Learning to Efficiently Detect Repeatable Interest Points in Depth Data

Stefan Holzer[1,2**], Jamie Shotton[2], and Pushmeet Kohli[2]

[1]Department of Computer Science, CAMP, Technische Universität München (TUM)
holzers@in.tum.de
[2]Microsoft Research Cambridge
Jamie.Shotton@microsoft.com,pkholi@microsoft.com

Abstract. Interest point (IP) detection is an important component of many computer vision methods. While there are a number of methods for detecting IPs in RGB images, modalities such as depth images and range scans have seen relatively little work. In this paper, we approach the IP detection problem from a machine learning viewpoint and formulate it as a regression problem. We learn a regression forest (RF) model that, given an image patch, tells us if there is an IP in the center of the patch. Our RF based method for IP detection allows an easy trade-off between speed and repeatability by adapting the depth and number of trees used for approximating the interest point response maps. The data used for training the RF model is obtained by running state-of-the-art IP detection methods on the depth images. We show further how the IP response map used for training the RF can be specifically designed to increase repeatability by employing 3D models of scenes generated by reconstruction systems such as KinectFusion [1]. Our experiments demonstrate that the use of such data leads to considerably improved IP detection.

1 Introduction

Recent developments in depth sensor technology have enabled the widespread use of inexpensive consumer devices such as the Kinect that is able to capture dense depth data at 30fps. This opens a wide range of opportunities for new applications based on depth information such as real time localization and object or scene reconstruction. Methods proposed for this and other such applications involve estimating the pose of the camera by matching the current depth map against a database of previously seen depth maps.

A common approach for solving the above-mentioned problem is to extract interest points in order to reduce the computational burden necessary to perform this matching task. A recent example where such a technique is of interest is the KinectFusion system [1], which demonstrated that depth sensors paired with efficiently parallelized programs running on high-end graphics cards allow

** This research was performed while Stefan Holzer was an intern at Microsoft Research.

to perform dense frame-to-frame depth tracking and enable accurate 3D reconstruction in real-time. However, memory restrictions force the reconstruction to be limited to small office-like environments. For reconstructing larger environments, *e.g.* complete buildings, the scene has to be split into several pieces, which by themselves, can be handled by the reconstruction system. A common way to connect such pieces is to extract interest points which can then be efficiently queried. However, estimating good interest points in noisy depth data is computationally expensive and therefore, only of limited use in online systems.

We reduce the computational load necessary to compute interest points by efficiently approximating response maps of interest point detectors using regression trees. We demonstrate the effectiveness of our approach by learning to predict the response of an interest point detector based on surface curvature. For training and evaluating the regression trees, we use a data set of depth and color image sequences, obtained using a Kinect sensor. Our data set also includes volumetric reconstructions of the recorded scenes as well as synthetic depth and surface normal maps. These have been created from the reconstructed data given by KinectFusion [1], and are therefore more accurate and include less noise than the raw images. Finally, we introduce a way of creating optimized response maps for interest point estimation and show that, by using these maps for training the regression forest model, we can improve the repeatability of online interest point detection.

2 Related Work

Although there has been a lot of research on interest point extraction in 2D color images, there is surprisingly little work on interest point extraction in dense depth data, as supplied for example by the Kinect depth sensor.

3D Interest Point Extraction. Steder *et al.* [2] used an approach based on the Laplacian-Of-Gaussian method to compute interest points in range images. However, this is computationally very expensive and not suitable for real-time or near-real-time operation. In [3], Steder *et al.* presented an interest point detector which first finds and classifies different kinds of 3D object borders and then locates interest points based on this information. Although efficient, this method is specifically designed for range images, which have different characteristics compared to the depth maps obtained from the Kinect sensor. In [4], Unnikrishnan presented a method for extracting interest points with automatic scale selection in unorganized 3D point clouds. However, this work does not take any view-point related information into account.

