Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image

Eric Brachmann*, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, Carsten Rother
TU Dresden
Dresden, Germany
*eric.brachmann@tu-dresden.de

Abstract

In recent years, the task of estimating the 6D pose of object instances and complete scenes, i.e. *camera localization, from a single input image has received considerable attention. Consumer RGB-D cameras have made this feasible, even for difficult, texture-less objects and scenes. In this work, we show that a single RGB image is sufficient to achieve visually convincing results. Our key concept is to model and exploit the uncertainty of the system at all stages of the processing pipeline. The uncertainty comes in the form of continuous distributions over 3D object coordinates and discrete distributions over object labels. We give three technical contributions. Firstly, we develop a regularized, auto-context regression framework which iteratively reduces uncertainty in object coordinate and object label predictions. Secondly, we introduce an efficient way to marginalize object coordinate distributions over depth. This is necessary to deal with missing depth information. Thirdly, we utilize the distributions over object labels to detect multiple objects simultaneously with a fixed budget of RANSAC hypotheses. We tested our system for object pose estimation and camera localization on commonly used data sets. We see a major improvement over competing systems.*

1. Introduction

This work considers the task of estimating the 6D pose of object instances from a single RGB image. A solution to this problem is of high relevance in a variety of application scenarios such as robotics and augmented reality. The range of applications is even broader when considering camera localization as a special case of object pose estimation, where the object represents the entire 3D scene.

For sufficiently textured objects, pose estimation is, more or less, considered to be solved. Solutions are based on the work of Lowe [17], where objects are represented by a sparse set of features. With the arrival of depth cameras, solutions have been developed for texture-less objects [7] or difficult scenes with repetitive or texture-less structures [23]. Many of these approaches are based on machine learning. In this work, we revisit the scenario of having RGB information only, and show that we can transfer the success story of machine learning-based approaches to this setting. This is especially interesting for practical applications, since many mobile devices are only equipped with a single RGB camera. Furthermore, current consumer depth sensors do not work with direct sunlight and reflective materials. Fig. 1 shows a result of our system, with augmented reality. This is a challenging case and the state-of-the-art method of [13], which operates on an RGB-D image, is not able to get a visually pleasing result. We show experimentally that our method can deliver a visually pleasing quality in 74% of test cases. Our approach surpasses competing RGB-based systems. Furthermore, an RGB-D variant of our system exceeds the state-of-the-art for object pose estimation and is on-par for camera localization.

Figure 1. (Left) Result of our method. The pose of the lamp is estimated sufficiently well for augmented realty. (Right) Result of a state-of-the-art system [13] that uses an RGB-D input image. The pose is less well suited for augmented reality.

Our method is inspired by the learning-based approaches [2] and [29], which achieve state-of-the-art results for RGB-D images. The idea is to utilize an intermediate repre-

sentation for objects, known as object coordinates, which is a dense, continuous object-part labeling. Object coordinates and object labels are jointly regressed for every pixel in the input image. This allows us to deal with multiple objects within one regressor. Object coordinates and labels are required at different processing stages, and instead of storing one single, optimal estimate we model their approximate distributions. We will see that this is crucial for applying our technical contributions, and achieving good results. The idea of modeling object coordinate distributions has also been explored in the recent work [29], in the context of camera localization from a single RGB-D image. We go beyond their work in various aspects. Firstly, their approach cannot directly be used in an RGB setting, since it relies on the calculation of pixel coordinates in camera space. We solve this problem by efficiently marginalizing object coordinate distributions along the pixel ray. Our approach is suitable for mixtures of anisotropic Gaussians and incorporates perspective effects. Secondly, by exploiting the uncertainty of object labels we are able to process any number of objects, known to the system, with a fixed budget of RANSAC hypotheses. This results in an efficient and scalable system with respect to the number of objects. A core part of our pipeline is an efficient regressor which iteratively reduces the uncertainty of object coordinate and class label predictions. Previously, a standard random forest has been used for this task [23, 29, 2, 14, 13]. Our approach builds upon the insight that the dense object coordinates contain a substantial amount of "structural" information, i.e. neighboring object coordinate predictions are statistically dependent. This makes it ideal for building the prediction step into an auto-context framework [28]. However, directly using an auto-context random forest hampered test performance, due to noisy outputs. We demonstrate that a new robust regularization of the multi-dimensional, dense labeling gives a major boost in performance.

Contributions. We give four contributions:

1. We present a generic 6D pose estimation system for both object instances and scenes (textured or textureless), which only needs a single RGB image as input. Our approach exceeds the state-of-the-art.

2. In order to deal with the missing depth information of the sensor, we present an efficient way to marginalize the object coordinate distributions over depth.

3. By exploiting label uncertainty we are able to process multiple object instances jointly with a fixed budget of RANSAC hypotheses.

4. A new robust (L_1-loss) regularization of the multi-dimensional, continuous output, such as object coordinates and labels, is crucial to get good performance with our auto-context regression forest approach.

We believe that the technical contributions, items $2-4$, are relevant for various related research projects. The question of how to deal with missing depth information, i.e. item 2, is of general interest to all works that deal with object coordinates, e.g. [2, 14, 23, 6, 29, 26], and would like to use an RGB camera. The idea of applying a fixed-budget RANSAC for multi-object estimation, i.e. item 3, is of interest to all methods where the final optimization depends on variables (explicitly or latent) such as object instance [2], person height [25] or object part [19]. Our L_1-loss regularization of object coordinates, i.e. item 4, is of general interest to other auto-context regression frameworks, *e.g.* auto-context Hough forests [11].

