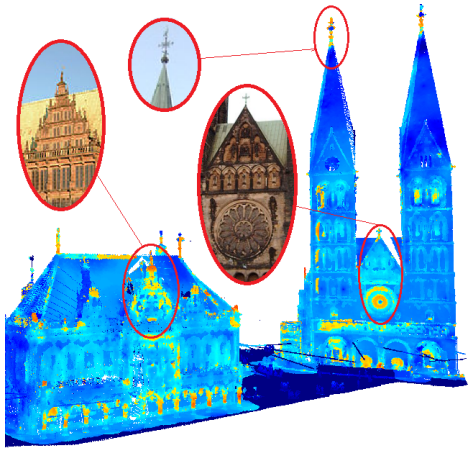
Saliency Detection in Large Point Sets

Elizabeth Shtrom
Technion
lizas@tx.technion.ac.il

George Leifman
Technion
gleifman@tx.technion.ac.il

Ayellet Tal
Technion
ayellet@ee.technion.ac.il

(a) Bremen city [3], colored according to height



(b) The best viewpoint, colored according to our saliency

Figure 1. **Detecting the salient features in a point set of an urban scene.** Given the noisy point set of the Bremen center (a), containing 12M points, our algorithm computes its saliency. The most salient points, such as the rosette and the crosses on the towers, are colored in yellow and red. The least salient points, belonging to the floor and the feature-less walls, are colored in blue. Our saliency map is utilized for finding the most informative viewpoint (b), displaying the most interesting buildings of the city – St. Peter's Cathedral and Bremen's town hall. In (b) we also show images of the parts that were found to be the most salient.

Abstract

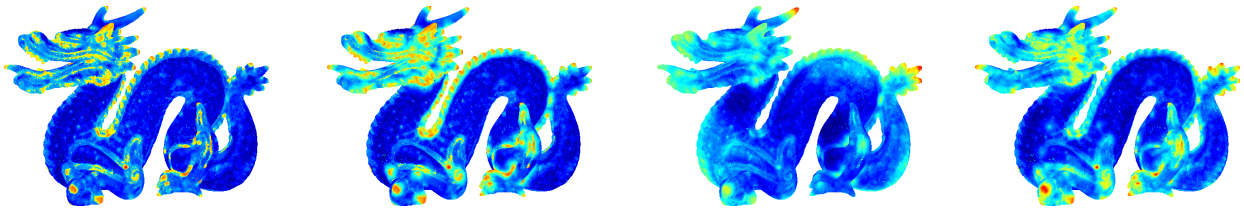
While saliency in images has been extensively studied in recent years, there is very little work on saliency of point sets. This is despite the fact that point sets and range data are becoming ever more widespread and have myriad applications. In this paper we present an algorithm for detecting the salient points in unorganized 3D point sets. Our algorithm is designed to cope with extremely large sets, which may contain tens of millions of points. Such data is typical of urban scenes, which have recently become commonly available on the web. No previous work has handled such data. For general data sets, we show that our results are competitive with those of saliency detection of surfaces, although we do not have any connectivity information. We demonstrate the utility of our algorithm in two applications: producing a set of the most informative viewpoints and suggesting an informative city tour given a city scan.

1. Introduction

Saliency detection has attracted a lot of attention in computer vision, with myriad applications. Examples include object recognition, similarity, registration, matching, downsampling, and visualization. Most of the works concentrate on images [2, 9, 12, 14, 28] and videos [11, 19, 22, 25]. Less work addresses saliency of 3D surfaces [5, 7, 17] and only a few papers handle point sets [1, 16].

Due to the fast development of 3D data acquisition techniques, 3D point sets have become widely spread. The traditional way to process and analyze point sets is to reconstruct the underlying surface and then apply well-known methods. However, when the size of the data is large, such methods are computationally expensive, if not impractical. Moreover, the data acquired is usually distorted by noise, which makes reconstruction challenging.

