Multi-Output Learning for Camera Relocalization

Abner Guzman-Rivera[‡]* Pushmeet Kohli[†] Ben Glocker[♭]* Jamie Shotton[†]
Toby Sharp[†] Andrew Fitzgibbon[†] Shahram Izadi[†]

Microsoft Research[†] University of Illinois[‡] Imperial College London[♭]

Abstract

We address the problem of estimating the pose of a camera relative to a known 3D scene from a single RGB-D frame. We formulate this problem as inversion of the generative rendering procedure, i.e.*, we want to find the camera pose corresponding to a rendering of the 3D scene model that is most similar with the observed input. This is a non-convex optimization problem with many local optima. We propose a hybrid discriminative-generative learning architecture that consists of: (i) a set of M predictors which generate M camera pose hypotheses; and (ii) a 'selector' or 'aggregator' that infers the best pose from the multiple pose hypotheses based on a similarity function. We are interested in predictors that not only produce good hypotheses but also hypotheses that are different from each other. Thus, we propose and study methods for learning 'marginally relevant' predictors, and compare their performance when used with different selection procedures. We evaluate our method on a recently released 3D reconstruction dataset with challenging camera poses, and scene variability. Experiments show that our method learns to make multiple predictions that are marginally relevant and can effectively select an accurate prediction. Furthermore, our method outperforms the state-of-the-art discriminative approach for camera relocalization.*

1. Introduction

The problem of estimating the pose of a camera relative to a known 3D scene from a single RGB-D (RGB and depth) frame is one of the fundamental problems in computer vision and robotics, which enables applications such as vehicle or robot localization [3, 18], navigation [5, 10], augmented reality [17], and reconstruction of 3D scenes [16, 1]. While it is relatively easy to compute what a scene looks like from a particular camera viewpoint (using a renderer), it is generally very hard to estimate the viewpoint, *i.e.*, the camera pose, given an image of some scene only.

Camera relocalization can be formulated as an inverse problem where we search for the camera pose under which the rendered 3D scene is most similar to the observed input. This is a non-convex optimization that is hard to solve. Previous approaches in the literature have proposed the use of different optimization schemes and similarity measures. While some have used similarity measures based on descriptors computed over sparse interest points [15, 18, 8, 20], others have resorted to a dense approach [13]. All such methods suffer from the problem that the optimization is prone to getting stuck in local optima.

Recently, Shotton *et al.* [19] addressed the camera pose estimation problem by training a random-forest based predictor discriminatively, as opposed to solving an optimization problem directly. Although this approach substantially outperforms conventional methods on a challenging set of scenes, it has a fundamental limitation: the many-to-one nature of the learned mapping fails to model uncertainty properly, especially in situations where different camera viewpoints are associated with a similar rendering of the model. This is the case, *e.g.*, in the stairs view in Fig. 3a. In this paper, we propose a hybrid discriminative-generative learning method which overcomes this problem. Instead of learning a single predictor, we learn a set of M predictors each of which produces an independent estimate for the camera pose. Next, a selection procedure based on a similarity function takes charge of inferring the best pose given the outputs of the predictors.

The main contribution of this work is in learning to generate predictions that are 'marginally relevant', *i.e.*, both relevant and diverse. In other words, we show how to train a set of predictors that make *complementary* predictions. This multi-output prediction is effective in dealing with uncertainty stemming from ambiguities and multi-modality in the data, and from certain approximation errors (*e.g.*, in the model or the inference algorithm). Further, it enables the specialization of predictors to difficult or new cases. As a second contribution, we investigate the effectiveness of selection procedures based on evaluating a distance between the input RGB-D frame and a (partial) reconstruction or rendering of the 3D scene. We also develop a mechanism that

*Work done while author was at Microsoft Research.

aggregates predictions and often yields increased accuracy.

We evaluate our method on a recently released dataset of challenging camera pose estimation problems. Experiments show that our method can indeed learn to make multiple predictions that are marginally relevant and can effectively select an accurate prediction. Furthermore, our method outperforms state-of-the-art discriminative learning based methods for camera relocalization.

Related Work on Multi-Output Prediction. A number of methods have been proposed for the problem of multiple output prediction. Yang and Ramanan [21] trained a single prediction model and estimated the N-best configurations under the model. This and related approaches have the problem that the N-best solutions tend to be very similar to each other. Batra *et al*. [4] proposed a method enforcing a penalty such that the multiple predictions are different from each other. Although this method generates better results, it comes at the cost of increased computational complexity.

The aforementioned approaches rely on a single-output model to generate multiple predictions. The model itself is trained either to match the data distribution (max-likelihood) or to give the ground-truth solution the highest score by a margin (max-margin) – both learning objectives do not encourage multiple predictions to be marginally relevant. Guzman-Rivera *et al*. [12] overcome this issue by posing the problem as a 'Multiple Choice Learning' problem. Instead of generating multiple solutions from one model, multiple models are trained so that each model would *explain* different parts of the data. Our work is related to [12] but has two key advantageous differences. Our method trains a set of predictors iteratively by computing a single marginally relevant predictor at every iteration. Thus, adding predictors to our model is straightforward since predictors already in the set need not be modified. In contrast, the approach in [12] requires re-training of all M predictors. Further, learning in general is more efficient in our approach due to its greedy nature, while the method in [12] is computationally expensive due to iterative re-training of all predictors.