Learning-based Interest Point Extraction. A number of learning-based approaches have been proposed for efficient estimation of interest points in color or gray value images. Rosten *et al.* [5] introduced FAST (Features from Accelerated Segment Test), a interest point extraction method which considers all pixels on a circle around the current point to decide whether this point is a feature or not. Although no learning is involved in this approach, it can be seen as a manually

designed decision tree. In [6], Rosten *et al.* extend their work such that the output of FAST is learned using a decision tree. Again, pixels on a circle around the center pixel are used in the tree features. In [7], Rosten *et al.* try to improve the repeatability of the interest point detection. In contrast to the previous approaches they consider more test pixels and use simulated annealing to optimize the decision tree with respect to repeatability and efficiency. The optimization is done by randomly modifying an initially learned tree and by checking whether this modification improves repeatability and efficiency.

The core idea of [8] is to take an existing binary-valued decision algorithm as a black box performing some useful binary decision task and to train the WaldBoost [9] classifier as its emulator. WaldBoost is a greedy learning algorithm which finds a quasi-optimal sequential strategy for a given binary-valued decision problem. It combines the AdaBoost [10] algorithm for feature selection and Wald's sequential probability ratio test (SPRT) for finding the thresholds that are used for making the decision. As examples, they learned the Hessian-Laplace and Kadir-Brady saliency detector. However, the resulting emulators are slow and not able to process images at reasonable frame rates.

Considering more high-level interest points, the face detector of Viola *et al.* [11] can also be seen as a detector of interest points, where the interest points are actually faces. In [12], the authors try to classify surface types, *e.g.* planes or valleys, in range data using perceptron trees. This does not fit exactly into the category of interest point detection, but it identifies regions of interest which can be used for similar tasks as interest points. In [13], Lepetit *et al.* train trees such that they output the probability that the considered interest point corresponds to a specific class. Although not intended by the authors, this might be seen as kind of an interest point detector for every class if applied densely on an image. Similar things have been done in [14] using Ferns. Shotton *et al.* [15] use decision trees to estimate body parts and use artificially rendered humans to train their trees.

However, the above-mentioned approaches neither consider learning the detection of interest points in depth data using regression trees, as we do in Sec. 5, nor creating artificial interest point response maps (for training the regression model) that are optimized to increase the detection performance, as we propose in Sec. 5.

3 High Curvature as Baseline Interest Point Detectors

A common approach [16, 17] used for estimating interest points in 3D data is to consider surface curvature and select points where curvature reaches a maximum. In the following, we explain how curvature can be computed using the normal vectors of the surface surrounding the point of interest. For this, we first describe the process for estimating surface normals and then discuss how these estimates can be used to compute a curvature response.

3.1 Normal Estimation

For surface normal estimation from a depth map, we use a modified version of the approach proposed in [18]. They consider the first order Taylor expansion of the depth function $\mathcal{D}(x)$

$$\mathcal{D}(x + \mathrm{d}x) - \mathcal{D}(x) = \mathrm{d}x^\top \Delta\mathcal{D} + \mathcal{O}(\mathrm{d}x^2), \tag{1}$$

where x is the 2D coordinate in the depth map, $\mathrm{d}x$ is a 2D offset, $\Delta\mathcal{D}$ is a 2D depth gradient, and $\mathcal{O}(\mathrm{d}x^2)$ represents higher order terms. To estimate the value of the gradient $\hat{\Delta\mathcal{D}}$ they use 8 neighboring points around the point of interest to create a stack of equations. A neighbor is only considered if the difference in depth is below a certain threshold α. In our experiments we use $\alpha = 5$ cm. From $\hat{\Delta\mathcal{D}}$, one can then compute three 3D-points as

$$X = \boldsymbol{v}(x)\mathcal{D}(x), \tag{2}$$

$$X_1 = \boldsymbol{v}(x + [1,0]^\top)(\mathcal{D}(x) + [1,0]\hat{\Delta\mathcal{D}}), \tag{3}$$

$$X_2 = \boldsymbol{v}(x + [0,1]^\top)(\mathcal{D}(x) + [0,1]\hat{\Delta\mathcal{D}}), \tag{4}$$

which form two vectors $\mathbf{v}_{X\to X_1}$ and $\mathbf{v}_{X\to X_2}$ between X and X_1 as well as X and X_2. The desired normal can be computed from these two vectors by computing the cross-product $\mathbf{n} = \mathbf{v}_{X\to X_1} \times \mathbf{v}_{X\to X_2}$.