Extending the existing techniques of saliency detection for 3D surfaces to operate directly on large point sets is not trivial. This is so, not only due to the large size of the



Low-level distinctness ($\mathcal{D}_{low}$) Low-level association ($\mathcal{A}_{low}$) High-level distinctness ($\mathcal{D}_{high}$) Final saliency map ($\mathcal{S}$)

Figure 2. **Algorithm outline.** The low-level distinctness is first computed, identifying small geometric features, such as the teeth and the spikes on the head and on the back of the dragon. Then, association is applied, grouping salient points and emphasizing the dragon's facial features. Next, the high-level distinctness procedure detects larger regions, such as the tail and the mouth. Finally, the maps are integrated to produce the final saliency map.

data, but also due to the absence of topological information regarding the point connectivity. In particular, the existing algorithms use geodesic distances and simplification and hence, cannot be applied to unorganized point sets.

Similarly to previous saliency detection algorithms, which operate on other types of data, our saliency detection algorithm is based on distinctness. The challenge here is to look for a distinctness definition that suits point sets and is computationally efficient. Moreover, taking into account the hierarchical human visual perception mechanism [10], we identify globally-distinct features in a multi-level manner. The low levels account for the detection of delicate features, while suppressing 3D textures. The high levels, on the other hand, identify entire unique regions. Additionally, according to [29], we need to consider the fact that visual forms may possess one or several centers of gravity about which the form is organized. Therefore, points that are close to the foci of attention are more salient than faraway points.

We propose a novel algorithm that detects salient points in a 3D point set (Figure 1), by realizing the considerations mentioned above. We discuss a compact point descriptor that characterizes the geometry of a point's neighborhood. For each level, a different neighborhood size is used. Additionally, to take the distance to foci into account, we adjust the point distinctness according this distance.

Our algorithm is general and competes favorably with state-of-the-art techniques for saliency detection of general objects, which typically consist of less than a million points. However, it also copes with point sets of urban scans, containing tens of millions of noisy points.

We demonstrate the utility of our saliency maps in two applications. The first application produces a set of the most informative viewpoints for a given point set, maximizing the accumulative viewed saliency. Second, for urban scenes, we construct an informative tour in the city, which maximizes the interesting area viewed by the tourist.

Our contributions are thus twofold. First, we propose a novel algorithm for detecting the salient points in large point sets (Sections 3-5). Second, we demonstrate the applicability of our results (Section 6).

2. General Approach

Given a point set, our goal is to efficiently compute its saliency map. Since human attention is drawn to differences, we look for points whose neighborhood is unique geometrically with respect to other neighborhoods. To capture distinctness, Section 3 discusses a point's descriptor that characterizes the geometry of a point's neighborhood. A point is considered distinct if its descriptor is dissimilar to all other point descriptors of the set. The choice of a good dissimilarity measure between the descriptors is of high importance and therefore is also discussed.

Taking into account the fact that object recognition is performed hierarchically, from local representations to abstract ones, our saliency detection algorithm analyzes a scene hierarchically. In particular, distinctness should be computed in a multi-level manner. In practice, we found that two levels suffice. In the low level, delicate unique features are highlighted, while in the high level, the distinctness of entire semantic parts is detected.

Finally, we wish to look for salient regions, rather than for isolated points [29]. This consideration follows the human tendency to group close items together. Therefore, we apply point association, which regards the regions near the foci of attention as more interesting than faraway regions.

