Graph-Based Deformable 3D Object Matching

Bertram Drost[1] and Slobodan Ilic[2]

[1]MVTec Software GmbH, [2]Siemens AG

Abstract. We present a method for efficient detection of deformed 3D objects in 3D point clouds that can handle large amounts of clutter, noise, and occlusion. The method generalizes well to different object classes and does not require an explicit deformation model. Instead, deformations are learned based on a few registered deformed object instances. The approach builds upon graph matching to find correspondences between scene and model points. The robustness is increased through a parametrization where each graph vertex represents a full rigid transformation. We speed up the matching through greedy multi-step graph pruning and a constant-time feature matching. Quantitative and qualitative experiments demonstrate that our method is robust, efficient, able to detect rigid and non-rigid objects and exceeds state of the art.

1 Introduction

The accurate and robust detection and localization of 3D objects in cluttered and noisy real-world data is crucial for many robotic and industrial applications. We present a method that is able to efficiently localize deformed 3D object instances in 3D point clouds. For this, we solve the assignment problem through graph matching and return a consistent set of scene-model-correspondences.

Recently, features that describe pairs of oriented 3D points were used successfully in 3D object recognition, rigid 3D object detection and as 3D feature point descriptors [1,2,3]. Such point pairs are invariant against rigid transformations, robust, fast to compute, and – due to their low dimension – fast to match. We show that the set of possible point pair features that describe the deformations of a model can be learned based on only a few training examples.

Drost *et al*. [3] use point pair features in a local voting scheme to find the best matching rigid transformation between a reference model and a 3D scene. We train their method using the point pairs of the deformed models to obtain an initial set of potentially inconsistent scene-model-correspondences. Based on this, we use a graph matching model similar to the one proposed by Leordeanu and Hebert [4] to assign relaxed weights to the assignment candidates based on their overall consistency. We augment the model by using an extended correspondence parametrization that takes 3D motion into account. Finally, a greedy dense subgraph extraction is performed to convert the relaxed assignment weights into a set of consistent correspondences. In essence, the graph matching globally optimizes the correspondences by finding the largest subset of consistent scene-model-correspondences.

The proposed method generalizes well over different object classes and requires no explicit deformation model. Most parameters can remain constant over a large range of

objects, making the method general and easy to use. In terms of performance, we obtain runtimes of around one second for an unoptimized implementation on large scenes. The method requires no feature detector and instead uniformly samples scene and model point clouds.

Note that this work concentrates on the recovery of approximate, but consistent scene-model-correspondences. Additional model and deformation dependent refinement steps, such as deformable ICP [5] or model fitting, are not performed. We evaluate the approach quantitatively and qualitatively on synthetic and real-world datasets, showing its generality, performance and robustness.

2 Related Work

Chui and Rangarajan [6] approach the point correspondence problem in 2D using their TPS-RPM framework that can deal with outliers and uses thin-plate-splines as deformation model. However, their approach was demonstrated on artificial 2D data only. It does not scale well to 3D data with large amounts of clutter due to the worst-case performance of $O(N^3)$. Anguelov *et al*. [7] solve the correspondence problem in 3D using a joint probabilistic model that preserves local geometry. Their method shows very good results when registering meshes of humans using a deformation model that preserves geodesic distance. While the two preceding methods are able to register deformed variants of point clouds, they are unable to deal with larger amounts of outliers, clutter, noise, or occlusion. They are also limited to a single or few deformation models. Those restrictions make the approaches unsuitable as generic 3D deformable object detectors.

Ruiz-Correa *et al*. [8] propose a deformable shape detector that uses a symbolic representation of shape components to represent and detect deformable objects. Their method can deal with occlusion and noise, and generalizes well over different deformation models in a "learn by example" way similar to our proposed approach. However, they report runtimes of over 12 minutes, making their method impractical for real-world robotic applications.