2. Problem Formulation and Preliminaries

Notation. We use $[n]$ to denote the set of integers $\{1, 2, \ldots, n\}$; $\mathcal{I}$ to denote an input RGB-D frame; H to denote a 6 d.o.f. camera pose matrix mapping camera coordinates to world coordinates[1]; and $\mathbb{M}$ to denote a 3D model of the scene.

2.1. Camera Pose Estimation as an Inverse Problem

A generative procedure (*e.g*., depth raycaster) can be used to *reconstruct* a 'view' of a scene from a given viewpoint. Let $\mathbb{R}(H; \mathbb{M})$ be the view of 3D model $\mathbb{M}$ from the viewpoint given by camera pose H. Then, camera pose estimation may be cast as an inverse problem for the camera pose H^* associated with the view *most similar* to the input frame $\mathcal{I}$. Formally, we would like to solve for the pose minimizing

$$H^* = \operatorname*{argmin}_H \Delta_\xi \left(\mathcal{I}, \mathbb{R}(H; \mathbb{M})\right) \ , \tag{1}$$

where ξ is a choice of metric (*e.g*., L1) for measuring the reconstruction error. As mentioned earlier, this is a non-convex optimization problem that is hard to solve.

2.2. The Direct Regression Approach

Instead of solving the inverse problem defined above, we could treat camera pose estimation as a regression problem. Given a set of RGB-D frames with known camera poses $S = \{(\mathcal{I}_j, H_j) : j \in [n]\}$, we want to learn a mapping g, taking an RGB-D frame as argument and predicting a 6 d.o.f. camera pose, such that the the empirical risk is minimized

$$L(\boldsymbol{\Theta}; S) = \frac{1}{n} \sum_{j=1}^{n} \ell(H_j, g(\mathcal{I}_j; \boldsymbol{\Theta})) \ . \tag{2}$$

Here, the task-loss ℓ measures the error between the ground-truth pose H_j and the predictor's output $g(\mathcal{I}_j; \boldsymbol{\Theta})$, and $\boldsymbol{\Theta}$ is the set of parameters to be learned.

Shotton *et al*. [19] propose the 'scene coordinate regression' method that is based on this principle. In their method, a set of regression trees (a forest) $\mathcal{F}$, parameterized by $\boldsymbol{\theta}$ is trained to infer correspondences between RGB(-D) image pixels and 3D coordinates in scene space. The forest is trained to produce 3D correspondences at any image pixel, which allows the approach to avoid the traditional pipeline of feature detection, description and matching. Given just three perfect image to scene correspondences, the camera pose could be uniquely recovered. However, to make the method robust to incorrect correspondences and noise, an energy function, which essentially measures the number of pixels for which the forest predictions agree with the camera pose hypothesis H, is minimized. More formally, the prediction function for the camera pose given frame $\mathcal{I}$ and regression forest $\boldsymbol{\theta}$ is defined as

$$g(\mathcal{I}; \boldsymbol{\theta}) = \operatorname*{argmin}_H \sum_{i \in \mathcal{I}} \rho \left(\min_{\mathbf{m} \in \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{p}_i)} \|\mathbf{m} - H\mathbf{x}_i\|_2 \right) \ , \tag{3}$$

where: $i \in \mathcal{I}$ is a pixel index; ρ is a robust error function; $\mathcal{F}_{\boldsymbol{\theta}}(\mathbf{p}_i)$ are the regression forest predictions for pixel $\mathbf{p}_i$ (one or more per tree); and $\mathbf{x}_i$ are 3D coordinates in *camera* space corresponding to pixel $\mathbf{p}_i$ (obtained by re-projecting depth image pixels). A modified preemptive RANSAC algorithm is used to minimize (3). For speed, the energy can be evaluated on a sparse set of randomly sampled pixels rather than densely over the image.

[1] For convenience, we slightly abuse homogeneous matrix multiplication to allow H to map a 3-vector to a 3-vector.

2.3. Pose Refinement using the 3D Model

The output of a predictor such as (3) can be *refined* using the 3D model. We represent the 3D model $\mathbb{M} : \mathbb{R}^3 \to [-1, 1]$ of a scene as a truncated signed distance function [7] evaluated at continuous-valued positions by tri-linearly interpolating the samples stored at voxel centers. The surface of the model is then given by the level set $\mathbb{M}(\mathbf{y}){=}0$ where $\mathbf{y}$ is a 3D position in scene space.

Let $\mathbf{x}_i \in \mathbb{R}^3$, $i \in [k]$ be a set of observed 3D coordinates in camera space. Then, a pose estimate H may be refined by searching for a nearby pose that best aligns the camera space observations with the surface represented by $\mathbb{M}$. Since $\mathbb{M}$ is a (signed) distance to the surface, the value of $[\mathbb{M}(\bar{H}\mathbf{x}_i)]^2$ will be small when $\bar{H}$ aligns the camera space observations closely to the current model of the surface. Thus, we can *refine* a camera pose by minimizing the following nonlinear least-squares problem after initializing $\bar{H}$ with the estimate

$$H_{\text{ref}} = \operatorname*{argmin}_{\bar{H}} \sum_i [\mathbb{M}(\bar{H}\mathbf{x}_i)]^2 \; . \tag{4}$$

This minimization problem can be solved by optimizing over the rotational and translational components of $\bar{H}$ using the Levenberg-Marquardt algorithm (this is similar to the method of [6]). In our experiments we will evaluate the effect of refinement on the poses estimated with the models we compare.