In contrast to [18], we compute the position of the 8 neighboring points based on the inverse of the depth of the point of interest instead of using a fixed position. This means, that we take a bigger image region into account for points further away. We do this, since the depth of 3D points located at further distance to the depth sensor is disturbed by stronger noise and discretization effects than that of 3D points close to the sensor.

3.2 Curvature Response

Having computed the normals, we select all neighboring points within a 15×15 pixel image window and project all their normals onto the plane defined by the normal of the point of interest. From these projected normals, we then compute the covariance matrix C and use its second eigenvalue as curvature response.

4 Learning Interest Point Detectors

In this Section we introduce the learning procedure for our proposed interest point detectors based on decision trees. For this, we first present the dataset we used for training (see Sec. 4.1), then introduce the learning procedure for obtaining regression trees which approximate the interest point response estimation process (see Sec. 4.2), and finally, explain the post-processing steps we use in order to obtain the interest points from the interest point response approximation provided by the regression trees (see Sec. 4.3).

Fig. 1. The shown color images present representative frames of each sequence used for training and testing. The first 23 images represent the training sequences while the last 10 images represent the test sequences.

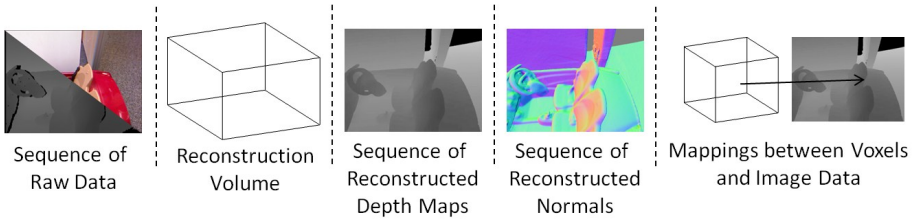

Fig. 2. The different types of data each test and training sequence contains.

4.1 Data Set

In total, we use 33 video sequences captured from the Kinect sensor: 23 sequences for training, and 10 sequences for testing (see Fig. 1). Each video sequence consists of approximately 900 continuously recorded color images and corresponding depth images. Both, the training as well as the test set of sequences depict typical real world scenes expected to be encountered in an office-like scenario, including desks, specialized work spaces, recreational areas as well as meeting rooms. To ensure a fair evaluation, we ensured that the test sequences did not record any of the same volume of 3D world space.

Each of the available sequences contains not only the raw data obtained from the Kinect sensor but also a volumetric reconstruction obtained using the KinectFusion [1] reconstruction system. Fig. 2 shows the different types of data we have available in the sequences. These include the raw depth and image data, the reconstructed volume, synthetic depth and normal maps obtained from the reconstructed volume, and for each frame, a mapping between the volume voxels and the pixels in the images. This volumetric mapping will let us create optimized synthetic interest point response maps as described in Sec. 5.

4.2 Learning using Regression Trees

In the following we describe the process for learning the structure of a binary regression tree that approximates interest point responses from depth maps. Every non-leaf node of the tree uses a depth comparison between two sample positions relative to the point under consideration to decide whether to follow the left or right tree branch emanating down from the node. Every leaf node of the tree stores an interest point response value which is the mean of the responses of all the training pixels that reached that leaf node. This leads to the following parameterization of a node:

$$n = \begin{cases} (f, n_l, n_r) & \text{if } n \text{ is node} \\ (m_c) & \text{if } n \text{ is leaf} \end{cases}, \tag{5}$$

where f is a feature, n_l and n_r are the left and right child node respectively, and m_c is the mean interest point response of the training examples that fall into the corresponding node. The feature f implements the depth comparison between two sample positions and is defined as

$$f = (x_1, y_1, x_2, y_2, t), \tag{6}$$

where $p_1 = (x_1, y_1)^\top$ and $p_2 = (x_2, y_2)^\top$ are the sample positions for the depth values and t is a threshold which is applied on the depth difference $\mathcal{D}(p_1) - \mathcal{D}(p_2)$. The sample points are placed within a $w_D \times h_D$ window. The first sample point p_1 is either placed in the center or randomly within this window around the center point, where both of these possibilities have equal chance. The second sample point p_2 is placed randomly within the window. In our experiments, we use $w_D = h_D = 41$ at a depth of 1 meter and we scale the window according to the depth. The thresholds as well as the offsets of the feature sample positions are selected automatically during the training.

