Computers & Graphics 79 (2019) 36–45

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Fast template matching and pose estimation in 3D point clouds☆

Richard Vock*, Alexander Dieckmann, Sebastian Ochmann, Reinhard Klein

University of Bonn, Germany

ARTICLE INFO

Article history:
Received 20 June 2018
Revised 19 December 2018
Accepted 20 December 2018
Available online 17 January 2019

Keywords:
Point clouds
Template matching
Pattern matching

ABSTRACT

Template matching for 3D shapes in point cloud data is an essential prerequisite for a multitude of applications such as bin picking tasks for known objects, detection and completion of redundant object instances during scanning endeavors, and verification of industrial assemblies. Building on existing approaches for template matching, especially on methods utilizing point tuple features for the quick generation of transformation guesses in a RANdom SAmple Consensus (RANSAC) setting, we introduce an improved, targeted sampling strategy as well as an efficient hypothesis validation approach to drastically improve the overall runtime. In our experiments the proposed optimizations lead to a performance increase by two orders of magnitude in comparison to an unoptimized implementation. Several experiments on diverse real-world and simulated datasets demonstrate the robustness of our proposed approach.

1. Introduction

Point clouds have become an ubiquitous representation for measured geometry in a variety of domains ranging from architecture and design to robotics and industrial applications. In practice they are either acquired using terrestrial laser scanners or reconstructed from RGB-D image data. In order to get true 3D data either multi-view laser scan data is combined in a separate registration or in the case of RGB-D data a frame-wise integration into a consistent 3D model is performed [1]. Detecting objects in such 3D point clouds and estimating their pose enables us to decide whether or not the scene has been fully scanned, how to grab an object in robot object picking applications, to verify the correct placement of objects in assembly scenarios and last but not least improve scene understanding in general. One particular category of methods is based on template matching where a known object or a part of an object is given as a template and then all occurrences are searched for in a larger, usually cluttered scene. These occurrences however may be only partial due to occlusions. The template might be provided as another point cloud, a subset of the scene, or a CAD model.

The main challenge of the template matching approach is to keep the search computationally feasible and fast while still being robust with respect to measurement errors or partiality.

Methods utilizing point tuple features [2] for the quick generation of transformation heuristics in either a RANSAC [3–5] or voting-based [2,6] scheme have proven to be well suited for this task. Among the competing approaches they proved to be the most versatile and robust especially w.r.t. partiality. Unfortunately, for large point clouds, their runtime becomes a major limitation by exceeding the interactive response times needed in many of the example applications mentioned above.

Therefore, in this paper we introduce two key improvements in order to accelerate these methods. First, we observed that in most scenarios 90–95% of the computation time was spent on correspondence estimation during the scoring phase and introduce a novel, voxel based scoring method with an early exit strategy. Second, we introduce a novel sampling strategy for the generation of transformation hypotheses which is built on the sampling of stable, salient points and which exploits the locality of possible template occurrences in order to avoid the generation of unnecessary transformation hypotheses. By combining these improvements, we show that RANSAC approaches are capable of rendering complex problems manageable.

In summary the main contributions of our work are:

- A targeted sampling strategy based on a novel edge detection approach efficiently selects salient points and point pairs while keeping stable candidates for robust transformation hypothesis generation.
- A two-step sample count estimation exploits locality to minimize generated hypothesis count.
- Hypothesis validation is vastly accelerated by using an approximate, voxel-based approach which allows us to process large datasets while preserving the accuracy given by the input data.
- Additionally, extensive hypothesis testing allows us to completely avoid the majority (usually more than 80%) of

☆ This article was recommended for publication by H Fu.

* Corresponding author.
E-mail address: vock@cs.uni-bonn.de (R. Vock).

https://doi.org/10.1016/j.cag.2018.12.007
0097-8493/

hypothesis validation computations in the first place, quadrupling the matching performance in our experiments.
- In our experiments our proposed changes lead to a performance increase by two orders of magnitude in comparison to an unoptimized implementation.

Some of these improvements are inspired by recent work on RGB-D data not directly tailored to 3D point cloud data.

We demonstrate our approach by applying it to several challenging simulated as well as real-world point cloud datasets.

2. Related work

As stated in the introduction we base our method on the well-established general framework introduced by the seminal work of Drost et al. [2] and Papazov and Burschka [3] due to their ability to efficiently cope with partial or erroneous data. The approach of Drost et al. [2] uses oriented Point Pair Feature (PPFs) [7] for generating rigid transformation hypotheses of free-form objects in point clouds. Furthermore a voting scheme to determine meaningful candidate transformations is used. The results are evaluated with respect to robustness against noise, clutter and partiality. Should additional intensity images be available Drost and Ilic [6] propose an extension to their method using multi-modal features. As a means to improve overall performance of PPF matching with subsequent Iterative Closest Point (ICP) refinement, Drost and Ilic [8] introduce a hierarchical voxel hash for nearest neighbor lookups. In our work, a similar idea is used for performing nearest neighbor queries in several steps. Similar to Drost et al. [2], PPFs are used by Papazov and Burschka [3] for 3D object recognition but using a RANSAC scheme for generating transformation hypotheses. They likewise use a hash table to quickly retrieve point pairs that are similar to pairs sampled in the scene. This method has been demonstrated for usage in robot grasping tasks in a follow-up work [4].

Another RANSAC based approach is presented by Thomas [5] which additionally evaluate the usage of point triples instead of pairs. While point triples are reported to slightly improve recognition rates, Hillenbrand and Fuchs [9] found point pairs to be more effective than triples if data quality is sufficient while point triples are increasingly beneficial in case of high noise levels. We found that the additional computational cost is prohibitive in large, high-density datasets as used in our work – especially in the context of an expected increase in data quality from next-generation acquisition devices. A method for template matching based on depth and color images from Kinect sensors is presented by Hinterstoisser et al. [10]. They propose using color information in order to prune invalid candidate transformations while depth data is used to improve pose estimation using ICP. In a follow-up work Hinterstoisser et al. [11] revisit PPFs and propose several improvements over the original implementation. In particular, a smarter point sampling strategy is proposed focusing on point pairs which are likely to lie on the template object. We apply a similar albeit simpler strategy to avoid sampling point pairs which are improbable or impossible in order to reduce computation costs. With their proposed approach Hinterstoisser et al. outperform all competing methods with respect to performance as well as robustness. It is therefore the most relevant work to compare our work to. Also using RGB-D data streams from mobile acquisition devices, Li et al. [12] devise a real-time system for detection and placement of CAD models in scanned scenes focusing primarily on fast detection rates. This comes at the cost of errors in model selection. It should be noted that our approach instead focuses on higher-quality point cloud data aiming for *exact* matches and poses of templates. Extensions to the features used by the original PPFs are proposed by Choi et al. [13] which distinguish surface and boundary points for building different PPFs. The authors argue that object boundaries are descriptive features in certain domains as demonstrated by their robot bin-picking scenario. We inherit the idea of using object boundaries not only for building our features but also for reducing the point pair sampling space. A more general overview of different local feature based methods is given in [14].

3. Method

As pointed out in the previous Section, our method is based on PPFs [2] used in a RANSAC scheme [15] to find and validate transformation hypotheses. The main differences w.r.t. previous methods however are a drastically different sampling strategy combining a targeted selection with a probabilistic sample count estimation as well as an improved voxel based scoring method with an early termination based on hypothesis testing.