2. Related Work

Object Instance Pose Estimation. For instances with sufficient texture, accurate pose estimation from RGB images has been shown by matching sparse features [17, 5, 23]. Recently, research focused on texture-less objects and RGB-D images [7, 22, 27, 2]. Hinterstoisser *et al*. were especially successful with LINEMOD [7], a template approach that relies on stable gradient and normal features. It was recently further improved by discriminative learning [22]. However, pose estimation from RGB images was not presented in [7] nor [22]. In the robotics community some approaches exist, *e.g.* [33], with the goal to grasp objects which requires medium accuracy. Some authors presented pose estimation for both textured and texture-less objects [27, 13, 32, 15, 1] but always based on RGB-D footage. Shotton *et al*. [23] introduced the concept of scene coordinates for camera localization. They learn a mapping from camera coordinates to world coordinates, and then fit a rigid body transform. This work has been extended in [29] by using full distributions of scene coordinates for refinement. However, the approach cannot be applied to RGB images because no camera coordinates can be calculated. Brachmann *et al*. [2] extended the scene coordinate framework by jointly predicting object labels and object coordinates. However, their pipeline also depends heavily on the depth channel. In [31], a CNN was shown to learn a descriptor which distinguishes object poses from each other for RGB-D and RGB images. However, its performance was not demonstrated in a full pose estimation pipeline. Kendall *et al*. [10] trained a CNN to directly regress the 6D pose of a scene from an RGB image but with moderate accuracy. In contrast to the approaches above we achieve accurate 6D pose estimation from RGB only.

Auto-Context Framework. Auto-context has been introduced in the work of Tu and Bai [28]. The idea is to train stacked classifiers where early classifiers provide input for subsequent classifiers, resulting in a substantial gain in performance. Auto-context has been widely used for segmentation [28, 24, 20, 12], detection [11], and human pose estimation [21, 3], *etc*. Montillo *et al*. [20] presented an auto-

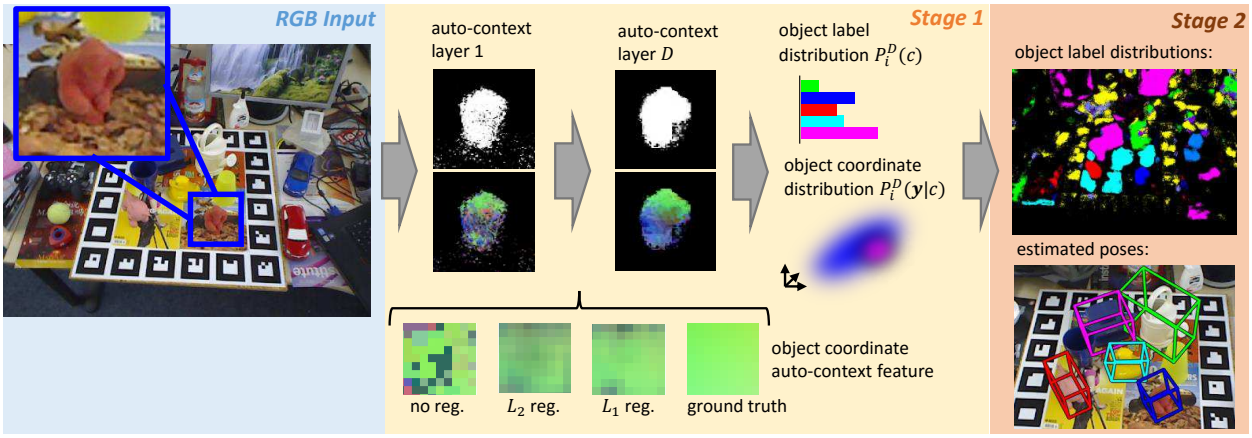

Figure 2. An RGB image is processed by our pipeline (left to right). We visualized two, out of three, stages of our method. In the first stage, the auto-context random forest processes the entire image to predict object labels (shown as probabilities) and object coordinates — zoom in on ape. The key to make the auto-context forest work well is to apply L_1-loss regularization to the multidimensional data, after each layer. The output of the first stage are pixel-wise distributions of object labels and object coordinates. In the second stage, the distributions over object labels are used to sample hypotheses for all objects at once. Preliminary pose estimates are found with pre-emptive RANSAC. In the third stage, these poses are refined using the object coordinate distributions.

context variant where a random forest has access to its own intermediate predictions for increased efficiency. In [12], this was extended by smoothing the intermediate predictions. This couples predictors of neighboring pixels, resulting in locally consistent output. We extend this idea further to multi-dimensional continuous data. Auto-context features usually access preliminary probabilities of discrete classes. Kontschieder *et al*. [11] showed that auto-context can also be deployed for intermediate multi-dimensional continuous output in a Hough forest for pedestrian detection, but they did not smooth the intermediate output. In this work, we show that robust smoothing of the multi-dimensional continuous object coordinates is essential for good performance.

Multi-Object RANSAC. The simultaneous detection of multiple objects (*e.g*. planes) in data with many outliers is an active field of research [34, 9, 18]. However, these methods try to fit multiple objects to points in the same coordinate space. While we similarly wish to find multiple objects within the same image, we have a regressor that predicts points in a separate coordinate space for each object. The fitting takes place within these separated spaces. One straight-forward solution for this task is to iterate through all objects and search for the best solution in each object coordinate space, as it has been done in [2, 23]. In contrast to this sequential procedure, we show how RANSAC can efficiently process multiple objects at once according to the evidence in the image.

3. Method

Our method consists of three individual stages, which are conceptually similar to [2]. In the *first stage*, a random forest predicts object labels and object coordinates jointly for every pixel of the input image (Sec. 3.1). In order to reduce the uncertainty of the predictions as much as possible, we extend the random forest to an auto-context random forest (Sec. 3.2) with L_1-loss regularization. In the *second stage*, we estimate the poses of multiple objects from the predicted 2D-3D correspondences using pre-emptive RANSAC guided by the uncertainty in object labels (Sec. 3.3). Finally, in the *third stage*, the poses are refined by exploiting the uncertainty of object coordinate predictions (Sec. 3.4). Fig. 2 shows an overview of our pipeline.

