasoning [8, 11–13]. The learning of different view-specific DPMs in such works is loosely coupled by either implicit latent variable assignments [2, 3] or explicit part sharing [10–12]. However, all these models inherently deal with discretized views, and strongly coupling part appearance across views is difficult (*e.g.* via expensive part selection mechanisms). While both [11] and [12] claim to learn object models on a continuous viewsphere (in [12] leveraging high-quality 3D CAD models), both works in practice resort to fine discretizations of the pose space. In contrast, our approach continuously models the pose space by means of continuous parameterization of object detectors, and tightly couples the learning across the entire space.

For the standalone problem of object pose estimation, various regression methods [4–6], also including methods relying on statistical manifold modeling [14, 15], have been proposed. While these methods are able to perform continuous pose estimation, they assume that the object localization is given. In practice, clean object foreground masks are hard to come by, and outputs from actual object detectors are rarely optimized for the subsequent regression stage. This mismatch ultimately degrades the pose estimation performance of regression methods. In contrast, we avoid this mismatch by learning a unified model that performs detection and pose estimation jointly.

A closely related work to ours is Yuan *et al.* [16], who learn “parameter sensitive detectors” for binary classification. Pose parameterization in [16] is achieved by multiplying a pose kernel $K_\theta(\theta, \theta')$ with the original kernel, and inference is performed by discretizing the pose and testing pose-specific classifiers. While we also use a multiplicative decomposition of the joint kernel, we formulate the problem in the structured-output domain, and produce continuous solutions for pose during inference.

Ionescu *et al.* [17] propose a structural SVM approach for joint object localization and continuous state estimation, also using a multiplicative joint kernel. Inference is performed in an alternating fashion where localization is initialized using a generic object detector. However, single initialization is suboptimal for our nonconvex objective; instead, our cascaded inference efficiently generates diverse initializations to better explore the search space. Also, we propose novel online constraint generation strategies for structural SVMs. The resulting online algorithm significantly outperforms the cutting plane method used in [17].

3 Mathematical Formulation

Suppose we are given a training set of n pairs $\{(x_i, y_i)\}_{i=1}^n$ where each example $x_i \in \mathcal{X}$ is a training image, and each label y_i belongs to a structured output space $\mathcal{Y}$. We focus on predicting bounding boxes and viewing angles, and take the structured output space $\mathcal{Y}$ to be $\Re^4 \times [0, 2\pi)^d$, where $d \leq 3$ is the number of angles (a complete parameterization of the viewsphere needs 3 angles).

Our goal is to learn a scoring function $f : \mathcal{X} \times \mathcal{Y} \to \Re$ such that the label y assigned to x maximizes $f(x, y)$. We parameterize f as $f(x, y) = \langle \mathbf{w}, \Psi(x, y) \rangle$, where $\mathbf{w}$ is a parameter vector, and $\Psi(x, y)$ is a *joint feature map*. The inner product $\langle \cdot, \cdot \rangle$ is defined in a reproducing kernel Hilbert space (RKHS), instantiated by a *joint kernel function* K: $K(x, y, x', y') = \langle \Psi(x, y), \Psi(x', y') \rangle$.

We intend to learn the scoring function f via regularized loss minimization. Assuming that the conditions of the Representer Theorem [18] are met, f has the following implicit representation, where $\mathcal{V}$ is the index set of "support vectors" $\{(x_j, y_j)\}$, and α_j are scalars:

$$f(x,y) = \langle \mathbf{w}, \Psi(x,y) \rangle = \sum_{j \in \mathcal{V}} \alpha_j K(x, y, x_j, y_j). \tag{1}$$

It is key to design the joint kernel K for our task. Given two input-output pairs $(x, (B, \theta))$ and $(x', (B', \theta'))$, we define K to be the product of two valid Mercer kernels K_s and K_p:

$$K(x, y, x', y') = K_s(\phi(x, B), \phi(x', B')) \cdot K_p(\theta, \theta'). \tag{2}$$

Here, $\phi(x, B)$ represents the feature vector extracted from the image region inside bounding box B in image x. The structural kernel K_s measures the similarity between two such feature vectors. The pose kernel K_p measures the similarity between poses θ and θ'. The joint kernel achieves a high value only for input-output pairs with similar inputs and similar outputs.

In this work, we are interested in the case where K_s is linear, *i.e.* $K_s(\phi, \phi') = \phi^T \phi'$, for efficiency considerations. On the other hand, in order for K_p to smoothly capture the complex effects of varying pose, we choose to use a nonlinear RBF kernel: $K_p(\theta, \theta') = \exp\left(-\gamma\, d(\theta, \theta')^2\right)$ where $d(\theta, \theta')$ is a distance measure, *e.g.* Euclidean distance or geodesic distance. Then, given image x and model $\mathbf{w}$, the inference problem in our model becomes:

$$\max_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(x,y) \rangle = \max_{(B,\theta) \in \mathcal{Y}} \sum_{j \in \mathcal{V}} \alpha_j \phi(x, B)^T \phi(x_i, B_j) \exp\left(-\gamma\, d(\theta, \theta_j)^2\right). \tag{3}$$

This is a complicated nonconvex optimization problem. In the next section, we will describe a cascaded solution to optimizing Eq.(3).