As illustrated in Fig. 1 our pipeline consists of the following steps:

1. In a preprocessing step, edge points are determined (Section 3.2) and a PPF hash map for the template is generated.
2. The main RANSAC loop (Section 3.3) matches sampled point-pairs with template pairs using hash map queries (Section 3.4) to determine pair *correspondences* and generate *transformation hypotheses* (Section 3.5).
3. These transformations are *scored* using a voxel distance field (Section 3.6) while stopping early in case of apparent misalignment and keeping track of the best candidate.
4. All sufficiently good candidates are used to fine-align the template with the scene point cloud via voxel-based ICP and corresponding scene points are removed for a potential restart of the RANSAC loop.

In the following subsections we provide detailed descriptions of the individual steps of our processing pipeline. Section 4 then shows results of our implementation on real-world datasets.

3.1. Preliminaries preprocessing

Most of the parameters used in our approach depend on two fundamental quantities that we determine early. The first is the diameter δ of the template point cloud $\mathcal{P}'$ and is computed as the diameter of its bounding box. The second is the approximate point cloud resolution ρ and is approximated as the average nearest neighbor distance in the template point cloud $\mathcal{P}'$. This also implies the assumption that the scene point cloud $\mathcal{P}$ and template point cloud $\mathcal{P}'$ share a common sampling resolution. We will also consider the case where this assumption is violated in Section 3.6.

An additional preprocessing step is the edge detection. Note that the detection in the *scene* point cloud $\mathcal{P}$ is an *online* computation step.

3.2. Edge detection

Since our method is based on sampling a large quantity of point pairs the most feasible approach to optimization is a reduction of potential sample candidates. At the same time providing some redundancy helps maintaining robustness w.r.t inaccuracies or partiality. Possible candidates for pruning include points that

- have weak descriptive potential w.r.t. the overall shape,
- are hard to localize in a stable way, or
- do not result in stable transformation candidates.

In order to balance both aspects – reduction and redundancy– we propose to consider only *edge points* while sampling, since they fulfill all of our requirements: edge points serve as a viable description of shape regardless of material and environment and

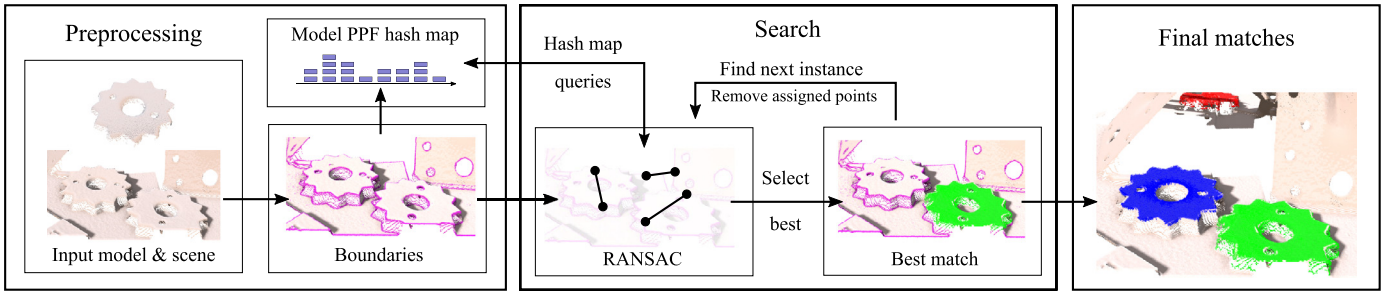

Fig. 1. Overview of our computation pipeline. Boundary/edge points are detected (left) and point pairs are sampled in order to generate transformation hypotheses in a RANSAC loop (middle). The best candidates with respect to a match fitness score then yield the final transformation set (right).

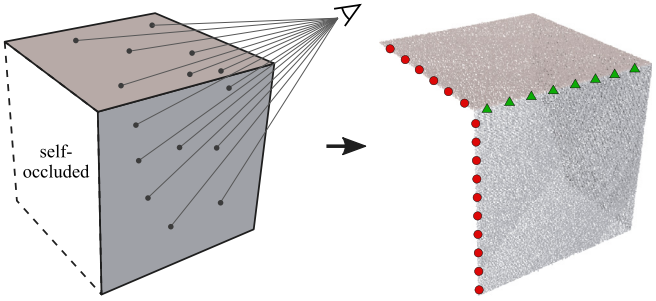

Fig. 2. Corners scanned from one side result in planar boundaries (red circles) rather than edges (green triangles). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

point pairs sampled from different edges provide stable transformation candidates. Additionally edges permit a stable detection even in noisy or partial data.

It is worth noting however that in real-world data a simple edge detection is not sufficient. Especially hard edges of objects are often only measured from one specific angle, occluding one side of the edge. It is therefore vital to also detect *boundaries* of near-planar surface patches as shown in Fig. 2.

In order to determine the subset $\mathcal{E} \subset \mathcal{P}$ of candidate points in our scene point cloud $\mathcal{P}$ we propose a hybrid approach combining a multitude of weak features in a simple binary clustering scheme to yield a stable detection of boundary as well as edge points with according directions. Note that in the remainder of this work we will refer to both as *edges* or *edge directions*.

For each point in both the scene and the template point cloud we compute a 5-dimensional feature vector

$$f(p) = \begin{pmatrix} f_a(p) & f_h(p) & f_p(p) & f_s(p) & f_v(p) \end{pmatrix}^T.$$

Following Drost and Ilic [16] and Mian et al. [17] we compute local descriptors based on the weighted covariance tensor of a point p's neighborhood $N_r(p)$ with radius r (chosen as a multiple of ρ),

$$\Sigma(p) = \frac{1}{|N_r(p)|} \sum_{p_i \in N_r(p)} h(\|p_i - \overline{p}\|_2)(p_i - \overline{p})(p_i - \overline{p})^T,$$

where $\overline{p}$ is the medoid of $N_r(p)$ and h is chosen as a Gaussian window function with fixed variance σ^2. The eigenvalue decomposition of $\Sigma(p)$ yields eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ and corresponding eigenvectors e_1, e_2 and e_3.

The first two features $f_a(p)$ and $f_h(p)$ are given by the angle and half-disc criteria as introduced by Bendels et al. [18]. The first one measures the largest angle gap between neighboring points with respect to its normal while the other one measures the distance between p and its neighborhood's centroid.

The remaining features $f_p(p)$, $f_s(p)$ and $f_v(p)$ are computed following Weinmann et al. [16]:

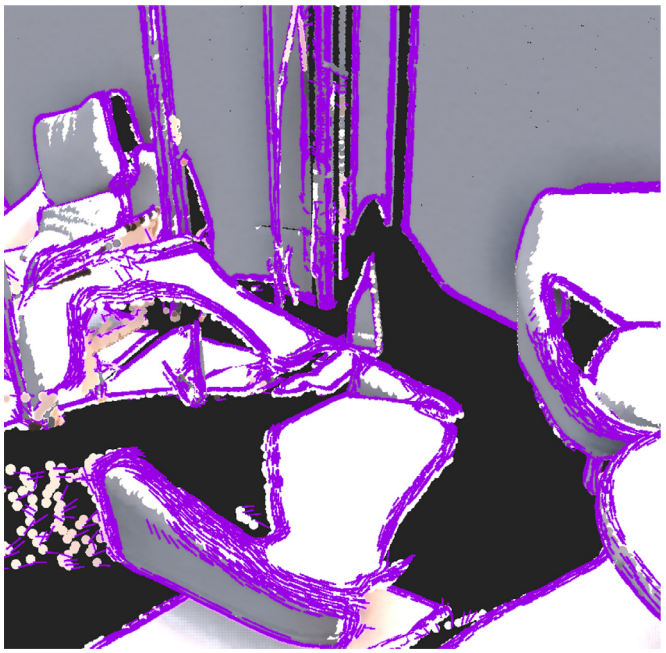

Fig. 3. Detected edge positions and directions.