The usage of graph matching algorithms in Computer Vision has a long tradition. An extensive overview is given by Conte *et al*. [9]. Graph matching allows a robust localization of deformed objects and is a promising method for such a challenge. While it has been shown extensively to work in 2D applications, its applications in 3D are mostly limited and restricted to artificial perfect-data scenarios (see, for example, Duchenne *et al*. [10]). Berg *et al*. [11] model the assignment problem as an Integer Quadratic Programming (IQP) problem and use a thin-plane spline for post-processing and outlier removal. Leordeanu and Hebert [4] proposed a relaxation of the binary assignment problem, showing that it's orders of magnitudes faster and more robust than IQP. The graph structure in our proposed method is based on their graph, where vertices represent point-to-point assignments, while edges connect geometrically consistent assignments. They also show the connection between the energy optimization and the eigenvector problem of the adjacency matrix. However, no evaluation on deformable 3D matching was performed.

Recently, hypergraphs were used for efficient image and point cloud registration. Zass and Shashua [12] proposed to use hypergraphs to model more complex relations between two feature sets. Chertok and Keller [13] build upon that work and show efficient hypergraph matching for 2D images. Duchenne *et al*. [10] use higher-order relations for the graph creation, showing good results in both 2D and 3D. However, they evaluate only on perfect 3D meshes and show no quantitative results in 3D. Also, their creation of the adjacency matrix is expensive and makes their method impractical for real-world applications. Leordeanu *et al*. [14] propose a new hypergraph matching algorithm, which they use to efficiently register images that contain deformations. Lee *et al*. [15] extend a random walk strategy to hyper-graphs and can include similarity measures of arbitrary orders. They outperform other methods on 2D when matching feature points on 2D images.

Several of the mentioned methods require feature point detectors and were shown on 2D image data only. While robust feature point detectors in 2D are available, 3D data often exhibits too little distinctive geometry for robust salient point or feature point extraction. The method proposed in this paper thus uses a all-to-all matching that does not require feature point extraction.

Several approaches deal with shape retrival, *i.e*., the identification of 3D point clouds or meshes. Passalis *et al*. [16] use a wavelet representation of objects for efficient shape retrieval in large databases. Mahmoudi and Sapiro [17] identify point clouds based on the distribution of several intrinsic measurements on that cloud, such as geodesic distances. While those approaches generalize well to rigid and non-rigid object classes, they require the objects to be segmented, making the approaches unsuitable to scenes with large amounts of clutter.

Drost *et al*. [3] detects rigid 3D objects in 3D point clouds using point-pair features and a voting scheme with local parametrization. Hinterstoisser *et al*. [18] demonstrate rigid 3D object detection using a high-performance template matching approach in RGB-D data. While both methods show robust results, they do not immediately generalize to non-rigid objects.

3 Method

Both model and scene are subsampled uniformly, to avoid any bias from different point densities throughout the point clouds. In practice, we use sampling distances between 3% and 5% of the model's diameter. We denote $\mathbf{m}_i \in M$ for points on the sampled model and $\mathbf{s}_j \in S$ for points on the sampled scene surface. Both point clouds are oriented, i.e., each point has a normal $\mathbf{n}$ associated with it. The objective is to find a deformed instance of the model in the scene by giving consistent correspondences between scene and model points. Due to occlusion, clutter, and noise, not every scene point has a corresponding model point and vice versa.

Overview In order to find those correspondences, we build a graph $G = (V, E)$, where each vertex $v \in V$ represents a possible correspondence between a scene point and a model point. An edge $e = (v_1, v_2) \in E$ indicates that some non-rigid transformation exists such that both correspondences v_1 and v_2 are aligned simultaneously. In other

words, vertices that represent consistent correspondences are connected. This graph model is based on [4]. If an instance of the model is present in the scene, the graph's vertices that connect the visible model points to their ground-truth scene points will be connected and form a dense subgraph of G. We will extract this subgraph using standard techniques, and thus recover the model-scene-correspondences. We will also show how the graph can be constructed sparsly (aiding performance) and how to extend the vertices by adding another parameter to the correspondence (aiding robustness).