3.1. Joint Classification-Regression Forest

In this subsection, we briefly describe our classification-regression forest, which is a variant of recent works in this area [23, 2]. For each object $c \in \mathbb{N}$, present in the image, we aim at estimating the 6D pose $H_c = [R_c|\mathbf{t}_c]$ (*i.e*. rigid-body transformation). Here, H_c maps a 3D coordinate in object space to a 3D coordinate in camera space. The set $\mathcal{C} \subseteq \mathbb{N}$ contains all objects known to the system, including background. To make this task simpler, we employ a random forest that predicts for each pixel i in the image, the distribution over object labels, i.e. $P_i(c)$. Furthermore, given an object c and pixel i, we also want to predict a distribution of coordinates in object space $\mathbf{y} \in \mathcal{Y} \subseteq \mathbb{R}^3$, namely $P_i(\mathbf{y}|c)$. We train both distributions jointly using one classification-regression forest, whose suitability for this problem has been shown in previous works, e.g [2].

Testing. A pixel i is passed down a tree T of forest $\mathcal{T}$, until it reaches a leaf node. For feature tests, we use standard pixel-value difference in RGB. This is similar to previous works [24, 23, 2], but we omit normalization of feature-

scale by depth. Each leaf stores distributions $P_T(\mathbf{y}|c)$ and $P_T(c)$. Predictions of individual trees are merged as follows

$$P_i(c) = \frac{\prod_{T\in\mathcal{T}} P_T(c)}{\sum_{c'\in\mathcal{C}} \prod_{T\in\mathcal{T}} P_T(c')}. \quad (1)$$

This results in a high contrast soft segmentation. Object coordinate distributions $P_T(\mathbf{y}|c)$ are averaged.

Training. During training, features are selected which have the highest information gain of the joint distribution $P(\tilde{\mathbf{y}}, c)$, where $\tilde{\mathbf{y}}$ are quantized object coordinates, serving as proxy classes. The quantization is done by nearest neighbor assignment to random cluster centers (we choose random object coordinates in the training data). The use of information gain and proxy classes has shown to be superior to unimodal regression scores in [26]. Each leaf stores the empirical distributions $P_T(\mathbf{y}|c)$ and $P_T(c)$ of training samples reaching this particular leaf. The distribution of object coordinates is stored as a Gaussian-Mixture model (GMM),

$$P_T(\mathbf{y}|c, l) = \sum_{(m,\boldsymbol{\mu},\Sigma)\in\mathcal{M}} m\mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \Sigma), \quad (2)$$

where $\mathcal{M}$ is the set of mixture components, found via mean-shift. We calculate the support weight m, the mean $\boldsymbol{\mu}$ and full covariance Σ for each mode. We dismiss modes with weights $m < 50\%$ of the highest weight in $\mathcal{M}$.

3.2. Object Coordinate Auto-Context

We now describe the extension of the random forest to an auto-context random forest, and in particular the efficient use of regularized object coordinates as a feature. Instead of one forest, we train a stack of forests $\mathcal{T}^d$, where $d \in \{0, \ldots, D\}$ denotes the stack level. The first level, $\mathcal{T}^0$, is trained exactly as described above. All subsequent forests $\mathcal{T}^{d+1}$ have access to the output of the previous forest $\mathcal{T}^d$, namely object probabilities $P_i^d(c)$ and object coordinate predictions $P_i^d(\mathbf{y}|c)$ of pixel i. Inspired by the "Geodesic Forests" approach [12], we enforce the coupling of outputs of neighboring pixels by smoothing the predictions before passing them to the next forest. In [12] a geodesic filter for object probabilities was used, because in their application a gradient in the input signal was often a strong indication for an object boundary. We do not observe this correlation in our application scenario, since strong gradients can very well appear within the same object. Instead, we deploy a median filter in a local neighborhood of each pixel. Thus, we define median-smoothed object probabilities $P_i^d(c)$ as

$$g_{\mathcal{C}}^d(c, \mathbf{p}_i) = \operatorname*{argmin}_{\tilde{P}_i\in\mathbb{R}} \sum_{j\in\mathcal{N}_i} |P_j^d(c) - \tilde{P}_i|, \quad (3)$$

where $\mathbf{p}_i$ is the position of pixel i and $\mathcal{N}_i$ is a small neighborhood around pixel i. The definition of a feature for this output, to be used in forest $\mathcal{T}^{d+1}$, is straight forward, $f_{\mathcal{C}}(\mathbf{p}_i, \boldsymbol{\theta}) = g_{\mathcal{C}}^d(c, \mathbf{p}_i + \boldsymbol{\delta})$. The parameter vector $\boldsymbol{\theta}$ consists of pixel offset $\boldsymbol{\delta}$ and the object index c. We define the smoothing of the object coordinate prediction $P_i^d(\mathbf{y}|c)$ in a similar fashion. The median filter is robust to outliers since it optimizes the L_1-loss. This property is crucial for the object coordinate prediction as well, since outliers are very likely to occur (see Fig. 2). If the local smoothing is not robust, outliers will have a strong influence on the result. Unfortunately, the median filter is not directly applicable to data with dimensionality larger than one. However, the optimum under L_1-loss can be calculated in any Euclidean space, resulting in the *geometric median*. Thus, similar to Eq. 3, we define our regularized object coordinate output as

