Sparse Keypoint Models for 6D Object Pose Estimation

Emal Sadran\textsuperscript{1}  Kai M. Wurm\textsuperscript{2}  Darius Burschka\textsuperscript{1}

Abstract—In this paper, we present an approach to generate sparse object models for keypoint-based 6D object pose estimation. Keypoint-based object models usually consist of thousands of keypoints. Our approach generates sparse models by identifying and removing keypoints that are not relevant to the object localization. It applies data association to detect duplicate keypoints and applies statistical analysis to identify keypoints that have not been detected reliably during model generation. Our approach furthermore ensures that keypoints are well distributed across the volume of the object model. We evaluated our approach using a SIFT-based 6D object localization system on the basis of real world datasets. In our experiments, we achieved a reduction of the model sizes to approximately 1\% of the original model size without a substantial loss of localization performance.

I. INTRODUCTION

One of the fundamental tasks of any intelligent robot is to detect objects in the environment. This ability is crucial, for example, when a robotic household assistant performs tasks such as cleaning a room, fetching objects, or storing away groceries.

A popular method to detect objects and to estimate object poses is to extract local features from a sensor measurement and to match them to object models in a pre-trained database. From successful matches, a rigid transform is computed that corresponds to a 6D object pose estimate (defined as a 3D position and orientation).

In this paper, we focus on generating sparse object models for pose estimation from visual keypoints. Visual keypoints are defined as locations on the object surface that can be identified by a local visual feature descriptor, for example, SIFT [11], SURF [3], or ORB [13]. Throughout the paper, we assume that triples of matched keypoints are used to estimate object poses.

Previous approaches store up to 14,000 keypoints per object model [6]. When triples of visual keypoints are used to locate box-shaped objects, the theoretical minimum number of keypoints is 3*6=18 points (one triple per face). This minimum, however, is unrealistic since it assumes that each keypoint can be perfectly detected in every measurement. In real world scenarios, detection will not be perfect because of changing lighting conditions, sensor noise, object occlusion, or slight deformations and variations of the objects. Yet there is a difference of three orders of magnitude between previous models and the theoretical minimum.

The runtime of keypoint-based object recognition depends on the total number of keypoints in the object model database. With larger databases, a higher number of keypoints will be detected in the measurements. This, in turn, increases the number of possible keypoint triples. For this reason, sparse object models will reduce the runtime of object detection considerably.

In this paper, we present an approach to generate sparse object models for 6D object pose estimation. The key idea of our approach is to identify and remove keypoints that are not relevant to the object localization: First, we detect duplicate keypoints that result from merging partial views of the object. Second, we identify unstable keypoints that cannot be detected from a sufficiently large range of viewing angles. To achieve a spatially well balanced model, we finally sub-sample keypoints based on their distribution in the volume of the object model. We evaluated our approach using a SIFT-based 6D object localization system on the basis of real world datasets. In our experiments, we achieved a reduction of the model sizes to approximately 1\% of the original model size without a substantial loss of localization performance. An example of the output of our algorithm can be seen in Fig. 1.

The remainder of this paper is organized as follows: After discussing related work, we describe the object recognition approach we apply in our evaluation. In Sec. IV, we present the algorithm we use to register partial views of an object. Our approach to reduce initial object models to sparse models is given in Sec. V. Finally, our experimental evaluation is presented in Sec. VI.
II. RELATED WORK

A number of approaches have been proposed to detect objects from SIFT keypoint models [1], [2], [4], [6], [11]. The process of generating models, however, is often not discussed in detail. In this paper, we focus on generating sparse keypoint models. Collet et al. [4] generate sparse 3D models by using bundle adjustment and clustering. In contrast to our approach, they do not consider keypoint stability and the distribution of keypoints in the model. Grundmann et al. [6] apply nearest neighbors search to cluster keypoints and they remove small clusters from the final model. Arbeiter et al. [1] apply a fastSLAM approach to construct 3D object models. In their approach, models consist of keypoints that have been detected more than an empirically determined number of times. Compared to these previous approaches, our approach does not rely on fixed number of sightings but instead applies a threshold on the estimated range of viewing angles. We furthermore propose to apply spatial sub-sampling to address the distribution of model points on the object surface. Additionally, we provide an analysis of the effect that the presented techniques have on localization performance.