3. Proposed Approach

The direct regression approach is not rich enough to model the one-to-many nature of the mapping between images and camera viewpoints. Our method for camera relocalization works by *generating* multiple hypotheses for the camera pose using a set (or 'portfolio') of regression-based predictors and then *selecting* the best hypothesis using the reconstruction error. In some sense, this can be seen as a regression-based approach for solving the inverse problem defined in (1). More formally, given multiple predictors $\{g(\cdot;\boldsymbol{\theta}_m) : \boldsymbol{\theta}_m \in \boldsymbol{\Theta}\}$ and a choice ξ of reconstruction error, we approximate (1) by

$$f(\mathcal{I};\boldsymbol{\Theta},\xi) = \operatorname*{argmin}_{g(\mathcal{I};\boldsymbol{\theta}_m)} \Delta_\xi \left(\mathcal{I}, \mathbb{R}(g(\mathcal{I};\boldsymbol{\theta}_m); \mathbb{M})\right) \; . \tag{5}$$

Thus, the inversion amounts to computing the errors corresponding to each predictor output and taking the minimum. Fig. 1 presents an overview of the architecture we propose.

For this approach to succeed, we need to develop a learning algorithm for a set of predictors whose predictions are marginally relevant, *i.e.*, the predictions must *cover* the space of all Hs well. In Section 3.1 we develop an algorithm for training such a set of predictors, where each of them is of discriminative form as in (3). We then discuss in Section 3.2 how different choices for the reconstruction error affect the accuracy of our method.

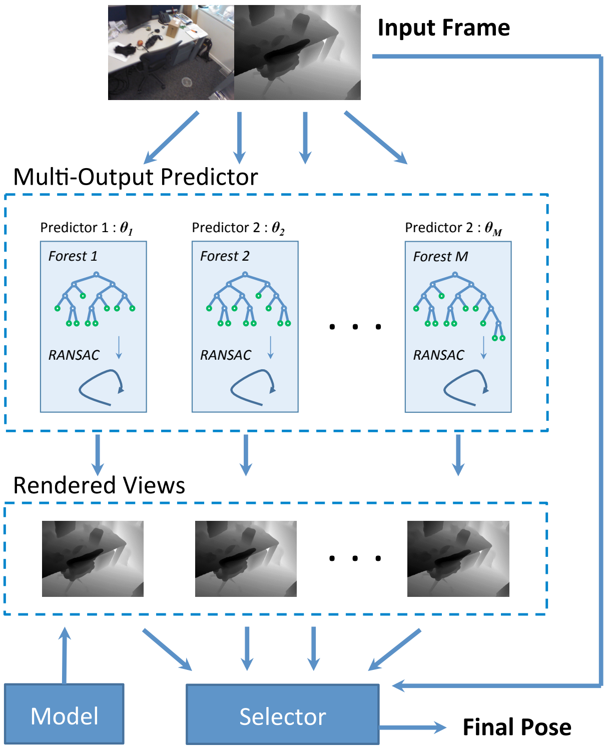


Figure 1. **Overview of the proposed method.** We propose a two-stage approach where the first stage generates M *marginally relevant* predictions; and the second stage infers an accurate pose from the first stage's predictions.

Algorithm 1 Learn Multi-Output Predictor

```
Inputs: $S = \{(\mathcal{I}_j, H_j) : j \in [n]\}, M, \xi, \ell, \sigma$
Initialize with uniform weights: $\mathbf{w}_0 \leftarrow \frac{1}{n} \cdot \mathbf{1}$
$\boldsymbol{\Theta}_0 \leftarrow \{\}$
for $t = 1, \ldots, M$ do
    $\boldsymbol{\Theta}_t \leftarrow \boldsymbol{\Theta}_{t-1} \cup \text{TrainForest}_{[19]}(S, \mathbf{w}_{t-1})$
    $\mathbf{w}_t \leftarrow \text{UpdateWeights}(\boldsymbol{\Theta}_t, \mathbf{w}_{t-1}, \ell, \xi, \sigma)$   (see (7))
end for
return $\boldsymbol{\Theta}_M$
```

3.1. Learning Marginally-Relevant Predictors

An approach to learning a diverse set of predictors is to train each of the predictors on different data, *e.g.*, [9, 12]. The key question here is to decide which data to use for training each predictor. Equivalently, the problem can be posed as the creation of multiple groups of training instances, such that training a predictor on each group yields an accurate set of predictors. Note that groups may overlap or, more generally, assignments of instances to groups may be probabilistic. Further, the number of groups could be determined during the optimization.

Grouping (or clustering) of data instances can be random (*e.g.*, [19]); performed in input space via algorithms such as k-means (*e.g.*, [11]); or, better, driven by a task-loss [12]. Here, we pursue a new iterative loss-driven approach that resembles boosting.

Algorithm 1 summarizes our procedure for training a multi-output predictor. At every iteration, a predictor of the form (3) is trained using the learning algorithm in [19], with the important difference that examples from each image j contribute to the forest training objective in proportion to some weight $\mathbf{w}(j)$. The first iteration uses uniform weights, but weights on later iterations are a function of the loss achieved by the predictors already trained. Importantly, this mechanism allows multiple modes in the empirical distribution of the training data to become re-weighted (as driven by the task-loss).