3.1 Model Generation

Feature and Database. We use oriented pairs of 3D points as features for the matching, similar to [1,2,3]. Each pair $(\mathbf{m}_1, \mathbf{m}_2)$ with normals $\mathbf{n}_1$ and $\mathbf{n}_2$ is described by

$$\mathbf{F}(\mathbf{m}_1, \mathbf{m}_2) = (|\mathbf{d}|, \angle(\mathbf{n}_1, \mathbf{d}), \angle(\mathbf{n}_2, \mathbf{d}), \angle(\mathbf{n}_1, \mathbf{n}_2)) \,. \tag{1}$$

where $\mathbf{d} = \mathbf{m}_2 - \mathbf{m}_1$. $\mathbf{F}$ is fast to compute, asymmetric and invariant against rigid motions.

In the online phase, given a scene point pair, we will need to identify all model point pairs that might be similar to the scene point pair under any trained deformation. For this, similar to [3], we discretize $\mathbf{F}$ by uniformly sampling its components and use a hash table H to store a mapping between sampled features and lists of corresponding point pairs. This allows constant-time lookup for similar point pairs.

Deformation Model. Real-world object classes exhibit a large variety of different deformations. In order to be independent from any particular deformation model, we learn the range of possible deformations based only on registered examples $M_1, M_2, \ldots, M_n$ given by the user. We write $\mathbf{m}_i^k \in M_k$ as position of model point $\mathbf{m}_i$ in the deformed example M_k. For each pair $(\mathbf{m}_i, \mathbf{m}_j) \in M^2$, we first collect all its deformations

$$D(\mathbf{m}_i, \mathbf{m}_j) = \{(\mathbf{m}_i^k, \mathbf{m}_j^k) : k = 1, \ldots n\} \tag{2}$$

from the provided examples. We then add all features of the point pairs within the convex hull of D to the database. Note that additionally, the discretization of the feature vectors adds a small range of possible deformations, since variations that do not change the discretized value do not affect the value retrieved from the hash table.

3.2 Vertex Parametrization

Our graph models correspondences between model and scene points. In 2D, a single point-to-point correspondence completely captures a rigid motion, assuming that normal vectors or gradients are available. In 3D, however, a single correspondence misses one degree of freedom: After aligning a scene and a model point as well as their normal vectors, one can still rotate around the normal vector. Using correspondences only is thus an underparametrization of an underlying rigid motion. For graph matching, this has the effect of aggregating vertices and thus probably introducing undesired cliques, making it more difficult to extract the correct correspondences.

To counter this, we explicitly include the rotation around the normal in the vertex parametrization. Each vertex in the graph then represents not only two corresponding

points $\mathbf{s}$, $\mathbf{m}$, but also a rotation angle α around the normal vector. $(\mathbf{m}, \alpha)$ are also called the *local parameters* w.r.t. $\mathbf{s}$. Together with the normals, those parameters completely parametrize a rigid transformation T. Formally, we follow [3] and define T as

$$T(\mathbf{s}, \mathbf{m}, \alpha) = L(\mathbf{s})^{-1} R_x(\alpha) L(\mathbf{m}) \tag{3}$$

where $L(\mathbf{x}) \in SE(3)$ is a transformation with $L(\mathbf{x}) = 0$ and $L(\mathbf{n}(\mathbf{x})) = (1, 0, 0)^T$, and $R_x(\alpha)$ is a rotation around the x-axis with angle α. The rotation angle α is sampled in d intervals, such that each vertex can be parametrized as $S \times (M \times [0; 2\pi]_d)$. The number of vertices in the full graph is then $|S||M|d$.

3.3 Graph Creation and Local Voting Scheme

Handling a graph with $|S||M|d$ vertices can become computationally expensive for larger scenes. In order to improve the matching speed, we prune the graph based on the results of the local voting scheme of [3], thus effectively removing parts which we deem unlikely to be relevant. Fig. 1 outlines the graph creation.