During the learning, we select for every node n_i a feature $f_j \in \mathcal{F}$ and a threshold t_k which best separate the set $\mathcal{E}$ of examples. We consider an example as a quintuple e with

$$e = (q, r, x, y, c), \tag{7}$$

where q is the index of training sequence, r the index of the frame within this sequence, $p = (x, y)^\top$ is the location of the example within the image r of sequence q, and c is the corresponding interest point response value. A feature f_j separates the set $\mathcal{E}$ best if it minimizes

$$\epsilon = v_l \frac{N_l}{N_l + N_r} + v_r \frac{N_r}{N_l + N_r}, \tag{8}$$

where v_r and v_l are the variances of the examples that fall into the left respectively right child node, and N_l and N_r are the corresponding numbers of examples. This variance reduction objective follows the standard entropy minimization strategy used for regression tree learning [19]. In our experiments we sample 1000 feature tests *ie.* $|\mathcal{F}| = 1000$ and select 10 thresholds which are

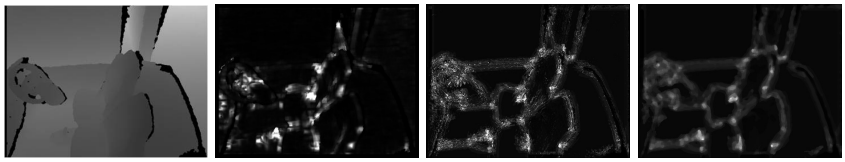

Fig. 3. Left to right: exemplary depth map with corresponding surface curvature map, the unfiltered response obtained from regression trees as well as its median filtered counterpart.

distributed over the range of the possible feature results. The example set $\mathcal{E}$ is created by selecting every second pixel in x and y from approximately 1000 images taken from our training sequences. However, we use an example only if curvature information is available and the depth is not larger than 4 meters.

At test time, for every image position, we follow the path down from the root to the leaf, and return as the result, the response m_c corresponding to the reached leaf. In case of multiple trees, the results of the individual trees are combined together by averaging them.

4.3 Post-processing

Fig. 3 shows the interest point response map approximation obtained from a depth map using a regression tree. As one can easily see it contains a significant amount of salt-and-pepper-like noise which has to be filtered out in order to get a stable response over multiple frames. Therefore, we apply a median filter of size 5×5 and then a Gaussian filter with sigma 3 in order to get distinctive peaks. Finally, the interest points are extracted as maxima of the resulting map.

5 Designing Optimal Interest Point Detectors

In Section , we have shown how random forests can be trained to predict the response of any existing curvature-based interest point detector. This process however, does not in itself lead to better interest points. In this section we show how to compute desirable interest points for training the random forest that may not be obtained from existing methods. For this, we first discuss the properties desired from a good interest point and then show how to compute such response maps from 3D reconstructions of the scenes.

5.1 Optimality Criteria

In the following we define interest points as a set of points in the image coordinate system which fulfil certain specific properties. We will use these properties to select an evaluation criteria. The following list of properties are desirable for a set of scene elements in order to be useful as interest points:

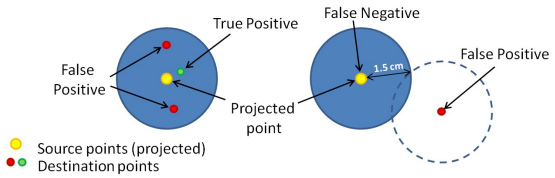

Fig. 4. Illustration of IP repeatability. The evaluation always considers a source frame and a destination frame. All the extracted interest points in the source frame are projected into the destination frame using the provided reconstruction. For every projected source point we search for corresponding destination points within a radius of 1.5 cm. Only the closest match within this radius is considered as a true positive and the others are considered as false positives. If no point is present, we count it as a false negative. If a destination point is not assigned to any projected source point then it is considered a false positive.