4 Cascaded Inference

Solving Eq.(3) is difficult: there are a large number of bounding boxes B in an image, and the objective is nonconvex in θ whenever there is a negative α_j. However, if either B or θ is fixed, the problem becomes significantly simplified and better studied: given B, θ can be estimated by regression; given θ, B can be obtained by a θ-specific detector. Both can be seen as extreme cases of a general cascaded scheme: first prune the search space, and then refine the answer.

Inspired by the reasoning above, we also propose to use a two-step cascade consisting of a pruning step and a refining step. However, we shall take the middle ground and keep multiple candidates for both B and θ after the first step, in order to avoid problems associated with the two extreme approaches, as discussed in Section 2.

Specifically, the pruning step returns a reduced search space $\tilde{\mathcal{Y}} = \{(B_k, \Theta_k)\}_{k=1}^K$, where each pair consists of a candidate bounding box B_k and an associated range Θ_k of plausible poses, *e.g.* $\Theta_k = \{\theta \,|\, d(\theta, \theta_k) < \delta\}$ for some θ_k and δ. The refining step performs further optimization and returns a final solution pair (B^*, θ^*).

4.1 Refining step

Barring a somewhat unconventional ordering, we shall first study the problem of refinement, in order to highlight desired properties for the pruning step. Assume for now that $\tilde{\mathcal{Y}}$ is given, and for $\forall k \in \{1, \ldots, K\}, \forall j \in \mathcal{V}$, denote $\eta_k^j = \alpha_j \phi(x_j, B_k)^T \phi(x, B_j)$. Then, the problem of refinement can be cast as:

$$\max_k \max_{\theta \in \Theta_k} \sum_{j \in \mathcal{V}} \eta_k^j \exp\left(-\gamma\, d(\theta, \theta_j)^2\right). \tag{4}$$

We employ a gradient ascent approach for optimizing Eq.(4) with respect to θ, as the use of a smoothly differentiable pose kernel permits the use of gradient algorithms, *e.g.* L-BFGS. For each B_k, we optimize over $\theta \in \Theta_k$ to find its matching pose θ_k, and finally maximize over k to pick the best-scoring pair (B^*, θ^*).

However, as Eq.(4) still is nonconvex in general, Θ_k's should be restricted in size so that there are few local optima. Also, the candidate bounding boxes $\{B_k\}$ should contain diverse elements so as to explore the search space. Lastly, K necessarily needs to be small since continuous optimization is a relatively costly operation. To summarize, the pruning step should:

- efficiently generate diverse B_k's to explore the solution space,
- produce small Θ_k's to reduce the number of local optima, and
- produce a small K to reduce the number of continuous refinements.

4.2 Pruning step

Now we are ready to propose our pruning strategy: uniformly divide the θ space into M intervals $\{\Theta_1, \ldots, \Theta_M\}$, specify a "seed pose" $\{\theta_1, \ldots, \theta_M\}$ within each interval (*e.g.* the geometric centers), and sample pose-specific detectors from the model to generate proposals for B.

Since our method learns a continuously parameterized object appearance model Eq.(1) over the entire pose space, we can efficiently sample pose-specific detectors from the unified model. For any fixed θ, our model reduces to a single linear classifier $\mathbf{w}_\theta$ (this observation is also made by [16]):

$$\mathbf{w}_\theta = \sum_{j \in \mathcal{V}} \alpha_j \exp\left(-\gamma\, d(\theta, \theta_j)^2\right) \phi(x_j, B_j), \quad m = 1, \ldots, M. \tag{5}$$

Our strategy satisfies all the desired properties: we can control the size of the intervals to limit the number of local optima within them; we can leverage existing techniques to efficiently evaluate the classifiers and generate bounding box proposals; and it is easy to control the number of generated proposals.

Given detectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_M\}$, we reuse existing techniques, for example sliding windows, to generate diverse bounding box proposals. Then, the refining step performs gradient-based continuous optimization using the seed poses as starting points. If the intervals are relatively small, single stage refinement is sufficient (we observe this to be the case in practice); coarse sampling can be handled by cascades where the solution is refined in a hierarchical fashion: first refining Θ to a smaller interval, constructing a better tuned detector specific to Θ, refining localization using this new detector, etc.

We emphasize that the pose-specific detectors are sampled from the original unified model by fixing θ; thus, their scores are directly comparable, since the scores are essentially given by the unified model. This contrasts with many 1-vs-all methods, where the classifier scores for different classes may be uncalibrated.