$$\mathbf{g}_{\mathcal{Y}}^d(c, \mathbf{p}_i) = \operatorname*{argmin}_{\mathbf{y}\in\mathcal{Y}} \sum_{j\in\mathcal{N}_i} \sum_{T\in\mathcal{T}^d} ||\boldsymbol{\mu}_{j,T} - \mathbf{y}||_2, \quad (4)$$

where $\boldsymbol{\mu}_{j,T}$ is the mean with the highest mixture weight of distribution $P_{j,T}^d(\mathbf{y}|c)$ for tree T at pixel j. Hence, $\mathbf{g}_{\mathcal{Y}}^d(c, \mathbf{p}_i)$ is a spatial smoothing, but also a combination of the predictions of trees $T \in \mathcal{T}^d$. Note that the geometric median minimizes the sum of distances as opposed to the sum of squared distances, hence we optimize a L_1-loss. To calculate $\mathbf{g}_{\mathcal{Y}}^d(c, \mathbf{p}_i)$ we use the iterative algorithm of [30]. We define the following feature on the smooth object coordinate output: $f_{\mathcal{Y}}(\mathbf{p}_i, \boldsymbol{\theta}) = [\mathbf{g}_{\mathcal{Y}}^d(c, \mathbf{p}_i + \boldsymbol{\delta})]_j$. The parameter vector $\boldsymbol{\theta}$ consists of offset $\boldsymbol{\delta}$, object index c, and dimension index $j \in \{x, y, z\}$. Function $[\mathbf{y}]_j$ returns the entry of $\mathbf{y}$ in dimension j. The training of forests $\mathcal{T}^d, d \in \{1, \ldots, D\}$ adheres to the same procedure as described in Sec. 3.1, but the set of feature types is increased by $f_{\mathcal{C}}$ and $f_{\mathcal{Y}}$.

3.3. RANSAC Pose Sampling

In the second stage of our pipeline, we efficiently find a preliminary pose for all objects present in the image. These poses are refined in stage three (Sec. 3.4). We first describe a standard procedure for pose estimation of a single object c', which is a combination of the pre-emptive RANSAC of [23] and the hypotheses sampling schema of [2]. Then, we formulate a new approach for handling multiple objects at once with a fixed budget of RANSAC hypotheses.

Single Object RANSAC. The auto-context forest predicts for each pixel i the object label distribution $P_i^D(c)$ (where D is the last level), and the 2D-3D correspondences $(\mathbf{p}_i, P_i^D(\mathbf{y}|c))$, *i.e.* the pixel position $\mathbf{p}_i$ and the uncertain object coordinate $\mathbf{y}$. We start by approximating the distributions $P_i^D(\mathbf{y}|c')$ by its main modes $\{\boldsymbol{\mu}_{i,T}|T \in \mathcal{T}^D\}$. As before, $\boldsymbol{\mu}_{i,T}$ denotes the mean with highest mixture weight of $P_{i,T}^D(\mathbf{y}|c')$ of tree T. Thus, the correspondences simplify to $(\mathbf{p}_i, \boldsymbol{\mu}_{i,T})$. We now define the re-projection error as $||\mathbf{p}_i - CH_{c'}\boldsymbol{\mu}_{i,T}||_2$ where C is the camera matrix, and $H_{c'}$ is the pose we are searching. We assume normalization of the homogeneous vector before calculating the L_2-norm. A

correspondence is an inlier when the re-projection error is below τ_{in}. At this stage, we aim at finding the pose which maximizes the inlier count:

$$H^*_{c'} = \underset{H_{c'}}{\operatorname{argmax}} \sum_{i \in \mathcal{W}(H_{c'})} \sum_{T \in \mathcal{T}^D} \mathbb{1}[||\mathbf{p}_i - C H_{c'} \boldsymbol{\mu}_{i,T}||_2 < \tau_{in}], \tag{5}$$

where function $\mathbb{1}[\cdot]$ is 1 if statement is true, otherwise 0. We restrict the calculation of inliers to window $\mathcal{W}(H_{c'})$, which is the projection of the 3D bounding box of object c'.

The objective function is maximized using locally optimized pre-emptive RANSAC [23]: Firstly, we draw a set of n_H pose hypotheses. A hypothesis $H_{c'}$ is drawn by choosing four correspondences $(i, \boldsymbol{\mu}_{i,T})$ and solving the perspective-n-point problem (PnP) [4, 16]. We draw the four initial pixels locations according to probability $P_i^D(c')$, which is an un-normalized distribution over all locations i in the image. Furthermore, pixels 2-4 are drawn within a box centered on the first pixel, which depends on the object size (see the supplement for details). The tree T from which to draw $\boldsymbol{\mu}_{i,T}$ is chosen randomly. In an initial validity check, a hypothesis is dismissed and redrawn if the re-projection error of the initial 4 pixels is too large. For all valid hypotheses, we draw a batch of n_B pixels within their respective windows $\mathcal{W}(H_{c'})$, and calculate the number of inliers. The hypotheses are ranked according to their inlier count and the lower half is dismissed. The remaining hypothesis are coarsely refined by re-solving PnP on the increased inlier set. This is repeated with additional batches of pixels drawn in each iteration. Finally, one hypothesis remains, which we use as preliminary pose estimate for object c', denoted by $\tilde{H}_{c'}$ with inlier set $\mathcal{I}(\tilde{H}_{c'})$.