- **Sparseness**: there should be only a small number of points in the scene.
- **Repeatability**: the points should be detected in all views of the scene.
- **Distinctiveness**: the area around an interest point should be unique.
- **Efficiency**: points could be estimated efficiently.

While analyzing the criteria, one sees that *sparseness* is hard to evaluate since it is usually defined using a threshold. *Distinctiveness* is highly dependent on the matching method, especially on the construction of the descriptor and thus is difficult to evaluate objectively. *Repeatability* of an interest point, however, is easy to measure since we have access to the reconstruction of the scene depicted in every training/test sequence. This enables us to propagate interest points from one frame to any other frame of the sequence and check the consistency of results.

We also evaluate *repeatability* with respect to the number of extracted interest points (see Sec. 6.1 and 6.2), which provides us with information about the detection performance with respect to *sparseness*. Furthermore, we evaluate *repeatability* with respect to the number and depth of trees in Sec. 6.3 and 6.4. Since the *efficiency* of the presented approach depends on the number and depth of trees used for detecting interest points, this allows us to quantify the trade-off between *efficiency* and *repeatability*. The influence of the tree parameters on the interest point detection performance is discussed in detail in Sec. 6.5.

As a measure for repeatability we use the number of true-positives, false-positives and false-negatives. The estimation of these numbers is described in Fig. 4. Repeatability is computed for a 5 frame difference as well as a 40 frame difference between compared images. This was done to compare repeatability for both, small- as well as wide-baseline matching applications.

5.2 Creating a Response Tailored for High Repeatability

In order to create a response which leads to highly repeatable interest points we make use of the mapping between pixels of the input depth maps and the vox-

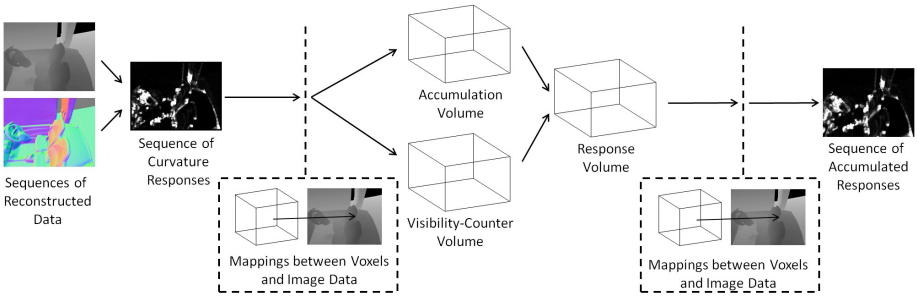

Fig. 5. Illustration of the synthetic response computation.

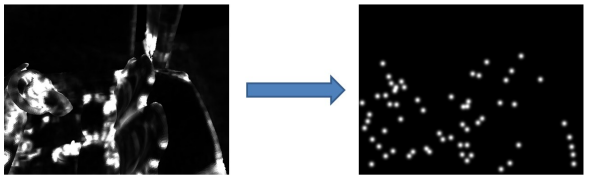

Fig. 6. Computing artificial response maps from accumulated responses.

els of the reconstruction volume, as shown in Fig. 5. For this, we first compute the curvature response for every image of each sequence based on the synthetic surface normals, which are obtained from the reconstruction. These curvature responses are then accumulated in an accumulation volume. A second accumulator volume is used to count how often a voxel was visible in one of the images of the sequence. Finally, we project the resulting accumulation volume into each frame of the sequence and select the best N interest points from the rendered image. Note that this procedure also correctly handles occlusions. The final response map is then created by creating a Gaussian response for each of the selected interest points, as shown in Fig. 6. This artificial response gives high responses for repeatable points and is then approximated using a regression tree. Results for this are given in Sec. 6.2.

6 Results

In the following we evaluate our proposed approach against the baseline method using the test set introduced in Sec. 4.1. As discussed in Sec. 5.1, the evaluation is based on the basis of repeatability of the obtained interest points. If not otherwise mentioned, the evaluations are done with respect to the number of interest points obtained from the detectors. The number of IPs is controlled by changing the threshold which is applied on the response map created by the specific IP detector. Selecting a high threshold results in only a few, but very stable IPs, while choosing a lower threshold increases the number of points. A higher number of points, on the other hand, generally leads to a worse true-positive rate.