4.3 Generating Diverse Proposals: Branch-and-Bound

With the M pose-specific detectors sampled from our model, bouding box proposals can be generated typically in time that is linear in M. We further speed this up via a branch-and-bound algorithm that is capable of generating diverse proposals in sublinear time. Our algorithm generalizes the Efficient Subwindow Search (ESS) [19] by Lampert *et al.* and operates on an augmented state space, encoding both object location and pose.

The input to our algorithm are the pose-specific detectors $\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M\}$ sampled from our unified model. Sets of candidate solutions, or *states*, are sorted in a priority queue Q according to a merit function indicating their promisingness. The algorithm iterates by splitting the most promising state in Q and inserting the resulting states back into Q, until it gets a singleton state. The amortized time complexity for branch-and-bound is on the order of $O(\log_2 M)$.

State representation: A state is parameterized as $s = (w, h, x_0, x_1, y_0, y_1, \theta)$, where w and h encode the bounding box size, x_0 and x_1 bound the x coordinate of its upper-left corner, and y_0 and y_1 bound the y coordinate. θ is one of $\{\theta_1, \ldots, \theta_M\}$. For each combination of the first six parameters, there are M unique states containing different values of θ.

Bounding classifier scores: If state s contains pose θ_m, then the corresponding classifier $\mathbf{w}_m$ is used to generate bounds for the set of bounding boxes contained in s. Once $\mathbf{w}_m$ is chosen, the scenario reduces to that of bounding the score of a single linear classifier over a set of rectangular regions. In this case, Lampert *et al.* [19] showed that tight bounds can be constructed for Bag-of-Words (BOW) features. We refer interested readers to [19] for the full derivation of the bounding techniques.

Diverse solutions: Instead of terminating the algorithm after obtaining the first singleton state, we store the state in an output buffer and continue the algorithm, until K singleton states have been returned (obtaining top-K solutions) or until the score is below a threshold (obtaining all "good" solutions). Non-maximum suppression is used to enforce diversity.

5 Learning

For learning our unified model for joint object detection and continuous pose estimation within a kernel machine framework, we use the structural SVM [20] to learn a large-margin model $\mathbf{w}$. The structural SVM is formulated as:

$$\min_{\mathbf{w},\xi} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \tag{6}$$

$$s.t. \ \ \langle \mathbf{w}, \Psi(x_i, y_i)\rangle - \langle \mathbf{w}, \Psi(x_i, \bar{y}_i)\rangle \geq \Delta(y_i, \bar{y}_i) - \xi_i, \ \ \forall i, \forall \bar{y}_i \in \mathcal{Y}, \tag{7}$$

where the loss term $\Delta(y_i, \bar{y}_i)$ encodes the penalty of predicting $\bar{y}_i$ instead of y_i. To jointly handle object localization and pose, we use a combined loss:

$$\Delta(y_i, y) = \Delta((B_i, \theta_i), (B, \theta)) = \beta\Delta_{loc}(B_i, B) + (1-\beta)\Delta_{pose}(\theta_i, \theta). \tag{8}$$

The localization loss is based on bounding box overlap [21]: $\Delta_{loc}(B, B') = 1 - \frac{Area(B \cap B')}{Area(B \cup B')}$, and pose loss is proportional to angular difference: $\Delta_{pose} \propto \angle(\theta, \theta')$.

The structural SVM usually is solved by constraint generation algorithms such as the cutting plane algorithm [20] or its one-slack reformulation [22]. To find a violated constraint for training pair (x_i, y_i), the following loss-augmented inference problem is solved:

$$\bar{y}_i = \arg\max_{y \in \mathcal{Y}} \ \Delta(y_i, y) + \langle \mathbf{w}, \Psi(x_i, y)\rangle \tag{9}$$

$$\approx \arg\max_{B, \theta_m} \ \beta\Delta_{loc}(B_i, B) + (1-\beta)\Delta_{pose}(\theta_i, \theta_m) + \mathbf{w}_m^T\phi(x, B). \tag{10}$$

Eq.(9) has the same structure as the test-time inference problem Eq.(3) and can also be solved by the two-step cascade. The pruning step Eq.(10) samples seed poses $\{\theta_m | m = 1, \ldots, M\}$ and performs branch-and-bound, and then continuous refinement can be applied. However, we found that fine-sampling poses (*e.g.* 16 equally spaced poses for the 1D case) without doing refinement actually works well in practice.