The standard task-loss ℓ for camera pose estimation (defined in (2)) compares a single prediction with ground-truth. To quantify the performance of a multi-output predictor $\mathbf{\Theta}_t$, we define the multi-output loss as

$$\ell_j(\mathbf{\Theta}_t;\xi) = \ell(H_j, f(\mathcal{I};\mathbf{\Theta}_t,\xi))\ , \tag{6}$$

where $f(\mathcal{I};\mathbf{\Theta}_t,\xi)$ is simply the camera pose hypothesis corresponding to the lowest reconstruction error among the outputs of predictors trained in iterations $[t]$ – as defined in (5). Equipped with a multi-output loss we define the following weight update rule

$$\mathbf{w}_t(j) \leftarrow \frac{\mathbf{w}_{t-1}(j)}{Z_t}\left(1-\exp\left\{\frac{-\ell_j(\mathbf{\Theta}_t;\xi)}{\sigma^2}\right\}\right), \tag{7}$$

where Z_t is a normalization factor and parameter σ roughly controls the *diversity* of the predictors – higher σ allowing larger weight ranges. This update rule adjusts the weight of individual training instances based on the loss incurred by the multi-output model.

It is important that during training we use the same definition of multi-output loss as will be used at test-time. This means we should use the same task-loss ℓ in (6) and the same reconstruction error ξ in (5), (6) for both training and testing.

3.2. Selecting a Good Hypothesis

Here we study the problem of selecting a good camera pose estimate from a set of candidates $\{H_m : m \in [M]\}$. More specifically, we investigate the effectiveness of several reconstruction errors, which we denote by ξ in (5).

Let D_{in} denote the input depth image and D_H denote a depth image raycast using pose estimate H and 3D model $\mathbb{M}$. As a first choice for ξ consider the L1 distance

$$\Delta_{\text{L1}}\left(\mathcal{I},\mathbb{R}(H;\mathbb{M})\right) = \sum_i |\, D_{\text{in}}(i) - D_H(i)\,|\ , \tag{8}$$

where the summation is over pixels.

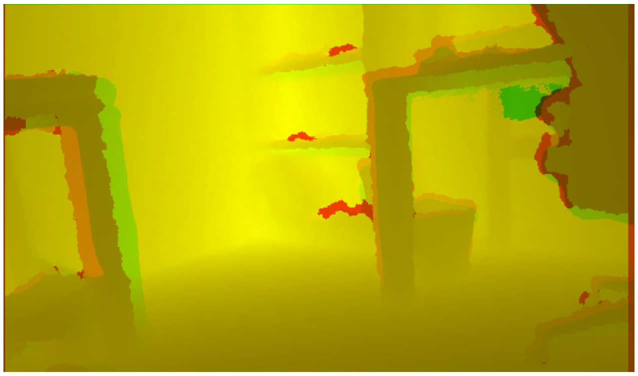


Figure 2. Difficult case for the L1 reconstruction error due to model distortion. Input depth (red channel) and depth raycast from ground-truth (green channel) are shown superimposed. Observe how it would be impossible to align the legs of both desks simultaneously.

We found the L1 distance to be sensitive to noise in the input depth images, and to distortions in the model (*e.g.*, due to camera-tracking drift during model reconstruction). For instance, see Fig. 2 where model distortion has led to a large distance between the input and the depth rendered from the ground-truth camera pose.

One way to make the distance robust to model-mismatch is to concentrate on shape. For instance, this can be done through the use of a symmetric chamfer distance on edges detected on the depth frames (2DDT). This approach, however, is problematic in that edge detection is also very sensitive to input noise and in that it completely ignores valuable depth information.

A robust error that we found superior to L1 is close to the error used in the objective (4) of our model-based refinement algorithm

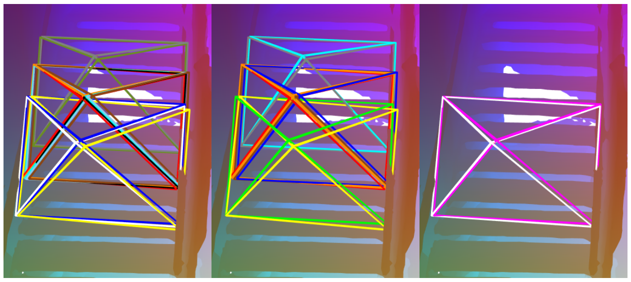
$$\Delta_{\text{3DDT}}\left(\mathcal{I},\mathbb{R}(H;\mathbb{M})\right) = \sum_i |\, \mathbb{M}(H\mathbf{x}_i)\,|\ . \tag{9}$$

We refer to this reconstruction error as 3D-Distance Transform (3DDT). An additional benefit of 3DDT compared to L1 or 2DDT is that it does not require raycasting.

4. Hypothesis Aggregation

The performance of the reconstruction errors above is impaired by noise in the input data and problems in the 3D models such as missing data or distortion. This led us to pursue the aggregation of poses in a way reminiscent of other ensemble methods. Algorithm 2 summarizes our pose aggregation procedure. It takes as arguments a set of pose hypotheses $\{H_m : m \in [M]\}$; parameters δ and ϵ for the clustering of hypotheses; and parameters ξ and ζ for cluster scoring and selection.

Procedure 'Cluster' performs agglomerative clustering on the poses using distance measure $\delta(H,H')$. Clustering stops when the distance between clusters exceeds ϵ. We explored several options for distance $\delta(H,H')$ between poses



(a) Predictions. (b) Cluster means. (c) Best cluster.