- $f_p(p) = 1 - (\lambda_2 - \lambda_3)/\lambda_1$ is the "non-planarity" of $N_r(p)$,
- $f_s(p) = \lambda_3/\lambda_1$ is the sphericity, and
- $f_v(p) = \lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$ is the surface variation.

These feature vectors are then normalized such that $f(p) \in [0, 1]^5$ by feature-wise subtraction of minimum occurring feature values and division by the feature's range of values. Subsequently these normalized features are clustered into two clusters using k-means clustering in order to distinguish between edge and non-edge points. The cluster with higher centroid norm then forms the set of edge points $\mathcal{E} \subseteq \mathcal{P}$ (or $\mathcal{E}' \subseteq \mathcal{P}'$ for the template point cloud) and $\forall p \in \mathcal{E}$ we call the second eigenvector e_2 the edge direction $t(p)$ of p (see Fig. 3 for an example scene with detected boundary points).

3.3. Sampling

As for all RANSAC methods the algorithm follows a sample-hypothesize-verify loop. An important question is when to stop sampling more candidates and consider the current sample count "sufficient".

Following Papazov and Burschka [3] we rely on the well-proven probabilistic framework proposed by Schnabel et al. [19] in the context of primitive shape detection. A direct application of this idea would be to consider the random sampling of point pairs

$(p_1, p_2) \in \mathcal{E} \times \mathcal{E}$ and determine sampling bounds based on the probability of this pair belonging to an instance of the template. The initial method by Schnabel et al. [19] assumes global hypotheses (e.g. the detection of infinitely supported planes in point clouds), but in our case point pair sampling can be bounded by an upper distance equal to the template diameter δ. In other words sampling can be split into two distinct steps: random sampling of the first point p_1 and subsequent sampling of the second point p_2 within the neighborhood $N_\delta(p_1)$ with maximum distance δ.

Our argumentation now follows that of Schnabel et al. [19] with a different probability of either the first or second point belonging to a template instance. Following, we briefly reproduce the original derivation altered to fit our usage scenario:

Considering a random first point $p_i \in \mathcal{E}$ we approximate the probability of p_i belonging to a matching subset Ψ as

$$P(p_i \in \Psi) = \frac{|\Psi|}{|\mathcal{E}|}.$$

The probability $P(p_i \in \Psi, s)$ of sampling a correct point after s candidate points is then the complementary of the probability of s consecutive wrong samples:

$$P(p_i \in \Psi, s) = 1 - (1 - P(p_i \in \Psi))^s.$$

Solving for s allows us to determine a minimum sample count n_1 such that $P(p_i \in \Psi, s) \geq p_t$ where p_t is a user-specified success probability (we used $p_t = 1 - 10^{-5}$ throughout our experiments). This yields the bound

$$n_1 \geq \frac{\ln(1 - p_t)}{\ln(1 - P(p_i \in \Psi))}.$$

With $p_j \in N_\delta(p_i)$ the number n_2 of second points to sample is then computed like above with the slight adjustment of setting

$$P(p_j \in \Psi) = \frac{|\Psi|}{|N_\delta(p_i)|}.$$

Note that when assuming $|\Psi| \approx |\mathcal{E}'|$ this probability is likely to be close to 1 leading to a small second sample count n_2. This, in practice, leads to a rather linear than quadratic complexity since the amount of points in a given, constant sized ball around first sample points is itself bounded by a constant and close to $|\mathcal{E}'|$.

We also limit the set of valid point pairs by imposing additional geometric constraints. More specifically we require that

$$\|d_{ij}\|_2 \in [d_{\min}, d_{\max}], \text{ and } |\langle d_{ij}, t(p_i)\rangle| < \cos(\theta_\alpha),$$

where $d_{ij} = p_j - p_i$ and $t(p_i)$, $\|t(p_i)\|_2 = 1$, is the edge direction of p_i. $d_{\min}$ and $d_{\max}$ are lower and upper bounds on the distance between point pairs and are chosen as constant multiples of δ (we use 0.4δ and 0.7δ). The angle threshold is usually a constant and in our experiments chosen (conservatively) as $\cos(\theta_\alpha) = 0.7$. Note that the angular condition is vital for stable transformation hypotheses (see Section 3.5).

Finally, in case the score computation (see Section 3.6) yields a sufficiently high score during sampling (above a given threshold), we terminate early under the assumption that finding a better transformation is unlikely.

3.4. Fast pair correspondences

Given a sampled scene point pair $(p_i, p_j) \in \mathcal{E} \times \mathcal{E}$ we want to quickly retrieve a (minimal) set of matching template point pairs in order to generate transformation hypotheses. Similar to Drost et al. [2] we base this correspondence estimation on a fast hash query structure mapping point pairs to similar point pairs via hashes of discretized 4-dimensional feature vectors.

More specifically given $(p_i, p_j) \in \mathcal{E} \times \mathcal{E}$ with corresponding edge directions $t(p_i), t(p_j) \in \mathbb{R}^3$ we compute the feature vector

$$f(p_i, p_j) = \begin{pmatrix} f_1(p_i, p_j) \\ f_2(p_i, p_j) \\ f_3(p_i, p_j) \\ f_4(p_i, p_j) \end{pmatrix} = \begin{pmatrix} \|p_j - p_i\|_2 \\ \angle(p_j - p_i, t(p_i)) \\ \angle(p_j - p_i, t(p_j)) \\ \kappa_{\min}/\kappa_{\max} \end{pmatrix},$$

where $\kappa_{\min}$ and $\kappa_{\max}$ are the principal curvatures at p_i and

$$\angle(x, y) := \tan^{-1}\left(\frac{\|x \times y\|_2}{|\langle x, y\rangle|}\right) \quad \forall x, y \in \mathbb{R}^3,\ x, y \neq 0,$$

denotes an orientation invariant angle between two vectors. Note that in case of unstable curvature estimation the ratio $\kappa_{\min}/\kappa_{\max}$ can be set to 0 without loss of detection quality.

These real-valued vectors are then discretized into a user-specified number of bins (we use 15 bins for distances, curvatures and angles) and combined into a single hash value for fast indexing. For this purpose we use a custom (functionally identical) reimplementation of the well tested *MurmurHash3* hash function [20]. Note that this binning introduces a distance tolerance threshold equal to the maximum distance divided by the number of bins. Drost et al. [2] use these as a subsampling resolution for the scene point cloud.