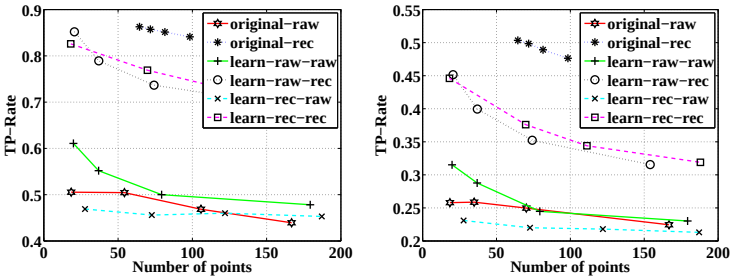

Fig. 7. Comparison of repeatability of detected interest points with respect to different training and test data characteristics. True-positive rate with image pairs with 5 frames difference **(left)** and 40 frames difference **(right)**.

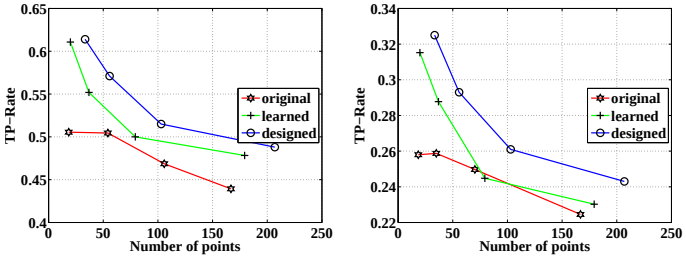

Fig. 8. Comparison of the interest point detector trained using designed response maps (described in Sec. 5 and Fig. 6) with the original curvature-based approach for a 5 frames difference **(left)** and a 40 frames difference **(right)**.

The drop in the true-positive rate with increasing number of obtained interest points is explained by the fact that a lower threshold applied on the response map results in more but less reliable points. The evaluation is conducted using a notebook with a 2.26GHz Intel(R) Core(TM)2 Quad CPU and 4 GB of RAM, where only one core is used for the computation.

6.1 Learning Interest Point Detector Responses

We now describe our experimental results.

Fig. 7 compares the repeatability of results obtained from learned and hand-coded detectors on both, raw and reconstructed data[1]. The results of the curvature-based interest point detector applied on the raw Kinect depth map are annotated by *original-raw*, while those obtained by applying it on the rendered depth map (obtained from the 3D model KinectFusion system) are annotated by *original-rec*. As expected, *original-rec* results are better than *original-raw* because the rendered depth maps have less noise, are smoother, and do not suffer from missing data.

[1] This section deals only with the input data. The use of designed response maps (the training labels) as described in Sec. 5 will be analyzed below in Sec. 6.2.

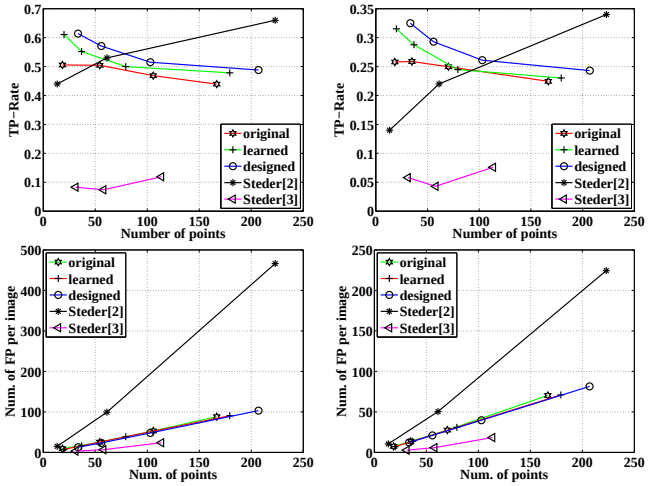

Fig. 9. Comparison of the interest point detector trained using designed response maps (described in Sec. 5 and Fig. 6) with the original curvature-based approach, its learned counterpart, as well as the detectors of Steder *et al.* [2, 3]. Results are for a 5 frames difference **(left)** and a 40 frames difference **(right)**.