Figure 3. Ambiguities on scene Stairs and results from our two-stage approach. (a) M predictions shown as camera frusta; ground-truth (white); and selector's pick (black). (b) Clusters (means) created during aggregation. (c) Poses in best-scoring cluster (pink); cluster mean (magenta); and ground-truth (white).

Algorithm 2 Aggregation of Pose Hypotheses

```
Input: $\mathbf{H} = \{H_m : m \in [M]\}$, $\delta$, $\epsilon$, $\xi$, $\zeta$
$\mathbf{C} \leftarrow$ Cluster($\mathbf{H}, \delta, \epsilon$) see Section 4
$C \leftarrow \text{argmin}_{C \in \mathbf{C}}$ Score($C, \xi, \zeta$) see (11)
return Mean($C$)
```

including: Euclidean distance between the translation components, angular distance between the rotation components, and absolute difference on reconstruction errors, *i.e.*,

$$|\, \Delta_{\bar{\xi}}\left(\mathcal{I}, \mathbb{R}(H; \mathbb{M})\right) - \Delta_{\bar{\xi}}\left(\mathcal{I}, \mathbb{R}(H'; \mathbb{M})\right) \,| . \tag{10}$$

Once the poses are clustered we use the following rule to score clusters

$$\text{Score}(C) \leftarrow \frac{\zeta^{|C|-1}}{|C|} \sum_{H \in C} \Delta_{\xi}\left(\mathcal{I}, \mathbb{R}(H; \mathbb{M})\right) . \tag{11}$$

This is a voting mechanism with a preference for larger clusters whenever $\zeta < 1$. Finally, we take the 'mean' of the poses in the best-scoring cluster as our aggregate prediction. More precisely, we combine the poses in the cluster linearly with uniform weighting as suggested in [2].

Parameters for clustering and scoring, as summarized in Algorithm 2, were tuned on held out validation data.

5. Evaluation

5.1. Experimental setup

Dataset. We use the 7 Scenes dataset of [19] to evaluate our approach. The dataset consists of 7 scenes ('Chess', 'Fire', 'Heads', 'Office', 'Pumpkin', 'RedKitchen', and 'Stairs') which were recorded with a Kinect RGB-D camera at 640 × 480 resolution. Each scene is composed of several *camera tracks* that contain RGB-D frames together with ground-truth camera poses. The authors of the dataset used the KinectFusion system [16] to obtain ground-truth poses and to reconstruct 3D models like those described in Section 2.3. Both the RGB and depth components of the input frames exhibit ambiguities (*e.g.*, repeated steps in Stairs), specularities (*e.g.*, reflective cupboards in RedKitchen), motion-blur, differing lighting conditions, flat surfaces, and sensor noise.

Metrics. Following previous work, we report accuracy as the percentage of test frames for which the inferred camera pose passes a correctness criterion. A correct pose must be within a 5cm translational error and 5° angular error of ground-truth (for comparison, the scenes have sizes up to 6m on a side). This metric corresponds to the $0/1$ loss,

$$\text{PC}_j(H; t, r) = \begin{cases} 0 & \text{if } \delta_{\text{t}}(H_j, H) < t \\ & \quad \text{and } \delta_{\text{r}}(H_j, H) < r \\ 1 & \text{otherwise} \end{cases} , \tag{12}$$

where δ_{t} and δ_{r} are translational and rotational pairwise distances respectively; $t = 5$cm; and $r = 5°$.

We report model performance at every iteration of training, *i.e.*, accuracy of the pose minimizing (5) (the selector's pick) for sets of predictors $[t]$ for $t \in [M]$. For intermediate models, *i.e.*, $t < M$, no aggregation is used since the aggregation procedure was tuned on validation data for full-models only (*i.e.*, using all M predictors). Thus, we only report accuracy for aggregate poses given M predictions.

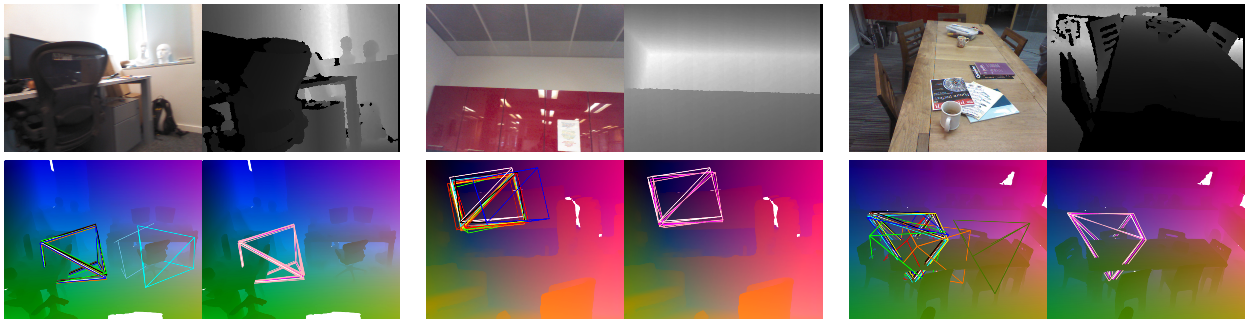
We report an 'Oracle' metric which is an upper-bound on the performance of the selector. The Oracle metric is the obtainable accuracy if we could always chose the best prediction within the set of candidates. As before, we report Oracle performance for models at every iteration.