3.5. Transformation hypotheses

For any sampled scene point pair $(p_i, p_j) \in \mathcal{E} \times \mathcal{E}$ with corresponding edge directions $t(p_i), t(p_j) \in \mathbb{R}^3$ we now want to quickly derive a minimal set of viable transformation hypotheses using hash map queries. Given (p_i, p_j) and a matching template point pair $(p_k, p_l) \in \mathcal{E}' \times \mathcal{E}'$, we compute a corresponding rigid transformation hypothesis $T_{ij,kl}$ as follows: Let u_{ij}, v_{ij} and $u_{ij} \times v_{ij}$ be a local reference frame computed as

$$u_{ij} = \frac{p_j - p_i}{\|p_j - p_i\|_2}, \quad v_{ij} = \frac{v'_{ij}}{\|v'_{ij}\|_2},$$
$$v'_{ij} = (\mathbb{1} - u_{ij}u_{ij}^T)\ t(p_i),$$

where $\mathbb{1} \in \mathbb{R}^{3\times3}$ is the identity matrix and

$$R_{ij} = \begin{pmatrix} u_{ij} & v_{ij} & (u_{ij} \times v_{ij}) \end{pmatrix} \in \mathbb{R}^{3\times3}$$

the rotation matrix aligning these local reference frames.

By setting $R_{ij,kl} = R_{kl}R_{ij}^T$ in the transformation chain

$$T_{ij,kl} = \begin{pmatrix} \mathbb{1} & p_k \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R_{ij,kl} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbb{1} & -p_i \\ 0 & 1 \end{pmatrix}$$

we get the final transformation matrix

$$T_{ij,kl} = \begin{pmatrix} R_{kl}R_{ij}^T & p_k - R_{kl}R_{ij}^T p_i \\ 0 & 1 \end{pmatrix} \in \mathbb{R}^{4\times4}.$$

In the special case that the template as well as the scene point cloud have a reasonable known "up-direction" d_{up}, we additionally propose only allowing transformations which keep this direction invariant by only considering transformations where

$$1 - |d_{\text{up}}^T R_{kl} R_{ij}^T d_{\text{up}}| < \varepsilon_{\text{up}},$$

for some small ε_{up}.

While significantly impacting performance this also severely limits the set of detectable similarity transforms. We therefore explicitly denote this in our evaluation where appropriate.

3.6. Validation of hypotheses fast point correspondences

Similar to previous approaches we utilize a fast overlap estimation between the template and scene point clouds as a score measure. However this requires a costly point correspondence estimation in the form of a nearest-neighbor or range query. In order to avoid these high costs we propose a method based on approximate neighborhood queries in a voxel grid precomputed for the template point cloud. Additionally we rely on early hypothesis testing in order to terminate score computation in case of a sufficiently low *expected* score. Note, that since we only need fast neighborhood queries in one representation, we do not require any voxel grid for the scene point cloud – only the template point cloud is voxelized.

This voxel grid is aligned to the bounding box of the template point cloud, while each cell of the voxel grid stores the index of the point nearest to its center. Resolution is chosen as the bounding box length divided by the point cloud resolution separately for each direction (e.g. in our test templates this lead to a grid size in the range [51,788] cells per axis). Given any point $p \in \mathcal{E}$ and a transformation T we define the *voxel query function* $V_T : \mathcal{E} \mapsto \mathcal{P}'$ such that $p' = V_T(p)$ is the template point closest to Tp while ignoring points p for which Tp is located outside of the voxel grid.

Let $\nu(p)$ of a point p be either the edge direction, if p has such, or otherwise its normal direction. Given a set of query points $\mathcal{Q} \subseteq \mathcal{E}$, we define the set of *correspondences* $\mathcal{Q}'_T$ as

$$\mathcal{Q}'_T = \{(p, V_T(p)) \mid \mathrm{pred}_T(p, V_T(p))\} \subset \mathcal{Q} \times V_T(\mathcal{Q}),$$

with the correspondence agreement predicate

$$\mathrm{pred}_T(p, p') = \|Tp - p'\|_2 < k\rho \wedge \frac{|\langle T\nu(p), \nu(p')\rangle|}{\|\nu(p)\|_2 \|\nu(p')\|_2} > \cos(\theta_\alpha),$$

where k is a constant tolerance factor provided by the user – we used $k = 3$ in our experiments – and α is an angle threshold we conservatively set to $\cos(\theta_\alpha) = 0.7$. This allows us to compute the *alignment score*

$$\sigma_T(\mathcal{Q}) = \sum_{(p,p') \in \mathcal{Q}'_T} \exp\left(-\frac{\|Tp - p'\|_2^2}{2k\rho}\right) \tag{1}$$

for each query set and candidate transformation.

Since we compute a local neighborhood $N_\delta(p_i)$ of radius δ for each first sampled point p_i, we only score *this* neighborhood for all generated transformation candidates. Should $\sigma_T(N_\delta(p_i)) > \theta_{\text{early}}|\mathcal{P}'|$ for some predefined θ_{early} we consider the transformation good enough to terminate the RANSAC loop early, otherwise we keep track of the best transformation and finally check $\sigma_T(N_\delta(p_i)) > \theta_{\text{final}}|\mathcal{P}'|$ for this best candidate. θ_{early} and θ_{final} are the most influential parameters with respect to match quality. While θ_{early} might be chosen as a high constant (we use 1.0 meaning full surface area coverage), suitable choices for θ_{final} depend heavily on the template as well as scene point clouds. One of the two factors that influences the choice of θ_{final} is partiality, i.e. the extent to which the surface of the object we are looking for is represented by the point cloud. If an occurrence of an object in the point cloud is only covered half by the scan, the highest reachable threshold is expected to be 0.5. The other factor is the difference in sampling resolution between scene and template point cloud – halving the scene resolution also implies half the maximum expected score. While adjusting for the first factor requires domain/measurement knowledge the second one can be accounted for by correcting θ_{final} by the approximated scene/template resolution ratio. In our experiments θ_{final} turned out to be the only parameter we manually tuned in order to achieve the desired result quality, although $\theta_{\text{final}} = 0.6$ is usually a good starting point.

For larger δ our optimized volume based point correspondence estimation still results in a very costly score computation due to the large amount of point transformations per hypothesis. However, most of these hypotheses lead to very poor alignment. Therefore, inspired by Schnabel et al. [19] who use hypothesis tests for approximate score comparisons, we introduce a hypothesis test to estimate the expected score and use the upper bound of the corresponding confidence interval to discard poor transformations early. To this end we split $\mathcal{Q} = N_\delta(p_i)$ into c disjoint subsets

$$N_\delta(p_i) = \bigcup_{j=1}^{c} Q_j, \quad \sigma_T(N_\delta(p_i)) = \sum_{j=1}^{c} \sigma_T(Q_j),$$

where in our experiments we chose c such that each Q_j contains 5% of the query points. Following Schnabel et al. [19] we estimate a projected upper score bound

$$\hat{\sigma}_T^k(N_\delta(p_i)) = -1 - f\left(-2 - \sum_{j=1}^{k} |Q_j|,\ -2 - |N_\delta(p_i)|,\ -1 - S^k\right),$$

where $S^k = \sum_{j=1}^{k} \sigma_T(Q_j)$ and

$$f(N, x, n) = \frac{xn + \sqrt{\frac{xn(N-x)(N-n)}{N-1}}}{N}$$

is the mean plus the standard deviation of the hypergeometric distribution. After testing each Q_j iteratively, we check if $\hat{\sigma}_T^k(N_\delta(p_i)) < \theta_{\text{final}}|\mathcal{P}'|$ in order to decide whether or not to discard the transformation candidate before testing all points.