In our experiments, we take measurements using the RGBD sensor Microsoft Kinect. Recently, a number of approaches have been proposed to generate surface models with this sensor. The KinectFusion approach proposed by Newcombe et al. [12] is able to generate surface models of complex and arbitrary indoor scenes. Henry et al. [7] presented their RGB-D Mapping framework to generate dense surfel-based models of indoor environments. Weise et al. [16] and Ruhnke et al. [14] presented approaches to model single objects using surfels. An overview and evaluation of stereo reconstruction algorithms in general is given in [15]. The approaches mentioned above are able to produce consistent models of object surfaces. However, they do not address the problem of sparse model generation for object pose detection.

III. OBJECT POSE ESTIMATION

To generate object poses in our experiments, we apply a pose estimation approach that is similar to the approach presented by Grundmann et al. [6]. The basic computational steps are given in Alg. 1.

The algorithm is based on SIFT keypoints that are extracted from triangulated stereo images or RGBD measurements, e.g., from the Kinect sensor. In a first step, the SIFT keypoints are matched to a database $D$ of object models. For each object model $d \in D$, a maximum of $i$ hypotheses is generated. To generate hypotheses, three keypoints are chosen randomly from the set of keypoints that has been matched to model $d$. An object pose hypothesis is then computed from these triples of matched points using the approach proposed by Horn et al. [8]. Finally, pose hypotheses are clustered and outliers are removed using the RANSAC algorithm [5].

The runtime of the object localization approach primarily depends on the number of keypoints in the database. While the keypoint matching step can be efficiently implemented, e.g., using kd-trees, the number of possible keypoint triples is cubic in the number of matched keypoints. With dense object models, the number of wrong pose hypotheses (outliers) will in general increase and a high number of iterations $i$ can become necessary to localize objects reliably.

IV. MODEL ACQUISITION

The input to our modeling approach is a set $z = \{z_i\}$ of overlapping partial views of the object. We assume that each measurement consists of depth and texture information. Our approach does not require the camera poses to be known beforehand. Views can be recorded using a turntable, by manually turning the object in front of the sensor, or by moving the sensor around the object. Objects can be shaped arbitrarily but we assume them to exhibit a salient texture that is suited to extract SIFT keypoints.

Algorithm 1 6D Object Pose Estimation

Require:
- $z$, input measurement
- $D$, object database

Ensure:
- $H$, set of pose hypotheses

1: extract SIFT keypoints from $z$
2: match keypoints to database $D$ 
3: for all object models $d \in D$ do
4:  for $i$ iterations do
5:     randomly choose three keypoints matched to $d$
6:     compute object pose hypothesis from matches
7:  end for
8:  cluster pose hypotheses for object $d$
9:  add clustered hypotheses to $H$
10: end for

To construct a consistent model of an object, we transform all input data to a common reference frame. Since we assume
that camera poses are unknown, we first compute an estimate of all camera poses $c_i$ in a common reference frame. The basic steps of our registration approach are given in Alg. 2. First, we extract SIFT keypoints from all measurements $z_i$. These keypoints are then used to construct a set of geometrical constraints $\Gamma$ between pairs of measurements. This step is implemented similar to the object localization described above. Based on the set of constraints $\Gamma$, the camera poses $c_i$ are estimated by applying an optimization approach (we use the optimization framework g2o [10] in this step). An example of the result can be seen in Fig. 2.

Once an estimate of the camera poses has been generated, we construct a joint object model by transforming all extracted keypoints into a common reference frame. This object model is suited for object pose estimation using an optimization approach such as the method described in Sec. III. However, the object model is necessarily dense since it contains every keypoint that has been extracted during model acquisition. In the following section, we present techniques to remove duplicate and ineffective keypoints to construct sparse object models.

V. SPARSE MODEL GENERATION

Ideally, an object model consists of unique keypoints that are detected stably across different measurements. An ideal model is furthermore sparse and its keypoints are distributed well across the surface of the object. In the following, we present techniques to remove redundant or ineffective keypoints from an initial object model generated using the approach described in Sec. IV.

A. Data Association

The majority of redundant keypoints in the initial model result from multiple sightings during model construction. This redundancy cannot be avoided since we rely on overlapping partial views during data registration. In fact, multiple sightings of the same keypoint can be seen as a strong indicator of the stability of the keypoint, as we will discuss below.