Multi-Object RANSAC. Often, it is unknown which objects are present in the given image. The procedure above could be repeated for all objects $\mathcal{C}$ and the final hypotheses could be filtered based on inlier count. However, this scales poorly when the number of potential objects $|\mathcal{C}|$ becomes large. We present now a new method for processing all objects at once, within the pre-emptive RANSAC framework. Instead of drawing n_H hypotheses for each object independently, we draw one *shared* set of hypotheses. During the sampling of a hypothesis we decide on the fly which object it belongs to. This decision is based on the object label prediction of the first pixel sampled. In detail, each hypothesis is created as follows: Assuming background as $c = 0$ we draw the first pixel location i according to $\sum_{c=1}^{|\mathcal{C}|} P_i^D(c)$, which is an un-normalized distribution over i. Then we draw the object index c' of the local distribution at pixel i, i.e. $P_i^D(c)$. The remaining three pixel positions are drawn according to the un-normalized distribution $P_i^D(c')$, subject to the bounding box constraint mentioned above. Finally, we perform pre-emptive RANSAC separately for each object which was elected by at least one hypothesis. Note that no hypotheses will be drawn for objects without sufficient evidence in the image. Furthermore, each hypothesis has to pass the initial validity check, which requires some consistency of the object coordinate predictions drawn.

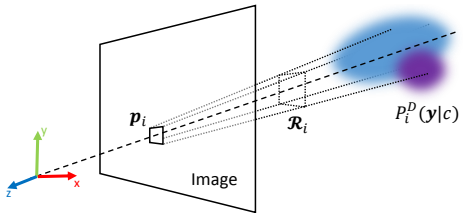

Figure 3. During pose refinement we need to marginalize the 3D object coordinate distribution $P_i^D(\mathbf{y}|c)$ over depth. We show that an approximation can be computed in closed-form for the projection pyramid at position $\mathbf{p}_i$. This is done by integrating along the ray (dashed line) while taking perspective effects into account.

3.4. Pose Refinement

The preliminary poses $\tilde{H}_c$ of RANSAC minimize the (truncated) re-projection error of a set of inlier correspondences $(i, \boldsymbol{\mu}_{i,T})$, see Eq. (5). Note that in the objective function each of the correspondences is treated with equal weight. However, we have access to full distributions $P_i^D(\mathbf{y}|c)$, which model uncertainty information in the object coordinate prediction. We aim to exploit this information for pose refinement, which is the third stage in our pipeline. To achieve this, we now introduce a novel procedure for efficiently computing the (approximated) marginalized object coordinate distribution in the image.

The basic idea for pose refinement is that we want to maximize the pose probability under the distributions $P_i^D(\mathbf{y}|c)$. In [29], this idea was explored by scoring hypotheses based on the log-likelihood of inlier correspondences:

$$H^*_c = \underset{H_c}{\operatorname{argmax}} \sum_i \log P_i(H_c^{-1}\mathbf{e}_i|c), \tag{6}$$

where $\mathbf{e}_i$ is the camera coordinate of pixel i. Note, the transformation $H_c^{-1}\mathbf{e}_i$ yields an object coordinate. However, unlike [29], we do not have access to depth information, and thus cannot recover $\mathbf{e}_i$. We only know that the true $\mathbf{e}_i$ must lie within the projection volume of pixel i (see Fig. 3). We denote this volume by $\mathcal{R}_i$. We can now substitute the likelihood of $\mathbf{e}_i$ in Eq. (6) with the probability of $\mathcal{R}_i$:

$$\begin{aligned} H^*_c &= \underset{H_c}{\operatorname{argmax}} \sum_{i \in \mathcal{I}(\tilde{H}_c)} \log P^D_{\mathcal{R}_i}, \text{where} \\ P^D_{\mathcal{R}_i} &= \iiint_{\mathcal{R}_i} P_i^D(H_c^{-1}[x,y,z]^T|c)\, dx\, dy\, dz\ . \end{aligned} \tag{7}$$

For robustness, we calculate the likelihood only over the inlier set $\mathcal{I}(\tilde{H}_c)$ of the current, preliminary pose $\tilde{H}_c$. We now explain how to approximate $P^D_{\mathcal{R}_i}$ efficiently. Instead of integrating over all dimensions, we integrate along the ray cast from the camera origin to the pixel center (see Fig. 3). Since the shape of the volume is a pyramid, we add a quadratic factor during integration. Because we would like to integrate in camera coordinate space, we first transform the Gaussian mixture components of $P^D_i(\mathbf{y}|c)$ (which is defined in terms of object coordinate space) to camera coordinates using $H_c = [R_c|\mathbf{t}_c]$. Furthermore, we apply rotation R_{xy} which maps the pixel with position $\mathbf{p}_i$ to $(0,0)^T$. Thus, we transform the mean of each mixture component to $\boldsymbol{\mu}_\mathbf{e} = R_{xy}(R_c\boldsymbol{\mu} + \mathbf{t}_c)$ and the covariance matrices to $\Sigma_\mathbf{e} = R_{xy}(R_c\Sigma R_c^T)R_{xy}^T$. After these transformations, the pixel ray aligns with the z-axis, keeping x and y constant in the integral. This allows us to approximate

$$P^D_{\mathcal{R}_i} \approx \sum_{(m,\boldsymbol{\mu},\Sigma)\in\mathcal{M}^D_i} m \int_{-\infty}^{\infty} z^2 \mathcal{N}([0,0,z])^T; \boldsymbol{\mu}_\mathbf{e}, \Sigma_\mathbf{e})\, dz \tag{8}$$

where a closed form solution exists for the integral. Note that the factor z^2 is crucial since volume $\mathcal{R}_i$ is a pyramid with the camera origin as its tip. By plugging this approximation into Eq. (7) and optimizing, we are able to refine preliminary poses $\tilde{H}_c$, to yield our final pose estimates H^*_c.

4. Experiments

We evaluate our work on three publicly available data sets. First we demonstrate pose estimation performance for a single object given an RGB image (Sec. 4.1). Here, we also report results for RGB-D input images. Next, we evaluate our system with respect to detection of multiple objects (Sec. 4.2). Finally, we consider the scenario of camera localization (Sec. 4.3).