Baselines. We compare our approach against the model of [19] that was shown to achieve state-of-the-art camera relocalization on RGB-D data. To obtain an even stronger baseline, we extended the RANSAC optimization of [19] to output the M best hypotheses ('M-Best'). We refer to the first baseline as CVPR13 and to the latter as CVPR13 + M-Best. Note that the CVPR13 baseline is equivalent to the predictor trained (with uniform weights) at the first iteration of Algorithm 1 (t=1 in the plots).

For the CVPR13 + M-Best baseline we report performance on the first t-Best hypotheses for $t \in [M]$. Also, we tune the aggregation procedure and report aggregate pose accuracies using all M hypotheses.

Train and test samples. For each scene we took 1000 uniformly spaced frames from the (concatenated) training camera tracks. This was done to reduce training time and because contiguous frames are largely redundant.

We also randomly sampled 1000 frames from the test camera tracks of each scene. Half of these were used for tuning the parameters of the aggregation procedure (Algorithm 2) and the other half were used for testing. Note that for some scenes, *e.g.*, Heads, this sampling yields all the available training and test data, while for others, *e.g.*, RedKitchen, only a small fraction of the available data is used.



(a) Office. (b) Pumpkin. (c) RedKitchen.

Figure 4. Qualitative results. Top row: input RGB-D frames. Bottom row: Pair-left: M predictions (colors); ground-truth (white); and selector's pick (black). Pair-right: Poses in best-scoring cluster (pink); cluster mean (magenta); and ground-truth (white).

5.2. Results

Qualitative. Fig. 3 and 4 show illustrative prediction examples. We observe that the predictions of the multi-output model are indeed complementary and often cover multiple possibilities for ambiguous inputs. For example, multiple steps in the Stairs scene (Fig. 3a), multiple desks/chairs in the Office scene (Fig. 4a), and multiple sections of the long flat cabinet in the Pumpkin scene (Fig. 4c).

Selector evaluation. We performed a set of experiments to compare the effectiveness of our different pose-selection mechanisms. These experiments were carried on all scenes and average results over the 7 scenes are reported.

We trained models using Algorithm 1 with parameters: M=8; $\xi \in \{\mathrm{L1}, \mathrm{3DDT}, \mathrm{Oracle}\}$; $\ell \in \{\mathrm{PC}, \mathrm{PCS}\}$; and $\sigma \in \{0.01, 0.1, 0.2, 1.0\}$. Here, PCS is a convex upper-bound on the $0/1$ loss

$$\mathrm{PCS}_j(H; t, r) = \max\left(\frac{\delta_{\mathrm{t}}(H_j, H)}{t}, \frac{\delta_{\mathrm{r}}(H_j, H)}{r}\right) . \quad (13)$$

We found re-weighting with PCS to be superior to PC because the latter leads to trivial weightings of zero-weight for examples with no loss and uniform weights for the rest of the examples.

Fig. 5a (top) compares L1 and 3DDT selector performance (averaged over all scenes). On average, 3DDT is superior to L1 (which in preliminary experiments we found to be superior to 2DDT). We report only results for models *trained* using ξ=Oracle as these are most meaningful for comparing the efficacy of different selectors (otherwise predictors are trained to compensate for the specific reconstruction error used during training).

We also compared selectors when hypotheses are refined using the model-based refinement from Section 2.3 *at test-time*. Fig. 5a (bottom) shows average performance for this experiment. Again, 3DDT is superior.

Given the superior performance and efficiency of 3DDT, our subsequent and more extensive end-to-end evaluation is limited to 3DDT.

End-to-end (no refinement). These experiments compare our two-stage approach with the baselines. We report average results over 5 runs for each scene (with different random seeds) and over the 7 scenes. We trained models using Algorithm 1 with parameters: M=10; ξ=3DDT; $\ell \in \{\mathrm{PC}, \mathrm{PCS}\}$; and $\sigma \in \{0.01, 0.1, 0.2, 1.0\}$.

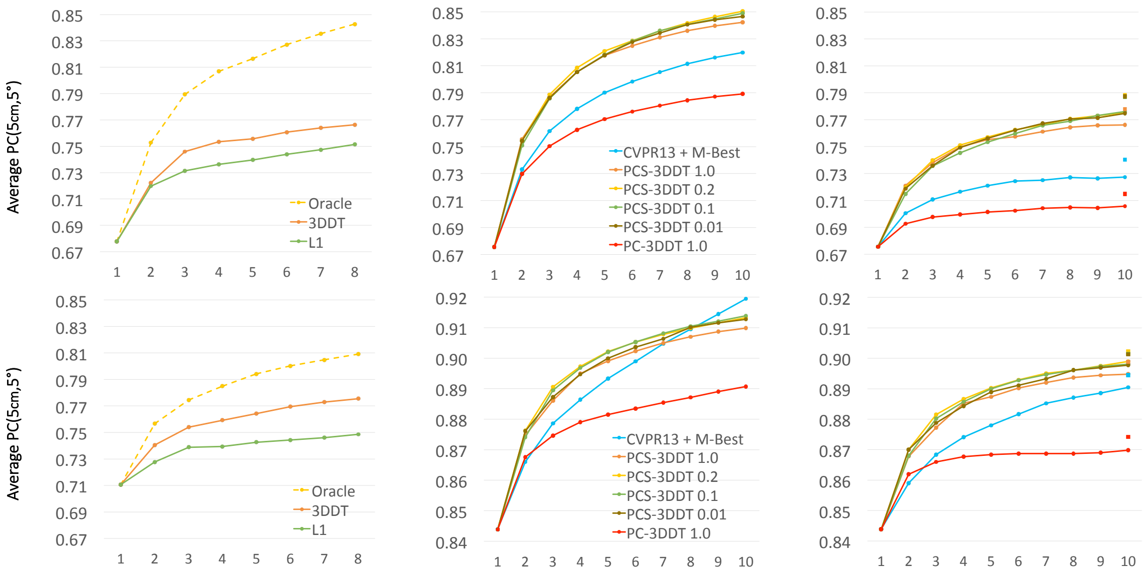
Fig. 5b (top) shows average Oracle performance of the baselines and several multi-output models. Fig. 5c (top) shows, for the same set of models, the average performance when using 3DDT for selection and when using the aggregation Algorithm 2 (squares at t=10). While scene-averaging does hide somewhat contrasting behaviors on the different scenes, a few observations are warranted. First, the CVPR13 + M-Best baseline does produce good hypotheses as shown by the high Oracle performance. However, these are not tuned to the selector and thus, it becomes more difficult to select good hypotheses at test-time.