We assume that keypoints which correspond to the same physical point on the model will be distributed in a local neighborhood with radius $r_{\text{DA}}$ in the model. This radius depends on the measurement noise and the registration error. In our experiments we found 3 mm to be a good value and it corresponds to the measurement noise reported for the Kinect device at approximately 1 m [9].

Often, clusters of keypoints will overlap and thus cannot be separated based on Euclidean clustering alone. For this reason, we furthermore require associated keypoints to have matching descriptors. Note that it is not sufficient to rely on descriptor matching alone since the same descriptor could be computed at different locations on the object surface.

In our approach, two keypoints are associated if they lie within a local neighborhood and their descriptors do not differ by more than $\epsilon_{\text{DA}}$ (in our experiments, we choose a value of $\epsilon_{\text{DA}} = 0.3$). Let $k_i = (p_i, d_i) \in k$ be keypoints with their corresponding 3D position and descriptor. Two keypoints $k_i$ and $k_j$ are associated if the following constraints hold:

$$|p_i - p_j| < r_{\text{DA}} \quad (1)$$
$$|d_i - d_j| < \epsilon_{\text{DA}}. \quad (2)$$

In the final object model, redundant keypoints do not improve localization and can therefore safely be ignored. Duplicate keypoints are removed from the final model by applying a clustering technique as described in Sec. V-C.

B. Keypoint Stability

The initial object model consists of every keypoint that was computed during data acquisition. Many of these keypoints have been detected from several camera positions, we refer to them as stable keypoints. Unstable keypoints, in contrast, could not be re-detected reliably. In previous approaches, unstable keypoints have been filtered by applying a threshold on the number of detections [1], [6]. This threshold, however, depends on the total number of views. It has to be adjusted whenever the method of acquiring a raw model is changed.

In our approach, we use the result of the data association to determine the set of views from which a cluster of keypoint was detected. We use the estimated camera origins of these views to determine the range of viewing angles. Given a set of associated keypoints $\{k_i = (p_i, d_i)\}$, let $C_d = \{c_i\} \subset C$ be the set of camera origins from which the keypoints were detected. The range of viewing angles $\alpha_d$ is then computed as the maximum angle between any two vectors pointing from a camera pose to the corresponding keypoint position:

$$\alpha_d = \max_{\alpha_{i,j}} \angle((p_i - c_i), (p_j - c_j)), \quad c_i, c_j \in C_d \quad (3)$$

Fig. 2. Estimated camera poses during one of our experiments. Measurements were taken while the object was rotated on a turntable. Camera poses are visualized as reference frames with the $z$ (forward pointing) axis depicted in blue.
An illustration of the angles $\alpha_{i,j}$ is given in Fig. 3. In 3D, $\alpha_d$ corresponds to the opening angle of a viewing cone.

To eliminate unstable keypoints from the final model, we apply a threshold $\alpha_{\min}$ on $\alpha_d$. See Sec. VI-B for a discussion on how to choose $\alpha_{\min}$.

### C. Local Clustering

To eliminate duplicate keypoints that resulted from multiple sightings, we apply local clustering as proposed by Collet et al. [4] and Grundmann et al. [6].

Each set of associated keypoints \( \{k_i = (p_i, d_i)\} \) is reduced to one representative. From Eq. 2, we know that within such a cluster the distance between descriptors $d_i$ is smaller than $\epsilon_{\text{DA}}$. To determine the cluster representative, we compute the mean position and the normalized mean descriptor. More complex representations of clusters would certainly be possible. In practice, however, this efficient method led to good recognition results. In the final model, the sets \( \{k_i\} \) are replaced by the representatives.

### D. Spacial Sub-sampling

To estimate object poses, groups of features are usually considered. For example, triples of keypoints are selected in Sec. III. In general, pose estimation will be less sensitive to measurement noise if the keypoints in these groups are well distributed on the observed surface. Groups of nearby keypoints, in contrast, are likely to lead to inferior pose estimates [2].

While the techniques above remove unstable and redundant keypoints, we now describe how to thin out inefficient clusters of keypoints. In our approach, we replace local clusters of keypoints by a few representative keypoints. To this end we apply spacial sub-sampling based on a 3D grid. The volume occupied by the model points is discretized into voxels (i.e., cubic sub-volumes of a given cube side length). This sub-sampling is performed after removing unstable keypoints from the model. Therefore we assume that all remaining keypoints are equally well suited for pose estimation. Of all keypoints in a voxel, we choose the keypoint that is closest to the center of the voxel as the representative of its cluster. The size of the voxels determines the amount of keypoints in the final model. An illustration of this process is given in Fig. 4.