4. Evaluation

We evaluate our template matching approach on a variety of real-world and simulated test datasets. Furthermore, although not tailored to single-view RGB-D data, we additionally compare our algorithm in such settings on state of the art RGB-D datasets. The real-world terrestrial laser scan datasets are listed together with timings in Table 5 while the simulated test datasets were generated in order to evaluate robustness w.r.t. partiality and noise.

In order to not only test our approach in different scenarios but to additionally have a sound comparison to previous approaches we implemented several of those. The most promising category of methods are based on point pair features (e.g. [2,3,7,13]) with the most prominent being the voting-based approach proposed by Drost et al. [2]. Especially for the larger datasets however our reimplementation of this original voting-based approach had an infeasibly slow runtime performance. The full resolution pump template (see Fig. 6) was even impossible to search for due to consuming too much memory for the hash data structure.

Several published methods improve on runtime and memory impact of voting-based approaches. For example Hinterstoisser et al. [11] proposed using point pair features in a localized neighborhood. Choi et al. [13] propose a modified point pair selection and hashing approach tailored to edge points. Papazov and Burschka [3] additionally restrict valid point pairs by their distance and relative angles.

We furthermore evaluated selecting first points (called "reference points" in [2]) using the probabilistic approach discussed in Section 3.3, while sampling second points in the same neighborhood of points. For very sparsely sampled point clouds however we sampled all points using the pair features proposed by Drost et al. [2] since in these cases using potentially less stable edges had no performance benefits anyway. In our experiments this new method *combining* previous approaches and our reference point sampling strategy proved to outperform each individual approach. We therefore decided to compare our RANSAC-based approach to this combined, voting-based approach.

In the following, we start by evaluating the quality and robustness showing that the results are still correct despite the greatly

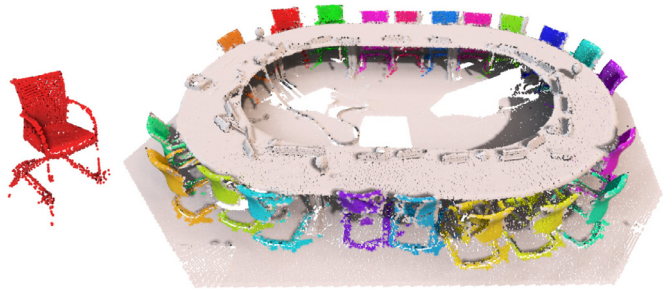

Fig. 4. Detected instances of chairs around a table. Individual instances are illustrated with different colors. In contrast to the scene the template was scanned completely and is shown on the left.

improved efficiency of our algorithm which we evaluate afterwards. We will conclude this Section with a discussion of the limitations of our approach.

Quality and robustness. The assessment of detection/localization quality and robustness was performed on a variety of point cloud datasets used in a multitude of applications and acquisition scenarios.

As one such area of application, we exemplify our approach on different indoor point cloud scans as shown in Figs. 4, 5, and 8. In all tested datasets, our approach was able to robustly detect the occurring template instances including their individual poses – even for partially occluded instances – while only requiring an adaption of the score threshold θ_{final}. All other parameters were derived from estimated quantities like point cloud resolution, diameter etc., or were set to the same default values for all tested datasets ($\theta_{\text{early}} = 1.0$, $\theta_{\text{final}} = 0.6$, $k = 3$). Despite our focus on high-quality point clouds, we also conducted an experiment using a scene reconstructed from a Kinect RGB-D stream. Example results when searching for either a valve or a pump template are shown in Fig. 6.

To assess the overall robustness as well as the effects of undersampling and noise we conducted a larger, systematic test on the T-LESS dataset [21]. To obtain high resolution, multi-view point cloud data, the provided CAD models and groundtruth transformations were used to construct new scenes. These scenes as well as template objects were then sampled at different densities and the resulting positional as well as normal data potentially perturbed. To obtain a meaningful assessment of result quality each template was then searched for in each scene and the detection quality measured by means of the mean recall rate *MR* as proposed by the authors in [22]:

$$MR = \text{avg}_{o \in O} \frac{\sum_{s \in S} |P(o,s)|}{\sum_{s \in S} |G(o,s)|},$$

where O and S are the sets of all template objects and scenes respectively, $P(o, s)$ the set of correctly detected poses and $G(o, s)$ the set of groundtruth poses of object o in scene s. This dataset consisted of 30 templates in 20 scenes. An example scene used for this test is shown in Fig. 7.

Table 1 shows mean recall rates for different sampling densities while Tables 2 and 3 show according values for scenes with normal angular noise as well as translational noise in normal direction. Additionally, results for the voting-based method in these noisy datasets are shown. Note that the voting-based approach in itself performs a subsampling as proposed by Drost et al. [2]. We therefore did not evaluate and compare the robustness of the voting-based approach w.r.t. different sampling resolutions. In general discretization and angular perturbation effects have little impact on the robustness of the detection while surface height deviation quickly leads to deterioration in detection quality. The direct comparison to the voting-based approach shows that the overall robustness of our approach is comparable or even better. In scenes with extreme positional noise however the voting-based approach performs better due to a more robust transformation generation as shown in Table 3. This is due to the transformation being generated by clustering of transformation parameters.

Table 1
Mean recall computed on differently sampled versions of the T-LESS dataset scenes. Note that even at only 25% of points our method successfully detects instances in most cases. Only in cases of extreme undersampling does the method fail in the majority of cases.

#Points template	#Points scene	Relative #Points (%)	Mean recall
7815	47,034	100	0.927
3907	23,515	50	0.875
1953	11,756	25	0.811
780	4699	10	0.510
390	2348	5	0.334

As shown in Fig. 9 a suitable choice of parameters does lead to correct positioning of the perturbed template point cloud. However, it also leads to a large amount of false scene point associations (e.g. in this case floor points).

In order to evaluate our algorithm in case of single view depth datasets we compare it with previously published methods on a state of the art benchmark [31]. Our results show that despite our algorithm not being tailored to this type of datasets it is able to achieve competitive results with respect to both, recall and efficiency.

The main difference between terrestrial laser scans and these datasets lies in the inhomogeneous sampling density. High quality terrestrial laser scans permit a homogenization of the sampling via uniform subsampling while this would lead to an infeasibly low point density for consumer-grade RGB-D cameras. Due to this, 3-dimensional overlap estimation methods usually perform worse than 2-dimensional pixel overlap estimations in depth image space. While image-space scoring is preferably implemented on the GPU we decided to implement such a score using the same multi-core CPU architecture with early stopping of score computation in order to stay reasonably close to our proposed implementation for terrestrial laser scan datasets. The rest of our algorithm was kept unchanged and the evaluation protocol (including parameters) of the SIXD challenge benchmark described in [31] was strictly followed.

Table 4 shows a comparison of average recall rates and timings as proposed by the benchmark paper of Hodaň et al. [31]. Recall rates and timings of the compared results were also taken from this publication.