The multi-output models trained with the PCS upper-bound give the best results, achieving a $\sim$5% average accuracy improvement w.r.t. the CVPR13 + M-Best baseline. Aggregation (carried only on full-models and shown by the square markers at t=10) achieved $\sim$1.5% average accuracy improvement w.r.t. the 3DDT selector.

Scene-averaging also hides effects of parameter σ of the re-weighting rule (7) but results on individual scenes (available in the supplementary) reveal that different scenes have different diversity requirements.

End-to-end with refinement. Fig. 5b (bottom) and 5c (bottom) show results for the *same* models when model-based refinement is applied at *test-time* (Section 2.3). Trends are roughly similar. Again, the CVPR13 + M-Best baseline has a very high average Oracle performance (in fact the highest of all models). Still, the multi-output models are best on the actual end-to-end evaluation and achieve a $\sim$1% average accuracy improvement.

Note that refinement was not used during training and thus the multi-output models are tuned for a different test scenario (*i.e.*, that without refinement). For refined poses aggregation has a limited effect but still achieves a $\sim$0.5% average accuracy improvement w.r.t. the 3DDT selector.



(a) Selector comparison (b) Oracle (c) 3DDT and aggregator (squares)

Figure 5. Average $\mathrm{PC}(5\mathrm{cm}, 5^\circ)$ (y-axis) over all scenes (5 runs per scene for b,c) *vs*. training iteration t (x-axis). Top row: no refinement. Bottom row: refinement at *test-time*. (a) Performance of reconstruction errors when the predictors are fixed. (b,c) Comparison of multi-output models and baselines. Legends indicate loss, selector and σ used during training. In (c) squares at t=10 correspond to aggregate poses. Note that the CVPR13 baseline of [19] corresponds to the performance at t=1.

5.3. Computational Implications

We now contrast the gains achieved with our multi-output prediction method with those obtained through model refinement. In Fig. 6 we include average results for individual scenes (5 runs per scene). Each plot compares the CVPR13 + M-Best baseline combined *with refinement* at test-time, with one multi-output model using *no refinement*. For each scene, we selected one of the multi-output models from previous plots (*i.e*., we tuned σ) using the validation data.

On these plots we see that our two-stage approach is superior to the CVPR13 baseline (*i.e*., the model of [19]) on *every* scene. The accuracy improvements range from ∼5% on Pumpkin to ∼20% on Stairs. Further, on scenes Chess, Office and RedKitchen our approach *without refinement* outperforms the CVPR13 + M-Best baseline *with refinement*. However, on scenes Heads, Pumpkin and Fire, refinement has a major effect with ∼44%, ∼14% and ∼12% accuracy improvements, respectively.

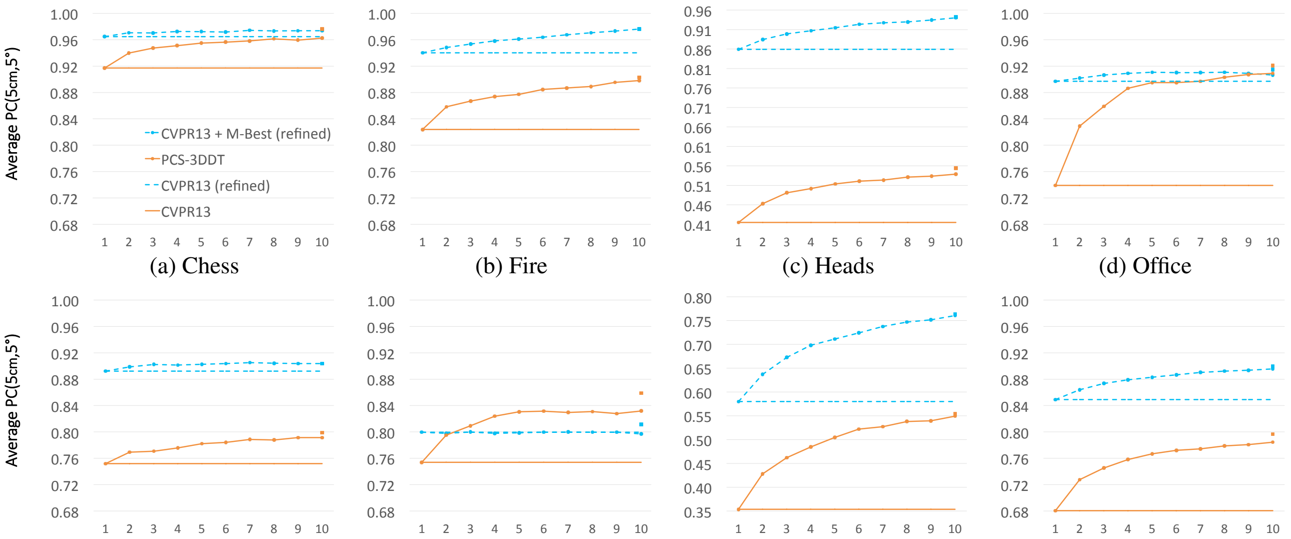
While both approaches, multi-output prediction and model-based refinement, lead to significant improvements they differ in their computational cost. The computational complexity of our multiple-output prediction system at test-time scales linearly with the number of predictors. This complexity could be significantly reduced by reusing tree structures and only updating the leaf distributions when generating multiple predictors. In contrast, the improvements obtained through model-based refinement come at a high computational cost because of the iterative nature. Furthermore, our multi-output method can be trivially parallelized by running individual predictors on different cores.