We briefly describe how to bound the loss term here. Firstly, given state s and the set $\mathcal{B}(s)$ of bounding boxes it contains, Δ_{loc} can be bounded as

$$\Delta_{loc}(B_i, B) = 1 - \frac{Area(B_i \cap B)}{Area(B_i \cup B)} \leq 1 - \frac{\min_{B \in \mathcal{B}(s)} Area(B_i \cap B)}{\max_{B \in \mathcal{B}(s)} Area(B_i \cup B)}, \tag{11}$$

which can be computed by considering the intersection and union regions of the set $\mathcal{B}(s)$. Secondly, $(1-\beta)\Delta_{pose}(\theta_i, \theta_m)$ is in fact a constant in any state, since θ_m is fixed. We add this constant to the upper and lower bounds for the localization loss to get bounds for the overall loss term.

We also generate multiple diverse solutions for the loss-augmented inference, as this speeds up empirical convergence for structural SVM learning, as shown in [23]. With branch-and-bound, this can be done in the same fashion as generating multiple solutions for test-time inference; the diversity of solutions can be enforced by performing non-maximum suppression on the loss values.

5.1 Online Mini-Batch Algorithm

In learning a continuously parameterized model over the pose space, all training examples are tightly coupled into a single learning problem. Efficient structural SVM learning algorithms are needed to keep the complexity of learning manageable. However, cutting plane methods [20, 22], which typically perform constraint generation over the training set multiple times, are usually slow to converge in practice. Instead of traversing the training set multiple times and revisiting examples for which constraints are likely to be satisfied already, we focus on learning from subsets of "important" examples and generating multiple diverse constraints per iteration.

We cast the learning problem into an online version by considering mini-batches of training examples. In iteration t, a subset S_t is sampled from the training set, and new constraints are generated only for examples in S_t. Model updates can be done using SMO-style dual updates [24, 25]. We then seek appropriate sampling strategies for finding "important" subsets that contribute to improving the current model. In practice, we consider two strategies: 1) sample according to the slack variables ξ_i, and 2) randomly permute the training set and take sequential mini-batches.

The rationale behind the first strategy is that a large ξ_i indicates severe constraint violation and an important example for improving the model. The second strategy deems the next subset of unseen examples as important. Despite their simplicity, in our experiments, both strategies enable the online algorithm to provide a speedup of an order of magnitude over cutting plane methods, while achieving comparable or better prediction performance.

6 Experiments

We evaluated our method on three publicly available datasets: EPFL Cars [7], Pointing'04 [26], and 3D Objects [1].

We intend to keep our object appearance models simple so as to cleanly demonstrate the effects of continuous pose parameterization. Therefore, all of our appearance models are *single rectangular templates*, using the linear kernel as K_s, without any notion of mixture components or parts. For pose parameterization, we use Gaussian RBF kernels as K_p. The RBF bandwidth γ and SVM's trade-off parameter C are determined via cross validation. During learning, we weight the localization and pose losses equally ($\beta = 0.5$).

To assess detection performance, we follow the Pascal VOC protocol [21] to compute the Average Precision (AP) for predicted bounding boxes. For continuous pose estimation, we report the Mean Angular Error (MAE). For completeness, we also quantize our continuous pose estimates into M bins and report the Mean Precision of Pose Estimation (MPPE), defined as the average along the diagonal of the M-way confusion matrix in [3].

In all experiments, in addition to comparing with leading methods, we compare to a baseline that learns 1-vs-all SVM classifiers for discretized poses with the same feature. Each SVM is initially trained using ground truth bounding boxes associated with the target pose as positive examples, and ground truth bounding boxes with other poses as negatives. Then hard negative mining is applied on the training set to iteratively add negative examples and retrain the SVM until convergence.

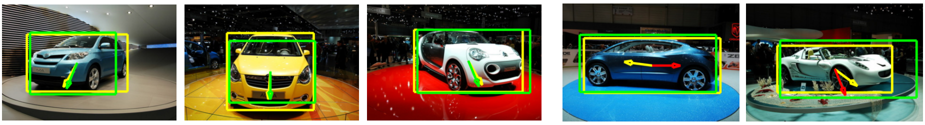
Table 1: Performance comparison on the EPFL Cars dataset [7]. Multi-view methods using 16 viewpoint bins [3, 12, 27] are listed. Regression methods [4–6] use ground truth bounding boxes (GT) as input. AP and MPPE: higher is better. MAE: lower is better. Top two results for each category in bold.

Method	Baseline	BnB	Refined	[3]	[12][b]	[27]	[6]	[5]	[4]	[15]
AP (%)	88.0	**100**	**100**	97	**97.5**	89.5	GT	GT	GT	GT
MAE°	36.7	**17.0**	**15.8**	27.2[a]	–	–	24.2	31.2	33.1	24.0
Median AE°	12.2	8.0	**6.2**	–	**6.9**	24.8	–	–	–	–
MPPE (%)	46.8	63.8	64.0	**66.1**	**69.0**	–	–	66.1[c]	70.3[c]	87.8[c]

[a] Obtained from direct correspondence with [3]'s authors.

[b] We cite [12]'s "3D^{2}PM-C Lin" variant with 16 viewpoint bins.