4.1. Single Object Pose Estimation

Hinterstoisser *et al*. [7] published an RGB-D data set of texture-less objects in a cluttered scene (see Fig. 2 for an example). Each image of the data set is annotated with the ground truth 6D pose of one object and the ID of this object is assumed to be known. Colored 3D models of the objects are available for the generation of training images. However, learning from synthetic images is beyond the scope of this work. Instead, we follow the setup of Brachmann *et al*. [2], and apply various methods to this setting for comparison. For the sake of comparability with these methods we only report results for 13 objects (out of 15) where proper 3D models exist. Each object sequence is split in training and test data. Training images are selected such that the associated object poses have a minimum angular distance of 15°. Doing so selects $\approx 15\%$ of the images for training. The remaining images serve as test set. We segment training images so that the scene context cannot be learned. During training, we sample patches with random scales according to an object distance between 65cm and 115cm (the range given for training in [7]). For pose estimation from RGB we simply omit the depth channel of each test image.

Parameters. We train 3 trees (max. depth 64) per auto-context layer and 3 layers in total. The inlier threshold is set to $\tau_{in} = 3$px and we sample $n_H = 256$ hypotheses during pose optimization. A complete list of parameters and details on the implementation can be found in the supplement.

Metrics. We measure the percentage of images where the object pose was estimated correctly. Different measures have been proposed in the past. Hintertoisser *et al*. [7] define a threshold on the average distance of transformed 3D points. The exact tolerance to translational and rotational error depends on object size and shape. Shotton *et al*. [23] define this tolerance explicitly and accept a pose if the error is below $5cm$ and 5°. While appropriate for some applications, these two measures are not very well suited when applying visual effects to the 2D image, *e.g*. augmented reality. For example, pose accuracy in z direction is far less important for the visual impression than precision in x and y. Therefore, we additionally propose the following measure (see the supplement for a formal definition). We project the object model into the image using the ground truth pose and the estimated pose. We accept the estimated pose, if the average re-projection error of all model vertices is below 5px (we call this *2D Projection*). See Fig. 1 for a comparison of metrics. With the measure of Hinterstoisser *et al*. both results are correct, but only the left result is correct with the 2D projection measure. Finally, to evaluate 2D detection performance, we calculate the 2D bounding box overlap and accept it if the intersection over union (IoU) > 0.5 (we call this *2D Bounding Box*).

4.1.1 RGB Setting

Baselines.[1] The template-based approaches LINEMOD (for RGB-D images) and LINE2D (for RGB-images) have been introduced in [8] for object detection[2]. LINEMOD was extended to perform pose estimation in [7]. We created a variant of [7] based on LINE2D for pose estimation from RGB images. Of the various post-processing steps of [7] we apply only the color check because the other steps are based on depth.

[1] We were not able to compare to the method of [22]. The authors were not able to provide us source code or binaries, or to run the experiment for us. This method extends [8, 7] by discriminative learning of templates.

[2] Unfortunately, implementations of LINEMOD and LINE2D [8, 7] are unavailable. We used the code in OpenCV, which was optimized for synthetic images. To make it work well for real images we activated sampling of strong gradients in the object interior. We tested the detection performance of our LINE2D on the data set of [22] and achieve same results as in [22]. Testing on [8] is not possible due to unavailable data.

Table 1. Pose estimation results on the data set of [7] for a single object where the object ID is known in advance. AC means auto-context.

	RGB					RGB-D		
	Ours L_1 reg.	Ours L_2 reg.	Ours w/o reg.	Ours w/o AC	LINE2D[8]	Ours L_1 reg.	Krull *et al.* [13]	Brachmann *et al.* [2]
2D Projection	**73.7%**	68.6%	38.0%	59.3%	20.9%	**95.7%**	82.6%	81.7%
2D Bounding Box	**97.5%**	97.1%	90.3%	96.2%	86.5%	**99.6%**	98.8%	99.1%
6D Pose (Metric of [7])	**50.2%**	46.0%	19.6%	30.1%	24.2%	**99.0%**	93.9%	97.4%
6D Pose (5cm 5°[23])	**40.6%**	34.1%	11.0%	22.6%	8.1%	**82.1%**	73.1%	52.1%

Figure 4. Pose estimation from an RGB image. (Left) Four objects, partially overlaid with 3D models with the estimated pose. (Right) Detecting multiple objects at once. Six out of ten objects have been detected (We accept hypotheses with at least 400 inliers). The bounding box color encodes the object ID.

Results. Results are shown in the left half of Table 1. In 73.7% of test images our approach delivers an accuracy which is suitable for visual effects. See Fig. 4 for qualitative results. The high accuracy is reflected in low median errors for translations and rotation, *i.e.* $2.3cm$ and $5.9°$. The translational error occurs predominantly in z-direction. Our final refinement step, using uncertainty (see Sec. 3.4), helps reducing this error. Without this step we loose 4.2% with the 2D projection measure and 17.9% with the measure of Hinterstoisser *et al*.

Our auto-context (AC) framework boosts performance substantially *e.g*. by 14.4% with the 2D projection measure (see *Ours w/o AC* in Table 1). We observe that regularization of the intermediate auto-context feature channels was absolutely essential. Omitting this step leads to unstable results and performance was actually worse than omitting auto-context altogether (see *Ours w/o reg.*). Using L_2 regularization, compared to L_1, results in a loss of 5.1%.

LINE2D detects objects relatively well (see *2D Bounding Box* score) but fails to reliably estimate the correct poses. Without depth information it relies mostly on gradients on the object silhouette. This makes accurate estimation of rotation very difficult.