Edge detection. In our high resolution datasets, we found the edge detection to be exceptionally stable with respect to edge directions and result quality. This however is mostly attributed to the fact, that the search itself works quite well as long as there are *some* edge points with sufficiently correct edge directions. In contrast to normals, edge directions are usually very stable to estimate using PCA, resulting in enough samples for transformation generation. On the other hand choosing conservative parameters to overclassify points as edge candidates has less of an impact on runtime than other sampling related parameters. Problems however arise in case the point density is too low to compute a numerically stable Principal Component Analysis (PCA). Since the benefit of using edge points with their tangent directions instead of surface points with their normals decreases with lower point cloud resolution, we use points with normals in case of low point densities by setting $\mathcal{E} = \mathcal{P}$, $\mathcal{E}' = \mathcal{P}'$ and $t(p_i) = n_i$, where n_i is the normal of point p_i.

Efficiency. Timings for finding *all* occurrences for specific templates are also given in Table 5. In all our experiments the majority of computation time was spent on correspondence estimation

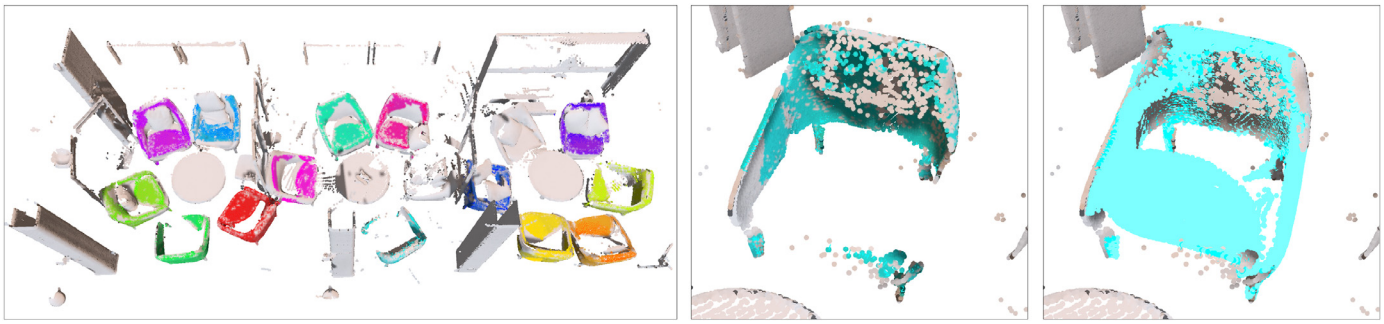

Fig. 5. *Left:* detected instances of measured armchairs. Individual instances are illustrated with different colors. The red instance served as a partially scanned template point cloud. *Middle and right:* example of partial matches due to occlusion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Mean recall (MR) for the voting-based as well as our approach computed on the T-LESS dataset scenes with angular normal noise in both, the template (*left* table) and scene point clouds (*right* table). Perturbation angle was uniformly sampled in the interval $[0, \alpha_{\max}]$.

$\alpha_{\max}$	MR (voting-based)	MR (ours)	$\alpha_{\max}$	MR (voting-based)	MR (ours)
0.5°	0.817	0.815	0.5°	0.845	0.925
1°	0.852	0.915	1°	0.863	0.906
2°	0.863	0.858	2°	0.866	0.913
3°	0.886	0.900	3°	0.857	0.879
5°	0.857	0.852	5°	0.824	0.890

Table 3
Mean recall (MR) for the voting-based as well as our approach computed on the T-LESS dataset scenes with translational noise in normal direction in both, the template (*left* table) and scene point clouds (*right* table). $\xi_{\max}$ denotes the maximum perturbation distance as a factor of the distance tolerance used for hashing (see Section 3.4). Note that a factor of 1 means that on average half of the point-pair queries to the hash map yield wrong results. As expected, this rarely leads to correct transformation hypotheses.

$\xi_{\max}$	MR (voting-based)	MR (ours)	$\xi_{\max}$	MR (voting-based)	MR (ours)
0.0	0.891	0.914	0.0	0.891	0.914
0.05	0.762	0.880	0.05	0.866	0.920
0.1	0.692	0.810	0.1	0.839	0.918
0.2	0.507	0.820	0.2	0.822	0.892
0.5	0.118	0.671	0.5	0.457	0.397
1.0	0.000	0.500	1.0	0.101	0.037

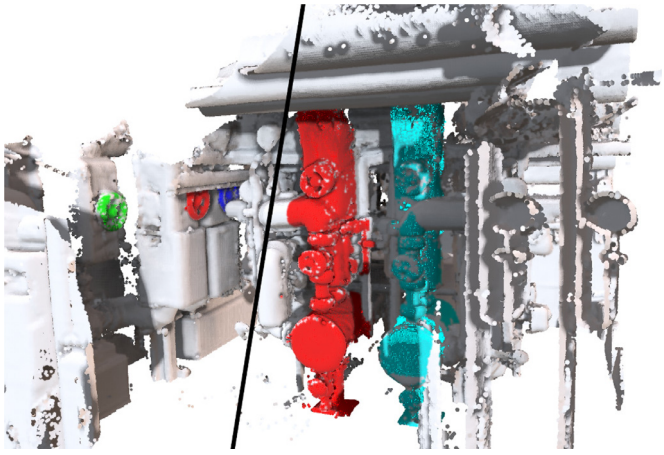

Fig. 6. Instances detected in a scene captured using a Kinect sensor and registered using a voxel hashing approach by Nießner et al. [1]. Individual instances are illustrated with different colors. Note that this image combines results for two templates – a valve and a larger pump – searched for in the same scene.

Table 4
Comparison of average recall scores and timings on the SIXD challenge benchmark [31] datasets. Timings of compared methods were taken from Hodaň et al.[31]. Note that [31] does not list results on the Toyota Light (TYO-L) dataset. For publications proposing multiple variants only the best performing is listed.

Method	IC-MI	IC-BIN	TUD-L	TYO-L	Time (s)
Ours	93.95	77.00	67.10	85.52	4.9
[23]	95.33	96.50	80.17	–	4.7
[2]	94.33	87.00	78.67	–	2.3
[24]	95.33	90.50	45.50	–	13.5
[25]	73.33	56.50	88.67	–	4.4
[26]	95.00	75.00	68.67	–	14.2
[27]	65.00	44.00	7.50	–	1.8
[28]	78.67	24.00	0.0	–	1.4
[29]	36.33	10.00	0.00	–	1.4
[30]	20.00	2.50	0.67	–	47.1

during the scoring phase (usually in the range of 90–95% of total processing time). This means that the fast correspondence estimation via voxel grid queries or complete avoidance of computations by means of intelligent sampling has a more severe impact on performance than fast hash queries for transformation generation. In fact, in the datasets used for the evaluation often the performance impact of using edge points instead of all points was not worth the time spent on edge detection. This on the other hand depends heavily on the type of scene captured in the point cloud.

Fig. 7. Example scene as constructed from the T-LESS dataset. This scene combines 4 different (yet very similar) kinds of fuses.

Table 5
Overview of datasets. Asterisk marks datasets where transformations were forced to keep the up-direction invariant. Time column states total online processing time to find *all* occurrences of the template.