We now analyze the effect of using raw depth (*learn-raw-...*), as well as rendered depth (*learn-rec-...*) for training the regression forest. These two different cases are then evaluated on raw depth data (*learn-...-raw*) as well as on reconstructed depth data (*learn-...-rec*). This leads to a total of four different possibilities. Note that the curvature response used for training the regression trees is always computed from the reconstructed data given by the KinectFusion system. We are changing here only the data on which the tree features are evaluated in order to decide on the split. As one would expect, training the regression tree using the data type it is later applied to gives superior results compared to training it from a different type of data.

Comparing the IP repeatability of the original surface curvature estimation approach (*original-raw*) with our proposed approximation using regression trees (*learn-raw-raw*) in Fig. 7, one can see that our approach is not only able to approximate the behaviour of the curvature-based IP detector with respect to repeatability, but it even gives better results. The improvement can be explained by the fact that we use the curvature estimated from the reconstructed data instead of the raw data for training our regression trees.

6.2 Learning Designed Response Maps

In Fig. 8 we compare the results obtained by a regression forest trained using the artificial IP response maps (which we introduced in Sec. 5 and Fig. 6) with the results of the original interest point detector based on curvature as well as the regression forest trained using its IP responses. Our results show that

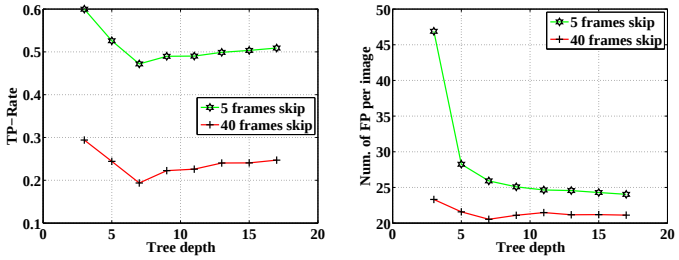

Fig. 10. Evaluation of the influence of the depth of trees on the true-positive rate and the number of false positives per image.

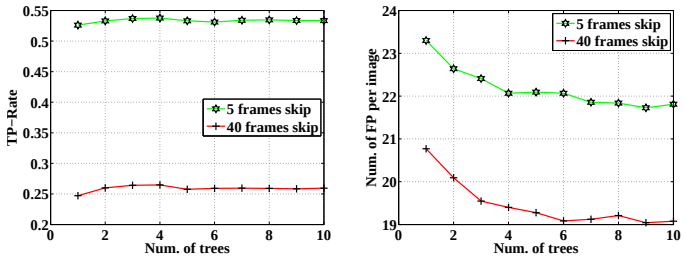

Fig. 11. Evaluation of the influence of the number of trees on the true-positive rate and the number of false positives per image.

it is possible to train a regression forest to output such desired IPs resulting in better repeatability performance compared to the curvature-based interest point detector.

In Fig. 9 we additionally compare to [2] and [3]. Although [2] tends to have a higher true-positive rate for large numbers of points, it also shows a higher number of false-positives. The approach presented in [3] on the other hand has a very low number of false positives, but also a far worse true-positive rate.

6.3 Depth of Trees

Fig. 10 evaluates the influence of the depth of a regression tree on the performance of the interest point detector. While the true-positive rate drops until a depth of 7 and then increases slowly with increasing depth, the number of false positives per image is high for trees with a depth less than 7.

6.4 Number of Trees

As Fig. 11 indicates, the number of trees has only a small effect on the resulting detection performance. While the true-positive-rate stays almost constant, the number of false-positives per image drops only by a value of less than 2 when going from a single tree up to ten trees. This can be explained by the applied