As an aside, we note that greater performance gains could be attained by combining multi-output prediction and model-based refinement (*i.e*., refinement would need to be used during training and testing of multi-output models).

6. Conclusion

We have proposed a hybrid, discriminative-predictor generative-selector, approach to inversion problems in computer vision that consists of: (i) a multi-output predictor; and (ii) a ‘selector’ or ‘aggregator’ that is able to select or infer a good prediction given a set of predictions. We proposed a procedure to train a set of predictors that make marginally relevant predictions and showed that the training procedure is able to tune the models for the selection stage to be used at test-time. We demonstrated that the proposed approach leads to significant accuracy improvements when applied to the problem of camera relocalization from RGB-D frames.

There are a number of interesting directions for future work. With regards to camera relocalization, while our approach can cope with certain sources of failure (*e.g*., ambiguity, multi-modality or test-train distribution mismatch), it would be beneficial to address other sources of failure for the model of [19]. Also, our approach is amenable to distributed learning, *e.g*., for camera relocalization in very large scenes. For such cases, predictors could be learned on disjoints subsets of training data (*e.g*., corresponding to different rooms) and, like we do here, the selection mechanism would be responsible for determining the right prediction at test-time.



(a) Chess (b) Fire (c) Heads (d) Office

(e) Pumpkin (f) RedKitchen (g) Stairs (h) All scene average

Figure 6. Average $\mathrm{PC}(5\mathrm{cm}, 5^{\circ})$ (y-axis) (5 runs per scene) *vs.* training iteration t (x-axis). Comparison of the proposed approach *without* refinement (orange) against the CVPR13 + M-Best baseline *with* model-based refinement (blue). For (a), (d) and (f) the accuracy improvement from our approach is higher than that of model-based refinement. Further, on all scenes our approach is better than the CVPR13 baseline of [19] (performance at t=1). Squares at t=10 correspond to aggregate poses.

In the context of multi-output prediction it would be interesting to investigate the possibility of training models under an explicit measure of diversity. For instance, we could devise a train-time loss that is augmented with a penalty for lack of diversity. Also interesting would be the evaluation of models such as determinantal point processes [14].

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009. 1

[2] M. Alexa. Linear combination of transformations. In *SIGGRAPH*, 2002. 5

[3] S. Atiya and G. D. Hager. Real-time vision-based robot localization. *Robotics and Automation*, 1993. 1

[4] D. Batra, P. Yadollahpour, A. Guzman-Rivera, and G. Shakhnarovich. Diverse M-best solutions in Markov random fields. In *ECCV*, 2012. 2

[5] M. A. Brubaker, A. Geiger, and R. Urtasun. Lost! Leveraging the crowd for probabilistic visual self-localization. In *CVPR*, 2013. 1

[6] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3D reconstruction using signed distance functions. In *RSS*, 2013. 3

[7] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996. 3

[8] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *PAMI*, 2007. 1

[9] O. Dekel and O. Shamir. There's a hole in my data space: Piecewise predictors for heterogeneous learning problems. In *AISTATS*, 2012. 3

[10] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation: a survey. *PAMI*, 24(2):237–267, 2002. 1

[11] Q. Gu and J. Han. Clustered Support Vector Machines. In *AISTATS*, 2013. 4

[12] A. Guzman-Rivera, D. Batra, and P. Kohli. Multiple choice learning: Learning to produce multiple structured outputs. In *Proc. NIPS*, 2012. 2, 3, 4

[13] G. Klein and D. Murray. Improving the agility of keyframe-based SLAM. In *ECCV*, 2008. 1

[14] A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3), 2012. 8

[15] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *PAMI*, 28(9), 2006. 1

[16] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. ISMAR*, 2011. 1, 5

[17] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, 2013. 1

[18] S. Se, D. G. Lowe, and J. J. Little. Vision-based global localization and mapping for mobile robots. *Robotics*, 2005. 1

[19] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013. 1, 2, 3, 4, 5, 7, 8

[20] B. Williams, G. Klein, and I. Reid. Automatic relocalization and loop closing for real-time monocular SLAM. *PAMI*, 33(9):1699–1712, 2011. 1

[21] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 2