### VI. Experiments

We evaluated our approach by modeling various real objects. The experiments are designed to show that our approach leads to object models that are sparse but that are well suited for object pose estimation.

We acquired models by placing objects on a turntable and then taking measurements with a Microsoft Kinect. The sensor was kept in a fixed position while objects were rotated. Each object was rotated through 360 degrees and an RGBD-measurement was taken every 10 degrees. Then, the objects were turned upside-down and the procedure was repeated. This process results in a total of 74 measurements per object (see Fig. 2 for a visualization).

#### A. Dataset

To evaluate the influence of the techniques proposed in this paper, we created initial model of three objects. The objects and their initial keypoint models are shown in Fig. 5. Additionally, test sets of 10 views per object were recorded. This test data was used to estimate object poses using the approach described in Sec. III.

Since ground truth object poses were not available, we determined baseline object poses using the initial (full) model database. The baseline pose is computed as the average result of 10 localization runs.

#### B. Keypoint Stability and Local Clustering

In this experiment, we analyzed the influence of the view angle threshold $\alpha_{\min}$ on the model size and localization error. We varied the threshold from 2 to 70 degrees. The localization was executed three times for each of the views in the test sets. The localization error was computed as the deviation from the baseline pose in translation only.

### TABLE I

<table>
<thead>
<tr>
<th></th>
<th>rice</th>
<th>salt</th>
<th>cereals</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial model</td>
<td>45,755</td>
<td>30,490</td>
<td>115,135</td>
</tr>
<tr>
<td>stable keypoints</td>
<td>5,133</td>
<td>12,697</td>
<td>25,960</td>
</tr>
<tr>
<td>clustered stable</td>
<td>2,008</td>
<td>3,677</td>
<td>9,714</td>
</tr>
<tr>
<td>sub-sampled</td>
<td>485</td>
<td>285</td>
<td>1,478</td>
</tr>
</tbody>
</table>
The resulting number of keypoints in the object models can be seen in Fig. 6. Compared to the initial number of keypoints (see Tab. I), a strong reduction in model points can be seen even for small values of $\alpha_{\text{min}}$. This is probably due to the high measurement noise of the Kinect sensor that leads to a high number of unstable keypoints. It can also be seen, that thresholds bigger than 20 degrees do not lead to strong reductions in the model sizes.

Localization errors for each object and each view angle threshold are given in Fig. 7. It can be seen that the localization error is smaller than the Kinect sensor noise of 3 mm up to a threshold of 20 degrees. For this reason, we decided to use a value of $\alpha_{\text{min}} = 20$ degrees in the following experiments.

In a second experiment, we performed local clustering as described in Sec. V-C to remove duplicate model points. The clustering was applied to the models after unstable keypoints were removed. It can be seen from Fig. 6 and Tab. I that more than half the points are filtered out in this step. Note that in Fig. 6 the numbers of keypoints after clustering are depicted as bars in lighter colors. Localization errors using the clustered models were comparable to those of the unclustered models.

C. Spacial Sub-sampling

In this experiment, we applied spacial sub-sampling using a voxel grid of various resolutions. The input to the sub-sampling were object models that have been thresholded on the view angle $\alpha_{\text{min}} = 20$ degrees and have been clustered locally. Again, the localization was executed three times for each of the views in the test sets and the results were compared to the baseline poses.

The resulting model sizes are given in Fig. 8 and the localization errors are plotted in Fig. 9. It comes as no surprise that sub-sampling leads to a substantial reduction in model sizes. Depending on the size of the object, however, a voxel size of, e.g., 30 mm may reduce the model to very few points so that such a model cannot be used for object recognition. But from Fig. 9 one can see that a sub-sampling with voxel sizes of up to 10 mm led to results that are comparable to those in Sec. VI-B. With a voxel size of 10 mm, the localization error is below the sensor noise of 3 mm, yet, the models are reduced to 10% to 20% compared to the models generated in Sec. VI-B.
we achieved a reduction of the model to approximately 1% of the original model size without a substantial loss of localization performance.

ACKNOWLEDGMENT

This work was funded in parts under the ARTEMIS Joint Undertaking as part of the project R3-COP, from the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS10004E.

REFERENCES