Name	#Points	#Edge points	Time	#Occurrences	Fig.
Chairs	170,184	39,095	2 s*	21	4
Armchairs	573,360	168,049	34 s*	14	5
Toilets	1,532,908	355,927	50 s*	3	8
Valves	1,336,606	639,033	12 s	3	6
Pumps	1,336,606	639,033	77 s	2	6

Table 6
Run-time comparison with a voting-based approach on full resolution and subsampled datasets. The latter are marked with a †. Asterisks mark datasets where transformations were forced to keep the up-direction invariant which in the case of voting-based datasets was performed but hat no performance impact. Note that the pump dataset was left out since the voting-based approach ran out of memory (32 GB RAM).

Dataset	#Points	Voting-based (s)	Ours	Factor
Chairs	170,184	342	3 s/2 s*	171 ×
Chairs†	10,271	20	267 ms/38 ms*	526 ×
Armchairs†	25,584	1034	6 s/6 s*	172 ×
Toilets	1,532,908	2707	53 s/50 s*	54 ×
Toilets†	17,282	73	999 ms/96 ms*	760 ×
Valves	1,336,606	2898	12 s	241 ×
Valves†	173,968	596	1310 ms	454 ×

Hypothesis generation. In order to evaluate the performance of our hypothesis generation we directly compared its performance to the voting-based method described above. The results in Table 6 show that for larger point clouds even the voting-based approach combining multiple optimizations resulted in inferior runtime performance. The comparatively higher performance increase on low resolution point clouds additionally shows the different runtime complexities: our approach behaves approximately linear while the voting based one exhibits the expected quadratic behavior.

It is also worth noting that Drost et al. subsampled the scene point cloud to match the discretization ratio used for hash generation. To this end the minimum distance between points is limited by the distance tolerance used during point pair feature discretization. Since this subsampling drastically improves the runtime performance of our algorithm but is not absolutely required, we compare our method with the voting-based implementation in both the original as well as the subsampled point clouds (this is annotated in Table 6).

Table 7
Median single score computation timings of different datasets with and without early stopping performed with either naïve nearest neighbor search in a kd-tree or our method. Last column shows performance ratio between both optimizations enabled/disabled. Asterisk marks datasets where transformations were forced to keep the up-direction invariant.

Name	Early stop		Score all		Speedup
	Naïve	Voxel	Naïve	Voxel	
Chairs	32 ms*	0.3 ms*	63 ms*	0.7 ms*	172 ×
Armchairs	172 ms*	3.4 ms*	1042 ms*	12 ms*	302 ×
Toilets	301 ms*	4.2 ms*	2205 ms*	19 ms*	525 ×
Valves	31.3 ms*	0.6 ms*	71.8 ms*	1.1 ms*	119 ×
Pumps	2058 ms*	15 ms*	10201 ms*	41 ms*	680 ×

Table 8
Comparison of estimated sample counts for our localized estimation compared to a global approach. First column shows average number of second sample points drawn in the neighborhood of each first point. Second column shows total sample *pair* count in our approach. Third column shows number of sample pairs drawn in a naïve pair sampling approach.

Name	Avg. # 2nd points	Total sample count	Global pairs
Chairs	14	345	14,473
Armchairs	4	161	2565
Toilets	22	1402	2300
Valves	20	3412	517,376
Pumps	57	1500	11,619

Score computation. Due to the lacking availability of reference implementations and missing implementation details especially with respect to the *point correspondence estimation* used for score computation, a direct comparison of this particular step with the most relevant previous work proved difficult. Instead we evaluated the performance increase of our correspondence estimation as well as scoring approach by comparing it with more "naïve" approaches.

In order to evaluate the performance of our method for fast score computations we compared it to naïve nearest neighbor search using a kd-tree for the model point cloud. To this end we replaced the entire voxel grid lookup of our approach by a standard nearest neighbor search as performed by the *FLANN* library. Note that the latter has on average logarithmic as opposed to linear complexity. Additionally we performed runs with and without stopping the score computation early based on hypothesis tests (but using the same samples). Results for three datasets are shown in Table 7. Both the fast neighbor query as well as the large amount of avoided score computations have a significant and consistent impact on runtime performance.

In general, since our sampling and scoring is bounded to a local neighborhood, the size and density of scene point clouds merely affect the number of points to sample, rendering our approach (in theory) applicable to arbitrarily large datasets. Due to the impact on the search radius during sampling, however, the model diameter also turned out to be a large impact factor w.r.t. performance (see e.g. the difference in timings between the "Valves" and "Pumps" searches in Table 5 which were both performed in the same scene but with different templates).

Nonetheless this local sampling approach and especially the localized estimated sample bounds (see Section 3.3) have dramatic effects on the sample count and therefore efficiency as shown in Table 8. The amount of necessary samples is reduced up to two orders of magnitude by not considering point pairs too far apart in the bound estimation.

Limitations. One area where our approach lacks robustness is with heavily heterogeneous point density. While this poses no

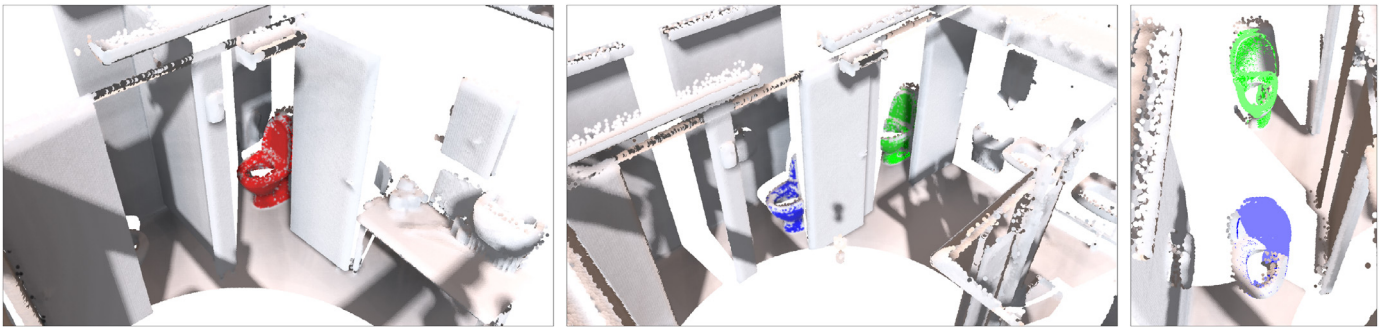

Fig. 8. Detected instances of a partially scanned toilet. Individual instances are illustrated with different colors. Red instance in left image was chosen as the template point cloud. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 9. Upper part shows original and noisy template. Bottom part shows detected instances with different colorization for each instance. Note the additionally associated scene points belonging to the floor.

problem for the transformation generation, it is a fundamental one for score computation. For example if $<90\%$ of points are on one flat surface the object is indistinguishable from the floor w.r.t. an area overlap measure. In this case view-dependent score/error measures are necessary (see e.g. [22]). While possible to implement for these particular scenarios we considered this problem out of scope of this work given the kind of datasets targeted. Another problem arises if edge detection in very low density point clouds yields incorrect or no results at all. Since most of the runtime improvements proposed in this work are targeting the scoring process one might fall back to an approach using just points and their normals instead of edge points.

5. Conclusion

With a novel strategy for the targeted sampling of stable, salient points and point pairs as required for robust transformation hypothesis generation, as well as with an efficient voxel-based validation step, our system provides a fast and accurate template search in challenging high-resolution point clouds.