[c] [4, 5, 15] report percentages of $AE < 22.5°$, or 8-bin MPPEs.



(a)

(b)

Fig. 2: EPFL Cars results: (a) typical success cases; (b) example errors. Ground truth: yellow. Detection/pose estimation: green (correct), red (incorrect).

6.1 EPFL Cars Dataset

The EPFL Cars dataset [7] contains 20 different cars with views captured roughly 2°-3° apart on rotating platforms. This dataset has been studied in many previous works, *e.g.* [3, 12, 27], and is suitable for studying continuous pose estimation due to its relatively fine-grained and accurate pose annotations. A major challenge in this dataset is handling the near-180° confusions (*e.g.* front-back, left-right) or flipping errors, as noted in [7].

Setup. We first extract dense SIFT features from training images and cluster them into a codebook of 500 visual words using K-means, and construct our image feature as a 3-level spatial pyramid of visual words, with dimensionality 10500. The LLC encoding scheme of [28] is used, and L_2 normalization is applied to the feature vectors [29].

Following the standard test protocol in [7], we train on the first 10 car instances and test on the remaining 10. At test time, the initial solutions for each image are obtained using branch-and-bound with M=16 equally spaced seed poses. We then perform continuous pose refinement and pick the top-scoring solution, since each image contains exactly one car in EPFL Cars.

Results. Example results are shown in Fig. 2. We report our results both with respect to initial solutions (`BnB`) and continuously refined solutions (`Refined`) in Table 1, and compare them to competing methods.

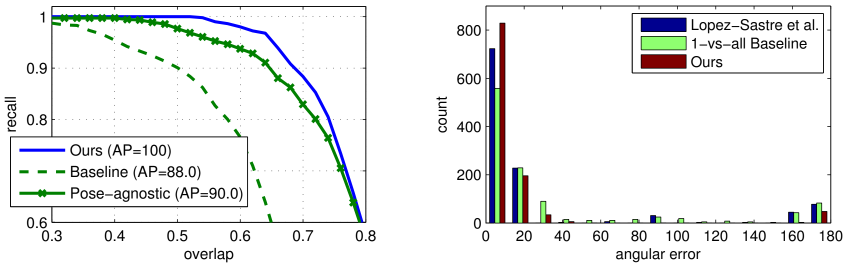


Fig. 3: Results on EPFL Cars. Left: closeup of overlap-recall curves for our method, the baseline, and a pose-agnostic detector learned with bounding box annotations. Right: histograms of angular errors for our method, the baseline, and Lopez-Sastre *et al.* [3].

Our method achieves **100%** AP and **15.8°** MAE, and outperforms all previous methods that report AP and MAE by a large margin, including regression methods [4–6] that use ground truth bounding boxes as input. We also achieve **6.2°** *Median* Angular Error, which is significantly better than [27]'s 24.8° and comparable to [12]'s 6.9° (16 viewpoint bins) and 4.7° (36 viewpoint bins). Note that [12] builds a much richer model by learning 3D part-based models directly from high-quality CAD models; in contrast, our model is a single part-free 2D object template, parameterized by pose, and learned on the original training set. When we quantize our continuous pose estimates into 16 bins, we obtain a 64.0% MPPE, which is comparable to the 69.0% and 66.1% reported by [12] and [3]. To our knowledge, our method gives the current state-of-the-art results in detection and continuous pose estimation on EPFL Cars.

To analyze detection results, in Fig. 3 we plot the overlap-recall curve [29]. The lowest overlap with ground truth for our detections is 53%, giving 100% AP when using the 50% overlap threshold. The 1-vs-all baseline produces a significantly worse overlap-recall curve, with the lowest overlap being just 14%. We also compare to a pose-agnostic detector trained using only bounding box annotations; its learning can be achieved in our formulation by only considering the localization loss, or setting $\beta = 1$ in Eq.(8). This improves the lowest overlap with ground truth to 23%, but still produces a noticeably worse overlap-recall curve compared to ours. We thus conclude that incorporating pose information into the detector helps improve detection performance.

Next, in Fig. 3 we plot the histograms of angular errors for our method, the 1-vs-all baseline, and Lopez-Sastre *et al.* [3], who learn a mixture of view-specific DPMs. We report [3]'s result since it gives the previous best performance with the original training set. The numbers of flipping errors (angular errors of more than 150°) for the three methods are: 52 (Ours), 129 (baseline), and 123 ([3]). We reduce the number of flipping errors in both methods by more than half. This confirms the benefit of learning a unified model over the pose space.