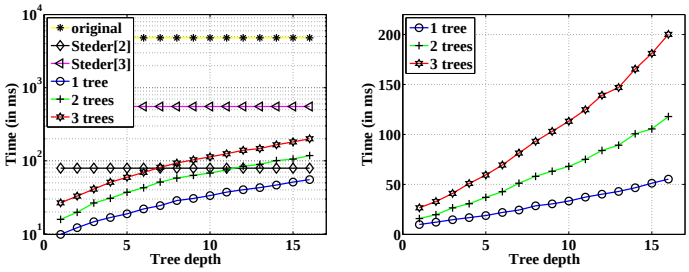

Fig. 12. Timings for interest point detection. **Left:** comparison of the original curvature-based approach with learned interest point detectors using different numbers of trees and tree depths as well as the detectors of Steder *et al.* [2, 3]. **Right:** zoomed-in version where we only the learned approaches are compared. For our approach we used the regression trees learned from the designed response maps (see Sec. 5), including the post-processing step (see Sec. 4.3).

filters (see Sec. 4.3), which are used to remove different types of noise from the obtained response maps, since adding more trees to the evaluation process has a similar effect on the response. Note, however, applying the described filters is computationally more efficient than using a large number of trees (see Sec. 6.5).

6.5 Processing Time

In Fig. 12 we compare the computation time of the original curvature-based IP detector with our random forest based detector using different number of regression trees with different depths and with the detectors of Steder *et al.* [2, 3]. As one can see, the processing time for our proposed approach using regression trees is much less than for the original approach based on surface normals. It is also faster than the approach of Steder *et al.* [3]. The single tree or low depth forest variants of our approach are faster than [2] also. Note that for the processing time of the original approach as well as for the approaches of Steder *et al.* [2, 3] we estimated only a single value since it does not depend on the depth of a tree. We visualize it as a line for better comparison. For the evaluation of [3], we used an open-source implementation which is available in the Point Cloud Library[2].

7 Conclusion

In this paper, we presented a novel regression forest based approach to efficiently detect interest points in depth maps. Our experimental results show that a curvature-based interest point detector can be approximated using the regression forest model. Furthermore, we show that by using a reconstruction of the available scenes we can create an improved interest point detector which gives interest points with higher repeatability.

[2] `http://www.pointclouds.org`.

Acknowledgements. The authors would like to thank Rasmus Kyng, Shahram Izadi, David Kim, Dave Molyneaux, and Otmar Hilliges for the inspiring discussions and for help in obtaining the training and test data.

References

1. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: ISMAR. (2011)
2. Steder, B., Grisetti, G., Burgard, W.: Robust place recognition for 3D range data based on point features. In: ICRA. (2010)
3. Steder, B., Rusu, R.B., Konolige, K., Burgard, W.: Point feature extraction on 3D range scans taking into account object boundaries. In: ICRA. (2011)
4. Unnikrishnan, R.: Statistical approaches to multi-scale point cloud processing. (2008)
5. Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: ICCV. (2005)
6. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: ECCV. (2006)
7. Rosten, E., Porter, R., Drummond, T.: Faster and better: A machine learning approach to corner detection. PAMI (2010)
8. Šochman, J., Matas, J.: Learning a fast emulator of a binary decision process. In: Proceedings of the 8th Asian Conference on Computer vision - Volume Part II. (2007) 236–245
9. Sochman, J., Matas, J.: Waldboost - learning for time constrained sequential detection. In: CVPR. (2005)
10. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. In: Machine Learning. (1999) 80–91
11. Viola, P., Jones, M.: Fast and robust classification using asymmetric adaboost and a detector cascade. In: Advances in Neural Information Processing System 14, MIT Press (2001) 1311–1318
12. Foresti, G.: Invariant feature extraction and neural trees for range surface classification. IEEE Transactions on Systems, Man, and Cybernetics (2002)
13. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. PAMI (2006)
14. Özuysal, M., Calonder, M., Lepetit, V., Fua, P.: Fast keypoint recognition using random ferns. PAMI (2010)
15. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from a single depth image. In: CVPR. (2011)
16. Stückler, J., Behnke, S.: Interest point detection in depth images through scale-space surface analysis. In: ICRA. (2011)
17. Gelfand, N., Mitra, N.J., Guibas, L.J., Pottmann, H.: Robust global registration. In: Eurographics Symposium on Geometry Processing. (2005)
18. Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: ICCV. (2011)
19. Criminisi, A., Shotton, J., Konukoglu, E.: Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. In: Foundations and Trends in Computer Graphics and Vision. (2012)