At its core, the local voting scheme is a Hough Transform that recovers the best local parameters $(\mathbf{m}, \alpha)$ given some fixed scene reference point $\mathbf{s}_1 \in S$, *i.e.*, the parameters for which the most scene points are aligned with the model. For this, the parameter space $M \times [0; 2\pi]$ is discretized using $[0; 2\pi]_d$ as described above. The method then iterates over all other scene points $\mathbf{s}_2 \in S$, computes $\mathbf{F}(\mathbf{s}_1, \mathbf{s}_2)$ and matches $\mathbf{F}$ against the hash table H. This returns a list of model point pairs $(\mathbf{m}_1, \mathbf{m}_2)$ for which a deformation exists such that the two point pairs are similar. For each such matching point pair, α_1 is computed by solving (3), and a vote is cast for $(\mathbf{m}_1, \alpha_1)$.

Contrary to [3], we perform the voting for all reference points simultanously. For each model point pair that matches a scene point pair, we obtain the symmetric parameter α_2 and cast a vote for reference point $\mathbf{s}_2$ at $(\mathbf{m}_2, \alpha_2)$. The two corresponding nodes of the graph, $(\mathbf{s}_1, \mathbf{m}_1, \alpha_1)$ and $(\mathbf{s}_2, \mathbf{m}_2, \alpha_2)$, are connected with an edge, since they can both be fulfilled simultanously. We create a sparse graph by adding only those vertices that have a high voting score. This removes vertices and edges that are unlikely to be a part of the object. In practice, for each scene reference point, we use the references with the highest 3% of voting scores.

The left images in Fig. 6 show an example of the pruned graph creation. For a full graph, each model vertex would be connected to each scene point. For our pruned graph, only a small subset of those connections remains. As outlined in Fig. 2, the pruning step improves the runtime of the graph matching by several orders of magnitude.

3.4 Graph Matching

In the following, we follow the notation of [10]. The problem is to find an assignment vector $X \in \{0, 1\}^V$, where X_v is 1 if the scene and model point represented by v correspond and 0 otherwise. This problem is relaxed, such that $X_v \in R^+$, and modeled as an energy optimization problem

$$X^* = \underset{|X|=1}{\operatorname{argmax}} \sum_{e=(v_i, v_j) \in E} X_{v_i} X_{v_j}. \tag{4}$$

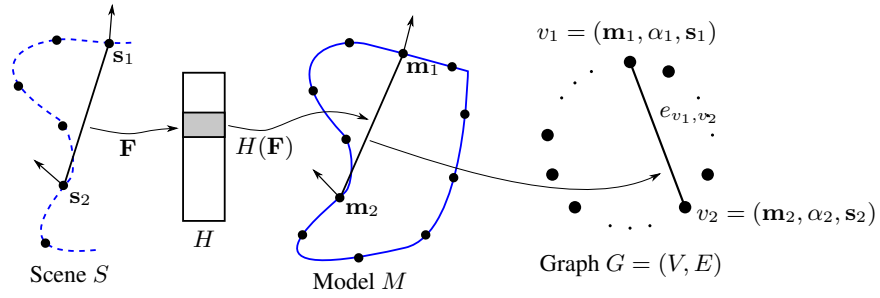

Fig. 1. Graph Construction. *From left to right*: For each scene point pair $(\mathbf{s}_1, \mathbf{s}_2)$, $\mathbf{F}$ is computed. The hash table returns a list $H(\mathbf{F})$ of all model point pairs that can be deformed to match $(\mathbf{s}_1, \mathbf{s}_2)$. *Right*: Each vertex v in the graph represents a possible correspondence between a scene and a model point. Edges are created between vertices that are consistent, *i.e.*, a deformable transformation between scene and model exists that fulfills both correspondences: For each match $(\mathbf{m}_1, \mathbf{m}_2) \in H(\mathbf{F})$, an edge is created.