Online mini-batch learning. We also compared our mini-batch online algorithm against the one-slack cutting plane algorithm [22] in learning our structural

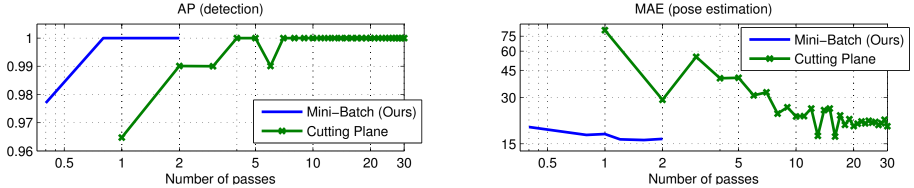


Fig. 4: Example learning curves from our online algorithm and the one-slack cutting plane algorithm [22] for learning the structural SVM model on EPFL Cars. Left: detection performance in AP. Right: pose estimation performance in MAE.

SVM model on EPFL Cars. We use the sequential mini-batch sampling strategy due to its simplicity. The time complexity is measured in *passes over training set*, or the average number of loss-augmented inference operations per training example. Learning curves for both methods are shown in Fig. 4.

Our online algorithm converges after two passes over training set, while the cutting plane algorithm typically requires 20–30 passes to converge, and to a worse solution (identical AP, higher MAE). We attribute the fast convergence of the online algorithm to the more frequent model updates and the ability to focus on "important" examples. The cutting plane algorithm emphasizes consistently improving the model with respect to the whole training set, but this may result in 1) increased computational efforts, and 2) the loss of emphasis on important examples, since their effects can be "averaged out". Our observations are consistent with those in the online learning community, *e.g.* [25].

6.2 Pointing'04 Dataset

Next, we turn to the problem of head pose estimation in the Pointing'04 dataset [26]. This dataset contains 2790 face images of 15 human subjects, captured from a total of 93 distinct poses parameterized by two angles $(pitch, yaw)$. The images all have clean backgrounds and are not challenging for the detection task; therefore, we only evaluate pose estimation performance by making use of ground truth bounding boxes, as also done in [5, 6]. The approximately annotated poses are at least 15° apart in Pointing'04

Setup. We use the same features as Hara *et al.* [6] by cropping out the face regions from the images and extracting HOG descriptors from three scales, resulting in a 2124 dimensional feature vector. We follow the standard protocol in [26] and report five-fold cross validation MAEs for *pitch* and *yaw*. However, with five folds the randomness in splitting data was found to be statistically significant. We thus tried 10 random five-fold splits, and report the mean and standard deviation for five-fold MAEs from the 10 trials. The best five-fold MAEs (having the lowest average) are also reported.

Results. Table 2 summarizes the results. Our method consistently outperforms the baseline by an average of 2.07° and 1.78° in *pitch* and *yaw* respectively, which again speaks to the benefit of learning a unified parameterized model.

Table 2: Head pose estimation performances on the Pointing'04 dataset. Results for our method (`BnB` and `Refined`) are reported from 10 random trials of five-fold cross validation, in the format of `mean±std/best`. Top two results for each category in bold.

MAE°	Baseline	BnB	Refined	KRF[6]	[5]	kPLS[30]	[26]
pitch	6.37±.17	4.30±.16/**4.01**	5.25±.15/4.95	**2.51**	6.73	6.61	15.9
yaw	7.14±.16	5.36±.15/**5.20**	5.91±.14/5.71	**5.29**	5.94	6.56	10.1
average	6.76±.16	4.83±.13/**4.61**	5.58±.13/5.33	**3.90**	6.34	6.59	13.0

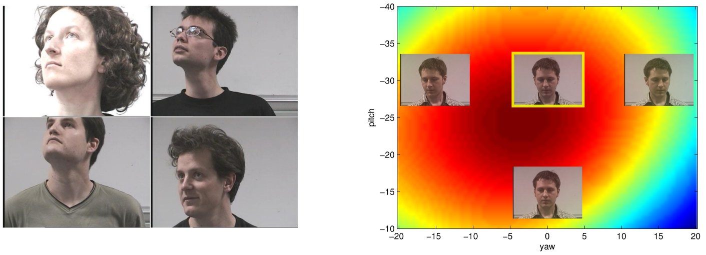


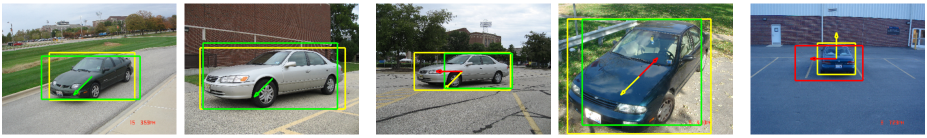
Fig. 5: Left: four training examples annotated with $(30°, 30°)$ in the Pointing'04 dataset. Approximate annotations affect the quality of our continuous model. Left: a test example (yellow box) annotated with $(-30°, 0°)$ and its three neighbors overlayed on top of the score map produced by our model. Although the scoring function does not peak exactly at $(-30°, 0°)$, it remains the top-scoring discrete pose.