4.1.2 RGB-D Setting

Our pipeline can be easily altered to make use of a depth channel. In this case, forest features can be depth-normalized, perspective-n-point is substituted by the Kabsch algorithm, inliers are defined in 3D camera space, and final refinement is based on camera coordinates as in [29].

Baselines. We compare to the state-of-the-art RGB-D pose estimation pipeline of Brachmann *et al*. [2]. We used the publicly available binaries to measure accuracy with our metrics. We also compare to the very recent method of Krull *et al*. [13] which is an extension of the pipeline of Brachmann *et al*. [2] by utilizing a CNN in the final pose optimization stage.

Results. Results are shown in the right half of Table 1. With the measure of Hinterstoisser *et al*. [7] multiple methods approach the limit of 100% correctly estimated poses. The measure of Shotton *et al*. [23] is more sensitive with respect to rotation. With this measure, our approach estimates 82.1% of poses correctly, which is considerably more than Krull *et al*. (-9%) or Brachmann *et al*. (-30%). We attribute this to the robust inlier-based pose optimization which is different from the general purpose CNN in [13] and the hand-crafted energy of [2]. In particular the energy in [2] has a higher emphasis on model fitting, while we rely on the discriminative power of the random forest. In the RGB-D setting, the use of auto-context results only in a small improvement (+2.4% with L_1 reg. under 5cm 5°, +1.9% with L_2 reg., and -8.9% w/o reg.). With the 2D projection measure, our approach estimates 95.7% of poses correctly, which is again a large improvement compared to the baselines (+13.1%, +14% respectively). This confirms the suitability of our approach for applying visual effects.

4.2. Multi-Object Detection

The previous experiments performed pose estimation of a single object. We now evaluate the detection performance of our approach, *i.e*. we do not know upfront which objects are present in a test image. We use the data set published by Brachmann *et al*. [2]. The authors annotated one image sequence of the data set of Hinterstoisser *et al*. [7] with the poses of all 9 objects present. The amount of occlusion can be very large making this data set extremely challenging.

We search for all 13 objects (even those not present) in all images and keep the strongest response per object. We accept a response if the IoU is at least 0.5. We rank all results according to response score (inlier count for our method and matching score for LINE2D), and plot precision vs. recall (see Fig. 5, left). We compare two variants of our method: Firstly, we divide the budget of 256 RANSAC hypotheses evenly among objects (*Ours w/o sharing*). Sec-

ondly, we apply the method described in Sec. 3.3, *i.e.* we sample hypotheses according to the object label distribution (*Ours w/ sharing*). It is important to note that the 256 hypotheses have to be valid (see middle of Sec. 3.3). Hence, the run-time can vary depending on the difficulty of finding a valid hypothesis.

Results. LINE2D achieves 0.21 average precision (AP), due to its sensitivity to occlusion. The two variants of our method score 0.49 AP (*Ours w/o sharing*) and 0.51 AP (*Ours w/ sharing*). As a further experiment, we re-train the auto-context random forest with additional objects to a total of 25 and 50 objects. We repeat the experiment above and measure AP and processing time[3], see results in Fig. 5 (right). With more objects, the performance of the schedule w/o sharing drops more rapidly while its run-time increases. Processing time is predominately spent on searching for valid hypotheses where objects are occluded or missing. Sampling hypotheses according to the label distribution scales much better. RANSAC processes all 50 objects in ca. 1s. Note that our implementation runs on CPU and is not optimized. We expect a large boost for a GPU port.

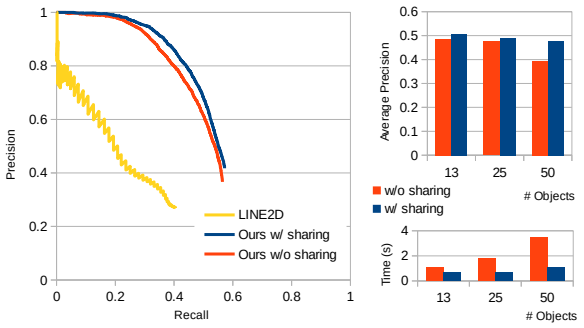

Figure 5. Detection experiment on the data set of Brachmann *et al.* [2]. (Left) Precision-recall plot for 13 objects. (Right) Average precision and run-time for increasing object count.

4.3. Camera Localization

Shotton *et al.* [23] published an RGB-D data set of 7 scenes with annotated camera poses. Multiple image sequences are given per scene, which were split by [23] in training and test set. We used the depth channel to calculate object coordinate labels for the training set. This labeling can also be rendered using the available 3D models. Otherwise, we ignored depth channels and estimate the camera pose from RGB only. We kept parameters largely unchanged with respect to Sec. 4.1, up to a few exceptions due to the large difference in object size (see the supplement).

Baselines. We compare to a state-of-the-art sparse-feature based approach from [23]. Furthermore, we compare to the very recent PoseNet [10] work, which is a CNN that directly regresses the 6D camera pose.

[3]We omitted final refinement. It had little impact on the detection performance but would have dominated run-time (ca. 100ms per object).

Table 2. Results on the 7 scenes data set using RGB only. The *Avg. Error* is calculated by averaging the median pose error per scene.