In terms of the graph's adjacency matrix $A = (w)_{i,j}$, this becomes

$$X^* = \underset{|X|=1}{\operatorname{argmax}} \sum_{i,j \in V} w_{i,j} X_{v_i} X_{v_j}. \tag{5}$$

Note that for the normalization $|X| = 1$, any norm can be used, since we will use the relative values of X only. The problem is then a scaled Rayleigh quotient problem [4,10], and X^* is an eigenvector associated to the largest eigenvalue of A.

We solve the optimization problem through gradient descend. X^0 is initialized to all ones, the update step is

$$X^{k+1} = \frac{AX^k}{|AX^k|} \tag{6}$$

This is equivalent to the power iteration that has proven convergence against an eigenvector of the largest eigenvalue of A.

Voting Scheme Interpretation The iteration 6 can also be seen as a repeated, re-weighted voting scheme: In the first step, each vertex votes for all connected vertices with a weight of 1, such that X_v^1 is the degree of v, *i.e.*, the number of connected edges. In subsequent steps, each vertex v votes again for all connected vertices, but this time with the number of votes it received in the last round, instead of 1. Through this feedback cycle, vertices of a strongly connected subgraph amplify each other, while the values of weakly connected vertices fall due to normalization. With this interpretation, the graph pruning is equivalent to performing the first iteration of (6) on the full graph and then removing vertices with low scores.

3.5 Dominant Consistent Subgraph Extraction

The power iteration gives us a weighted set of vertices or scene-model-correspondences. However, even though the correct correspondences obtain high scores, the set is not necessarily consistent. It might contain outliers as well as non-unique correspondences, i.e.,

two or more connections to a model or scene point. In [4], a greedy approach for extracting the most dominant, consistent dense subgraph was proposed. Their approach, however, is computationally expensive and requires a strong deformation model. [10] modeled the optimization based on the l_1-norm, giving an almost binary correspondence vector, which is easier to threshold. However, we found that this approach has a slower convergence and tends to drop correct nodes. We instead use a simple greedy subgraph extraction. Though this is somewhat of an ad-hoc solution, we found it performs well with little computational costs.

The vertex $v^* = \text{argmax}_{v \in V} X^*(v)$ with the largest score is used as seeding point, and the set of all vertices reachable over no more than two edges ("two hops") is extracted. We found that a single hop is not enough, since the desired subgraph is not a clique, while three hops has too much a chance of introducing incorrect correspondences. To avoid double-correspondences of scene or model points, if a scene of model point is part of two or more extracted correspondences, we only keep the correspondence with the highest value in X^*. Such double-correspondences mostly connect two neighboring points of one set to a single point in the other set, a result of the allowed deformation.

4 Results

We evaluated the proposed approach with several quantitative and qualitative experiments. Synthetic and real data with available ground truth was used for the quantitative evaluation, while the qualitative experiments were performed on a real dataset only.

Note that all parameters were kept constant over all experiments, showing that the method's robustness w.r.t. its parameters. Model and scene were subsampled with distance 3% of the model's diameter. For the hash table, the distance of feature $\mathbf{F}$ was also quantized in steps of 5% of the model's diameter, while angles were quantized in steps of 12°. Fig. 5 (left) motivates the choice for the distance sampling parameter, which is a tradeoff between matching accuracy and matching speed. For each scene, 10 iterations of Eq. 6 were performed.

The method was implemented in C and tested on a Core i5, 3.33 GHz. The off-line learning phase, *i.e.*, creation of the Hash Table H, took less than 1 minute for all objects. Feature matching required 0.05 to 2 seconds, the power iterations 0.1 to 2.5 seconds, depending on the complexity of the scene and the amount of clutter. Timings for the remaining steps, such as scene sampling and greedy dense subgraph extraction, were neglectable. We believe that an improved implementation and a better control over the number of iterations would significantly improve the runtime.

4.1 Quantitative

Synthetic data A first set of experiments was performed on synthetic data, where ground truth is available. We selected three different objects with different surface characteristics, a clamp, a pipe joint and the Stanford Bunny [19] (Fig. 4, left). For each object, 100 scenes were rendered with different amounts of clutter, occlusion, and deformation (Fig. 4, right). The objects were deformed using free-form deformation [20].