A seemingly surprising fact is that the continuous refinement step yields higher MAEs. We note that this has to do with annotation quality. As illustrated in Fig. 5, pose annotations in Pointing'04 are at least 15° apart, and also carry noticeable label noise. Compared to the case of EPFL Cars (2°-3° apart and less noisy), this more significantly affects the quality of continuous parameterization. As a result, the scoring function in general does not peak exactly at discretized poses, which can result in nonzero angular errors even for poses already correctly estimated by discrete initialization. However, continuous refinement is highly consistent with discrete initialization, altering nearest-neighbor assignments to discrete poses only 0.36% of the time. Despite the coarse parameterization, we still significantly improve upon the discretized 1-vs-all baseline.

We also compare to previous methods [5, 6, 26, 30] in Table 2. Our method significantly outperforms Fenzi *et al.* [5], Haj *et al.* [30] and Gourier *et al.* [26] despite the use of more complex models in all three methods (1-NN classifier, kernel Partial Least Squares, and facial structure detection, respectively). Our best average MAE of 4.61° is slightly higher than [6]'s 3.90°, but note that our result is achieved by linear classifiers, while [6] employs a highly nonlinear kernel regression forest that also performs feature selection. We would also like to point out that [6], as well as [5, 30], is a regression method that requires clean input, whereas our method is fully capable of doing joint object detection and pose estimation, as we demonstrate in the other two experiments.

Table 3: 3D Objects: performance comparison for classes bicycle and car. Results for our method (BnB) are from 10 random trials, in the format of mean±std/best. Top two results for each category in bold.

Method		Baseline	BnB	[12]	[3]	[11]	[27]	[2]
bicycle	AP (%)	78.2±5.2	95.1±1.4/**96.8**	**97.6**	91	87.0	–	–
	MPPE (%)	98.7±2.1	94.0±3.3/**97.6**	**98.9**	90	87.7	–	96.2
car	AP (%)	85.4±8.8	98.2±3.9/97.8	**99.9**	96	94.9	**99.2**	–
	MPPE (%)	97.7±3.2	87.9±3.4/**93.0**	**97.9**	89	82.6	84.9	92.0



(a) (b) (c)

Fig. 6: Example results on car images in the 3D Objects dataset [1]: (a) typical success cases; (b) correct detections, wrong poses; (c) complete failure. Note the variation in actual poses in images having the same pose annotation (first four images).

6.3 3D Objects Dataset

We also evaluate our method in object detection and discrete pose classification on the 3D Objects dataset [1]. The dataset contains 8 object classes each having 10 instances, roughly annotated with 8 discrete viewpoints spaced at 45° apart. With such sparse views, object appearances tend to form discrete clusters rather than vary smoothly, making discrete classification methods more suitable. In fact, previous methods relying on mixtures of view-specific DPMs [3, 12] have obtained very competitive results on the 3D Objects dataset.

We shall only evaluate our branch-and-bound algorithm with discrete poses on the 3D Objects dataset, since its granularity of pose samples and accuracy of annotation are both insufficient for learning our continuously parameterized detector. Nevertheless, we are still interested in evaluating whether or not coarse parameterization and feature sharing can improve detector performance.

Setup. We use the same feature representation that we used with EPFL Cars: 3-level spatial pyramid of 500 SIFT visual words, with LLC encoding and L_2 normalization. We focus on the bicycle and car classes, as they are the most representative and they have been extensively studied in the literature.

As noted in [11], despite the large number of studies on 3D Objects, different test configurations have been reported, making a fully comprehensive comparison difficult. As in experiments with Pointing'04, we tried 10 different splits of the dataset, each time using images from 7 object instances for training, and the rest for testing. For viewpoint classification, we compute MPPE on the set of correct detections. We report from the 10 trials both the average and best performance (having the highest average of AP and MPPE).

Results. Table 3 reports results obtained by our method (BnB). We achieve significant improvements in AP over the baseline for both bicycle and car, on average by 16.9% and 12.8%, respectively. This again shows the benefit of pose parameterization for object detectors. However, the baseline does have higher MPPEs than BnB. Since the MPPE is computed on correct detections, this means that the 1-vs-all SVMs learned by the baseline are highly specific to their correponding views. However, collectively they produce many more misdetections for which pose estimates are hardly useful. The 1-vs-all SVMs are independently learned, and give essentially uncalibrated scores; this can lead to mutual confusions and degradation of collected detection performance.

We show example detection and pose estimation results for the car class in Fig. 6. Many of our pose classification errors are next-bin errors (by 45°) due to approximate annotations (see Fig. 6(b) for an example).

As can be seen in Table 3, our best results (96.8% AP and 97.6% MPPE for bicycle, 97.8% AP and 93.0% MPPE for car) are collectively better than those from all competing methods only except [12], while our average performance is also competitive. Again, we note that [12] learns rich 3D part-based models directly from high-quality CAD models, while our method learns a single parameterized 2D object template from roughly annotated images. Remarkably, our method consistently outperforms another more complicated system by Schels *et al.* [11], who also learn from CAD models and construct a dense part-based object representation over the viewsphere by fine-sampling viewpoints.