Pose (Scene Known)	5cm 5°	Avg. Error
Sparse RGB[23]	40.7%	-
PoseNet[10]	-	46.9cm, 5.4°
Ours	**55.2%**	**6.1cm, 2.7°**
Pose (Scene Unknown)		
Ours w/ sharing	**50.0%**	**8.5cm, 3.3°**
Ours w/o sharing	33.1%	15.0cm, 8.5°

Results. In a first experiment we follow Shotton *et al.* and assume that it is known, which of the 7 scenes is present in the test image (see top part of Table 2). In 55.2% of test cases we estimate the pose correctly, *i.e.* within the 5cm 5° threshold. This is an improvement of 14.5% over the sparse feature baseline. Our auto-context framework boosts performance by 3.2% and refinement using uncertainty by 1.0%. Compared to PoseNet, our results have a translational error that is one order of magnitude smaller. Applying the RGB-D variant of our pipeline, we have 88.1% of correct poses, which is on-par with the results of the state-of-the-art RGB-D method [29] (89.5%).[4]

In a second experiment, we estimate the pose and scene ID jointly from an RGB image (see bottom part of Table 2). We train one auto-context random forest for all scenes, and let RANSAC sample 256 object hypotheses per image, according to the object label distributions (see *Ours w/ sharing*). Despite the increased difficulty, we observe only a medium loss in performance, i.e. 5.2%. Furthermore, this is substantially better than evenly distributing the budget of hypotheses over the 7 scenes and adjusting parameters to achieve equal run-time (see *Ours w/o sharing*).

5. Conclusion and Future Work

We presented scalable object detection and 6D pose estimation from a single RGB image. The system is broadly applicable, ranging from small objects to entire scenes. It can be easily adapted to exploit a depth channel if available. It would be interesting to explore the limits of the approach w.r.t. the number of objects. Also, the auto-context forest could be trained in a way to be robust w.r.t. occlusion, which we did not consider in this work.

Acknowledgements: This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 647769).

[4]We observed no improvement when applying auto-context in the RGB-D setting, although the intermediate prediction quality of the forest increased.

References

[1] L. Bo, X. Ren, and D. Fox. Unsupervised Feature Learning for RGB-D Based Object Recognition. In *ISER*, June 2012. 2

[2] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *ECCV*, 2014. 1, 2, 3, 4, 6, 7, 8

[3] M. Dantone, J. Gall, C. Leistner, and L. Van Gool. Human pose estimation using body parts dependent joint regressors. In *CVPR*, 2013. 2

[4] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *TPAMI*, 2003. 5

[5] I. Gordon and D. Lowe. What and where: 3d object recognition with accurate pose. In *Toward Category-Level Object Recognition*. 2006. 2

[6] A. Guzman-Rivera, P. Kohli, B. Glocker, J. Shotton, T. Sharp, A. Fitzgibbon, and S. Izadi. Multi-output learning for camera relocalization. In *CVPR*, 2014. 2

[7] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *ACCV*, 2012. 1, 2, 6, 7

[8] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, 2011. 6, 7

[9] M. Jaberi, M. Pensky, and H. Foroosh. Swift: Sparse withdrawal of inliers in a first trial. In *CVPR*, 2015. 3

[10] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 2, 8

[11] P. Kontschieder, S. R. Bulo, A. Criminisi, P. Kohli, M. Pelillo, and H. Bischof. Context-sensitive decision forests for object detection. In *NIPS*, 2012. 2, 3

[12] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi. Geof: Geodesic forests for learning coupled predictors. In *CVPR*, 2013. 2, 3, 4

[13] A. Krull, E. Brachmann, F. Michel, M. Y. Yang, S. Gumhold, and C. Rother. Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *ICCV*, 2015. 1, 2, 7

[14] A. Krull, F. Michel, E. Brachmann, S. Gumhold, S. Ihrke, and C. Rother. 6-dof model based tracking via object coordinate regression. In *ACCV*, 2014. 2

[15] K. Lai, L. Bo, X. Ren, and D. Fox. A scalable tree-based approach for joint object and pose recognition. In *In Twenty-Fifth Conference on Artificial Intelligence (AAAI*, 2011. 2

[16] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o (n) solution to the pnp problem. *IJCV*, 2009. 5

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 1, 2

[18] L. Magri and A. Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *CVPR*, 2014. 3

[19] F. Michel, A. Krull, E. Brachmann, M. Y. Yang, S. Gumhold, and C. Rother. Pose estimation of kinematic chain instances via object coordinate regression. In *BMVC*, 2015. 2

[20] A. Montillo, J. Shotton, J. Winn, J. E. Iglesias, D. Metaxas, and A. Criminisi. Entangled decision forests and their application for semantic segmentation of ct images. In *IPMI*, 2011. 2

[21] V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh. Pose machines: Articulated pose estimation via inference machines. In *ECCV*, 2014. 2

[22] R. Rios-Cabrera and T. Tuytelaars. Discriminatively trained templates for 3D object detection: A real time scalable approach. In *ICCV*, 2013. 2, 6

[23] 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, 6, 7, 8

[24] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008. 2, 3

[25] M. Sun, P. Kohli, and J. Shotton. Conditional regression forests for human pose estimation. In *CVPR*, 2012. 2

[26] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The Vitruvian Manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, 2012. 2, 4

[27] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim. Latent-class hough forests for 3d object detection and pose estimation. In *ECCV*, 2014. 2

[28] Z. Tu and X. Bai. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *TPAMI*, 2010. 2

[29] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. S. Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *CVPR*, 2015. 1, 2, 5, 7, 8

[30] Y. Vardi and C.-H. Zhang. The multivariate l1-median and associated data depth. In *Proceedings of the National Academy of Sciences*, 2000. 4

[31] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, 2015. 2

[32] C. Zach, A. Penate-Sanchez, and M.-T. Pham. A dynamic programming approach for fast and robust object pose recognition from range images. In *CVPR*, 2015. 2

[33] M. Zhu, K. G. Derpanis, Y. Yang, S. Brahmbhatt, M. Zhang, C. Phillips, M. Lecce, and K. Daniilidis. Single image 3d object detection and pose estimation for grasping. In *ICRA*, 2014. 2

[34] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multi-ransac algorithm and its application to detect planar homographies. In *ICIP*, 2005. 3