Fig. 2. Effect of matching with a sparse graph using the local voting scheme for the scene shown in Fig. 6

	$\|S\|$	$\|M\|$	Vertices $\|V\|$	Edges $\|E\|$	Runtime
Dense	13106	300	135.566	98.886.050	1163.6 s
Sparse	13106	300	34.095	42.832	1.1 s

Fig. 3. Average precision, recall, and relative error of the returned correspondences for the synthetic scenes

Model	Precision	Recall	Rel. Error
Clamp	0.93	0.57	3.6%
Pipe joint	0.99	0.69	2.2%
Bunny	0.96	0.51	4.1%

For training, 10 deformed instances of each object, which were not part of any of the evaluation scenes, were used.

We measure the performance of the method in terms of precision, recall, and error of the recovered correspondences. A recovered correspondence is a true positive if its scene point is on the object and its model point is at most 10% away from its ground truth position. The relative error measures for each true positive correspondence the distance of the corresponding model point to the ground truth model point, divided by the diameter of the object.

Fig. 3 shows the average results for the three objects. The recovered correspondences show a very high precision, indicating that most of the recovered correspondences were correct. The average recall is larger than 0.5, meaning that on average more than half of the correct correspondences were recovered.

Real data We evaluated our approach on the dataset of Mian *et al*. [23,24]. The dataset contains 50 scenes of 5 rigid objects, obtained with a high-precision laser scanner and with available ground truth. Fig. 5 (right) shows the detection rates w.r.t. the occlusion of the objects

Note that even though the objects are rigid, detection still benefits from using our graph approach. This is evident from the fact that we exceed the baseline method of Drost *et al*., which we use to initialize our graph. We also outperform several other state of the art methods.

4.2 Qualitative

We evaluated the proposed method on a set of real-world scenarios. Over 50 scenes containing pretzels, bananas, cappys and stressballs were acquired using both an industrial stereo sensor and a Primesense RGB-D sensor and matched against the corresponding

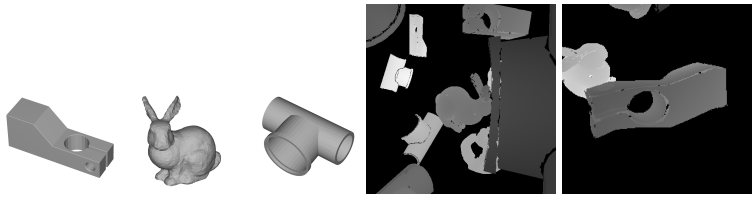

Fig. 4. Left: Objects used for the synthetic tests (clamp, bunny, pipe joint). Right: Example scenes of the synthetic dataset, showing clutter and deformation.

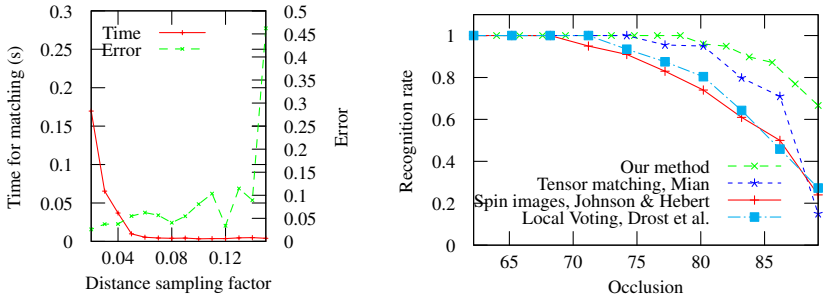

Fig. 5. Left: Effect of changing the distance sampling parameter of the feature database for an exemplary synthetic scene. Matching accuracy and robustness drops significantly when sampling with more than 0.1, while matching time raises significantly when sampling with less than 0.05. In practice, we use 0.05 over all our experiments. **Right**: Detection results on the dataset of Mian *et al*. [21]. Our approach exceeds the rigid baseline method of Drost *et al*. [3] and successfully detects 96.3% (181 of 188) of all objects, and 98.8% (168 of 170) of objects with less than 84% occlusion. Our method also outperforms spin images of Johnson and Hebert [22] and the tensor voting of Mian *et al*. [23].

model. Note that since the stereo sensor does not return an RGB-image, its scenes are visualized in 3D only.