7 Conclusion

We propose a structured formulation to jointly perform object detection and pose estimation, by learning a single, continuously parameterized, discriminative object appearance model over the entire pose space. To solve the associated nonconvex inference problem, we design a cascaded algorithm with an efficient pruning step to generate diverse proposals, and a refining step that performs continuous optimization. For efficient model learning, we give simple but effective costraint generation strategies for a mini-batch online structural SVM learning algorithm, which converges significantly faster than batch algorithms. On three standard benchmarks in the combined task of object detection and pose estimation, our method performs better than, or on par with, state-of-the-art systems that are usually of higher complexity.

We focus on 1D and 2D viewing angles in this paper as this provides a very common parameterization of object appearance, appropriate for nearly any object and imaging scenario. Nevertheless, our formulation is general and can work with other continuous or even discrete factors that parameterize object appearance, such as articulated pose, directional lighting, phenotypes, and object subcategories. We are interested in exploring these factors in future work.

Acknowledgments. This work was supported in part by U.S. NSF grants 0910908 and 1029430.

References

1. Savarese, S., Fei-Fei, L.: 3D generic object categorization, localization and pose estimation. In: ICCV 2007
2. Gu, C., Ren, X.: Discriminative mixture-of-templates for viewpoint classification. In: ECCV 2010
3. Lopez-Sastre, R.J., Tuytelaars, T., Savarese, S.: Deformable part models revisited: A performance evaluation for object category pose estimation. In: ICCV 2011 Workshops
4. Torki, M., Elgammal, A.: Regression from local features for viewpoint and pose estimation. In: ICCV 2011
5. Fenzi, M., Leal-Taixé, L., Rosenhahn, B., Ostermann, J.: Class generative models based on feature regression for pose estimation of object categories. In: CVPR 2013
6. Hara, K., Chellappa, R.: Growing regression forests by classification: Applications to object pose estimation. In: ECCV 2014
7. Ozuysal, M., Lepetit, V., P.Fua: Pose estimation for category specific multiview object localization. In: CVPR 2009
8. Stark, M., Goesele, M., Schiele, B.: Back to the future: Learning shape models from 3D CAD data. In: BMVC 2010
9. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE TPAMI **32**(9) (2010)
10. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: CVPR 2012
11. Schels, J., Liebelt, J., Lienhart, R.: Learning an object class representation on a continuous viewsphere. In: CVPR 2012
12. Pepik, B., Gehler, P., Stark, M., Schiele, B.: 3D^{2}PM - 3D deformable part models. In: ECCV 2012
13. Xiang, Y., Savarese, S.: Estimating the aspect layout of object categories. In: CVPR 2012
14. Mei, L., Liu, J., Hero, A., Savarese, S.: Robust object pose estimation via statistical manifold modeling. In: ICCV 2011
15. Zhang, H., El-Gaaly, T., Elgammal, A., Jiang, Z.: Joint object and pose recognition using homeomorphic manifold analysis. In: AAAI 2013
16. Yuan, Q., Thangali, A., Ablavsky, V., Sclaroff, S.: Multiplicative kernels: Object detection, segmentation and pose estimation. In: CVPR 2008
17. Ionescu, C., Bo, L., Sminchisescu, C.: Structural SVM for visual localization and continuous state estimation. In: ICCV 2009
18. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. The annals of statistics (2008) 1171–1220
19. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Efficient subwindow search: A branch and bound framework for object localization. IEEE TPAMI **31**(12) (2009)
20. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR **6**(9) (2005)
21. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL visual object classes (VOC) challenge. IJCV **88**(2) (2010)
22. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural SVMs. Machine Learning **77**(1) (2009)
23. Guzman-Rivera, A., Kohli, P., Batra, D.: Faster training of structural SVMs with diverse M-best cutting-planes. In: AISTATS 2013

24. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. In: Advances in Kernel Methods. MIT Press, Cambridge, MA, USA (1999) 185–208
25. Bordes, A., Usunier, N., Bottou, L.: Sequence labelling SVMs trained in one pass. In: ECML PKDD 2008
26. Gourier, N., Hall, D., Crowley, J.L.: Estimating face orientation from robust detection of salient facial structures. In: ICPR 2004 Workshops
27. Glasner, D., Galun, M., Alpert, S., Basri, R., Shakhnarovich, G.: Viewpoint-aware object detection and pose estimation. In: ICCV 2011
28. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR 2010
29. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: ICCV 2009
30. Haj, M.A., Gonzalez, J., Davis, L.S.: On partial least squares in head pose estimation: How to simultaneously deal with misalignment. In: CVPR 2012