As demonstrated, utilizing the proposed sampling and scoring improvements, RANSAC approaches are capable of rendering otherwise complex problems manageable. In particular scoring, i.e. what and how to score, has by far the most impact on performance while result correctness and robustness are primarily a result of correct and efficient sampling strategies. By focusing on improving both aspects we believe to have substantially improved previous approaches. We have also shown that partiality and measurement errors are easily mitigated by a sufficiently tolerant matching and scoring scheme leaving the decision of what constitutes similarity to the user.

Our future plans include pushing the detection efficiency such that we can justifiably claim real-time performance.

Acknowledgement

This work was supported by the DFG projects KL 1142/11-1 (DFG Research Unit FOR 2535 Anticipating Human Behavior) and KL 1142/9-2 (DFG Research Unit FOR 1505 Mapping on Demand).

References

[1] Nießner M, Zollhöfer M, Izadi S, Stamminger M. Real-time 3D reconstruction at scale using voxel hashing. ACM Trans Graph (ToG) 2013;32(6):169.
[2] Drost B, Ulrich M, Navab N, Ilic S. Model globally, match locally: efficient and robust 3D object recognition. In: Proceedings of the 2010 IEEE conference on computer vision and pattern recognition (CVPR). IEEE; 2010.
[3] Papazov C, Burschka D. An efficient RANSAC for 3D object recognition in noisy and occluded scenes. In: Proceedings of the Asian conference on computer vision. Springer; 2010. p. 135–48.
[4] Papazov C, Haddadin S, Parusel S, Krieger K, Burschka D. Rigid 3D geometry matching for grasping of known objects in cluttered scenes. Int J Robot Res 2012;31(4):538–53.
[5] Thomas U. Stable pose estimation using RANSAC with triple point feature hash maps and symmetry exploration. In: Proceedings of the international conference on machine vision applications (MVA; 2013.
[6] Drost B, Ilic S. 3D object detection and localization using multimodal point pair features. In: Proceedings of the 2012 second international conference on 3D imaging, modeling, processing, visualization and transmission (3DIMPVT). IEEE; 2012. p. 9–16.
[7] Mian AS, Bennamoun M, Owens R. Three-dimensional model-based object recognition and segmentation in cluttered scenes. IEEE Trans Pattern Anal Mach Intell 2006;28(10):1584–601.
[8] Drost B, Ilic S. A hierarchical voxel hash for fast 3D nearest neighbor lookup. In: Proceedings of the German conference on pattern recognition. Springer; 2013.
[9] Hillenbrand U, Fuchs A. An experimental study of four variants of pose clustering from dense range data. Comput Vis Image Underst 2011;115(10):1427–48.
[10] Hinterstoisser S, Lepetit V, Ilic S, Holzer S, Bradski G, Konolige K, et al. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: Proceedings of the Asian conference on computer vision. Springer; 2012. p. 548–62.
[11] Hinterstoisser S, Lepetit V, Rajkumar N, Konolige K. Going further with point pair features. In: Proceedings of the European conference on computer vision. Springer; 2016. p. 834–48.
[12] Li Y, Dai A, Guibas L, Nießner M. Database-assisted object retrieval for real–time 3D reconstruction. In: Proceedings of the computer graphics forum, 34. Wiley Online Library; 2015. p. 435–46.
[13] Choi C, Taguchi Y, Tuzel O, Liu M-Y, Ramalingam S. Voting-based pose estimation for robotic assembly using a 3D sensor. In: Proceedings of the 2012 IEEE international conference on robotics and automation (ICRA); 2012.
[14] Guo Y, Bennamoun M, Sohel F, Lu M, Wan J. 3D object recognition in cluttered scenes with local surface features: a survey. IEEE Trans Pattern Anal Mach Intell 2014;36(11).
[15] Fischler MA, Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 1981;24(6):381–95.
[16] Weinmann M, Jutzi B, Mallet C. Feature relevance assessment for the semantic interpretation of 3D point cloud data. ISPRS Ann Photogramm Remote Sens Spat Inf Sci 2013(2):313–18.

[17] Hackel T, Wegner JD, Schindler K. Contour detection in unstructured 3D point clouds. In: Proceedings of the 2016 IEEE conference on computer vision and pattern recognition (CVPR); 2016. p. 1610–18.
[18] Bendels GH, Schnabel R, Klein R. Detecting holes in point set surfaces. J WSCG 2006;14.
[19] Schnabel R, Wahl R, Klein R. Efficient RANSAC for point-cloud shape detection. In: Proceedings of the computer graphics forum, 26. Wiley Online Library; 2007. p. 214–26.
[20] Wikipedia Contributors. Murmurhash – Wikipedia, the free encyclopedia. 2018. [Online; accessed 15-October-2018]; URL https://en.wikipedia.org/w/index.php?title=MurmurHash&oldid=843544841.
[21] Hodaň T, Haluza P, Obdržálek Š, Matas J, Lourakis M, Zabulis X. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In: Proceedings of the IEEE winter conference on applications of computer vision (WACV); 2017.
[22] Hodaň T, Matas J, Obdržálek Š. On evaluation of 6D object pose estimation. In: Proceedings of the European conference on computer vision. Springer; 2016.
[23] Vidal J, Lin C-Y, Martí R. 6d pose estimation using an improved method based on point pair features. In: Proceedings of the fourth international conference on control, automation and robotics (ICCAR). IEEE; 2018. p. 405–9.
[24] Hodaň T, Zabulis X, Lourakis M, Obdržálek Š, Matas J. Detection and fine 3D pose estimation of texture-less objects in RGB-D images. In: Proceedings of the 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE; 2015. p. 4421–8.
[25] Brachmann E, Michel F, Krull A, Ying Yang M, Gumhold S, et al. Uncertainty–driven 6D pose estimation of objects and scenes from a single RGB image. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 3364–72.
[26] Buch AG, Kiforenko L, Kraft D. Rotational subgroup voting and pose clustering for robust 3D object recognition. In: Proceedings of the 2017 IEEE international conference on computer vision (ICCV). IEEE; 2017. p. 4137–45.
[27] Kehl W, Milletari F, Tombari F, Ilic S, Navab N. Deep learning of local RGB-D patches for 3D object detection and 6d pose estimation. In: Proceedings of the European conference on computer vision. Springer; 2016. p. 205–20.
[28] Brachmann E, Krull A, Michel F, Gumhold S, Shotton J, Rother C. Learning 6D object pose estimation using 3D object coordinates. In: Proceedings of the European conference on computer vision. Springer; 2014. p. 536–51.
[29] Tejani A, Tang D, Kouskouridas R, Kim T-K. Latent-class hough forests for 3D object detection and pose estimation. In: Proceedings of the European conference on computer vision. Springer; 2014. p. 462–77.
[30] Buch AG, Petersen HG, Krüger N. Local shape feature fusion for improved matching, pose estimation and 3D object recognition. SpringerPlus 2016;5(1):297.
[31] Hodaň T, Michel F, Brachmann E, Kehl W, Buch AG, Kraft D, et al. BOP: benchmark for 6D object pose estimation. In: Ferrari V, Hebert M, Sminchisescu C, Weiss Y, editors. Proceedings of the European conference on computer vision–ECCV 2018. Cham: Springer International Publishing; 2018. p. 19–35. ISBN 978-3-030-01249-6.