For training, several deformed instances of each object were acquired, manually segmented and registered using deformable ICP [5]. We used only 5 to 15 examples for each class for the training, showing that the method is able to generalize from only few examples.

Fig. 7 show several example scenes. Fig. 6 shows on two examples how the graph creation leads to a sparse graph (1) and how the graph matching extracts a consistent set of correspondences from that graph (2). The effect on the computational costs are shown in Fig. 2. Additional examples are available in the supplementary material.

Overall, we found that the method performs very well even in cases of severe clutter, occlusion, and noise.

4.3 Conclusion

We presented a deformable 3D object detection scheme that generalizes well over different object classes and requires few parameters. We showed how the combination of all possible deformations can be learned based on only a few deformed training samples. The graph matching scheme of [4] was extended by augmenting the correspondences with another parameter, making them more expressive in 3D. We prune the graph by

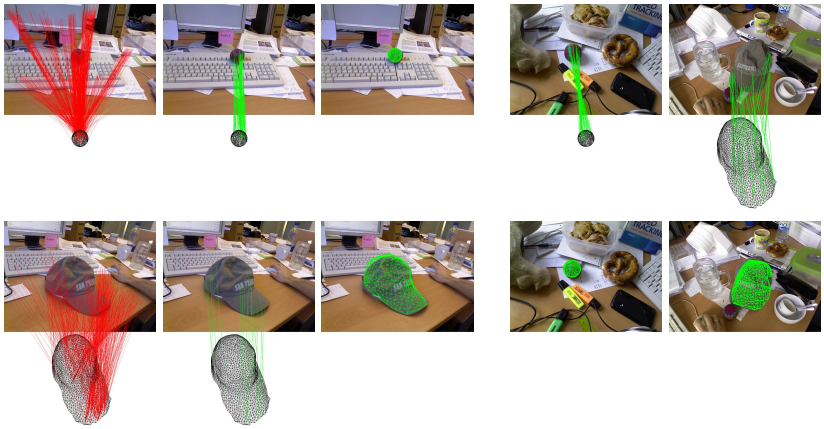

Fig. 6. Graph matching examples. **Left three images**: (1) Initial correspondences, created by thresholding the results of the local voting scheme. Each correspondence is a vertex in our graph. (2) Correspondences extracted after graph matching by the greedy subgraph extraction. Note that only a consistent set of correspondences from the original set of correspondences remains. (3) The correspondences were transformed into a rigid transformation. **Right two images**: Additional examples. The matching was performed on the depth image only, while the RGB image was used for visualization only. Images best viewed in color.

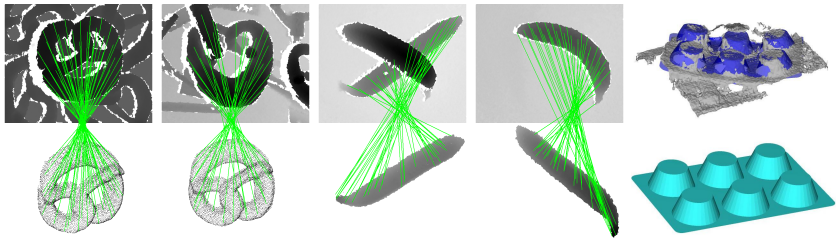

Fig. 7. Qualitatives results on scenes acquired with a stereo sensor. Challenges include clutter, occlusion, multiple instances and strong deformations. The rightmost scene shows the model (bottom) and fitted result (top).

using the method of [3] to create only a sparse set of correspondences that are likely to be correct. Using 3D point pairs makes the method invariant against any rigid 3D transformations. Finally, a greedy dense subgraph extraction is used to find a consistent set of correspondences, which can be used to obtain an approximate rigid transformation or to initialize a deformable ICP.

Our experiments show that the proposed method is able to robustly and quickly detect rigid and non-rigid objects in challenging 3D point clouds despite heavy clutter and partial object occlusion. For rigid objects, we outperform prior art.

References

1. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: ICRA. (2009) 1, 4
2. Wahl, E., Hillenbrand, G., Hirzinger, G.: Surflet-pair-relation histograms: A statistical 3d-shape representation for rapid classification. In: 3DIM. (2003) 1, 4
3. Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3D object recognition. In: CVPR. (2010) 1, 3, 4, 5, 9, 10
4. Leordeanu, M., Hebert, M.: A spectral technique for correspondence problems using pairwise constraints. In: ICCV. (2005) 1, 2, 4, 6, 7, 9
5. Myronenko, A., Song, X.: Point set registration: Coherent point drift. PAMI **32**(12) (2010) 2262–2275 2, 9
6. Chui, H., Rangarajan, A.: A new point matching algorithm for non-rigid registration. CVIU **89**(2) (2003) 114–141 2
7. Anguelov, D., Srinivasan, P., Pang, H.C., Koller, D., Thrun, S., Davis, J.: The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In: NIPS. Volume 17. (2004) 33–40 2
8. Ruiz-Correa, S., Shapiro, L.G., Meila, M.: A new paradigm for recognizing 3-d object shapes from range data. In: ICCV, Citeseer (2003) 1126–1133 2
9. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. IJPRAI **18**(03) (2004) 265–298 2
10. Duchenne, O., Bach, F., Kweon, I.S., Ponce, J.: A tensor-based algorithm for high-order graph matching. PAMI **33**(12) (2011) 2383–2395 2, 3, 5, 6, 7
11. Berg, A.C., Berg, T.L., Malik, J.: Shape matching and object recognition using low distortion correspondences. In: CVPR. (2005) 2
12. Zass, R., Shashua, A.: Probabilistic graph and hypergraph matching. In: CVPR. (2008) 3
13. Chertok, M., Keller, Y.: Efficient high order matching. PAMI **32**(12) (2010) 2205–2215 3
14. Leordeanu, M., Zanfir, A., Sminchisescu, C.: Semi-supervised learning and optimization for hypergraph matching. In: ICCV, IEEE (2011) 2274–2281 3
15. Lee, J., Cho, M., Lee, K.M.: Hyper-graph matching via reweighted random walks. In: CVPR, IEEE (2011) 1633–1640 3
16. Passalis, G., Kakadiaris, I.A., Theoharis, T.: Intraclass retrieval of nonrigid 3D objects: Application to face recognition. PAMI **29**(2) (2007) 218–229 3
17. Mahmoudi, M., Sapiro, G.: Three-dimensional point cloud recognition via distributions of geometric distances. Graphical Models **71**(1) (2009) 22–31 3
18. Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., Lepetit, V.: Gradient response maps for real-time detection of textureless objects. PAMI **34**(5) (2012) 876–888 3
19. Turk, G., Levoy, M.: Zippered polygon meshes from range images. In: Proc. 21st annual conference on Computer graphics and interactive techniques, ACM (1994) 318 7
20. Sederberg, T.W., Parry, S.R.: Free-form deformation of solid geometric models. In: ACM Siggraph Computer Graphics. Volume 20., ACM (1986) 151–160 7
21. Mian, A.S., Bennamoun, M., Owens, R.A.: Automatic correspondence for 3D modeling: An extensive review. International Journal of Shape Modeling **11**(2) (2005) 253 9
22. Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3d scenes. PAMI **21**(5) (1999) 433–449 9
23. Mian, A.S., Bennamoun, M., Owens, R.: Three-dimensional model-based object recognition and segmentation in cluttered scenes. PAMI **28**(10) (2006) 1584–1601 8, 9
24. Mian, A.S., Bennamoun, M., Owens, R.: On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. IJCV (2009) 1–14 8