Figure 2 illustrates our approach. First, low-level distinctness, $\mathcal{D}_{low}$, is computed, highlighting small features, such as the teeth and the spikes on the head and on the back of the dragon. Then, we apply association, $\mathcal{A}_{low}$, which increases the saliency in the neighborhood of the most distinct points. Next, high-level distinctness, $\mathcal{D}_{high}$, is computed, detecting large features, such as the tail and the mouth. Finally, the above three components are integrated into the final saliency map, $\mathcal{S}$, defined for a point p_i as follows:

$$\mathcal{S}(p_i) = \frac{1}{2}\left(\mathcal{D}_{low}(p_i) + \mathcal{A}_{low}(p_i)\right) + \frac{1}{2}\mathcal{D}_{high}(p_i). \quad (1)$$

3. Point Descriptor and Dissimilarity Measure

In order to determine whether a point is distinct, we should first characterize it by a descriptor. Then, we should

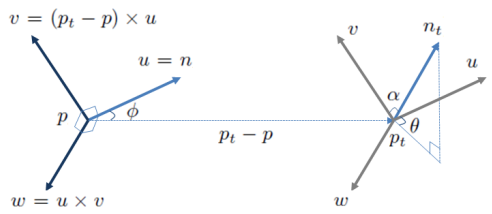


Figure 3. **Darboux frame**. To compute the relative angular differences between two points, p and p_t, and their normals, n and n_t, a fixed coordinate frame uvw is defined at one of the points.

define a dissimilarity measure that compares these descriptors. Finally, using the dissimilarities of a point to other points in the set, the distinctness is computed. Below we discuss our point descriptor and our dissimilarity measure. The next section will define distinctness.

Point descriptor: We seek a descriptor that has good expressive power of the local shape geometry. Moreover, due to the size of the data, it should also be compact and efficient to compute. Furthermore, the descriptor should be invariant to rigid transformations and robust to noise, as the data is typically very noisy.

We have experimented with various descriptors and found that a slightly-modified *Fast Point Feature Histogram (FPFH)* [24] best suits our problem. We describe it and our modification hereafter.

The FPFH captures the relative angular directions of the normals with respect to one another. To compute it for a point p, the neighbors enclosed in a sphere with radius r are first found. Then, for every point p_t inside the sphere, we define a Darboux uvw frame (Figure 3): $u = n, v = (p_t - p) \times u, w = u \times v$, where n is the estimated normal of p. The following angular variations are then computed:

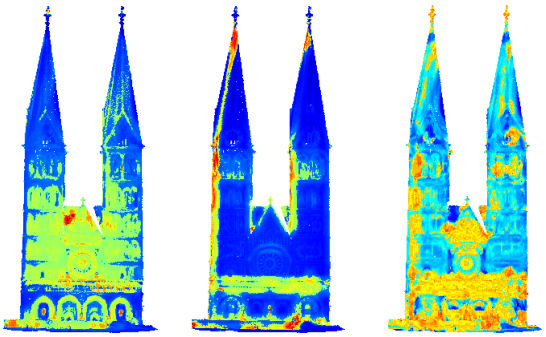
$$\begin{aligned} \alpha &= v \cdot n_t, \\ \phi &= u \cdot \frac{p_t - p}{||p_t - p||}, \\ \theta &= \arctan(w \cdot n_t, u \cdot n_t). \end{aligned} \tag{2}$$

Next, a histogram of the quantized angles (α,ϕ,θ) between p and its neighbors is computed, and termed the *Simplified Point Feature Histogram (SPFH)*. Finally, FPFH is calculated using the SPFH of a given point and a weighted sum of the SPFH of its neighbors:

$$FPFH(p) = SPFH(p) + \frac{1}{K} \sum_{k=1}^{K} \frac{SPFH(p_k)}{||p - p_k||}, \tag{3}$$

where K is the number of neighbors of p.

Since the FPFH descriptor is not invariant to reflection, which is crucial in our case, we use a modified FPFH, where the triple (α, ϕ, θ) gets the absolute values.



(a) Spin Images (b) SHOT (c) Our descriptor

Figure 4. **Comparison to other point descriptors:** The low-level distinctness produced using our descriptor (Equation 6) outperforms others. It detects the fine features, such as the rosette, the crosses on the top of the towers and the sculptures in the windows.

This descriptor satisfies our requirements. It is compact and quick to compute. Moreover, it is invariant to rigid transformations and robust to noise.

Comparison to alternative descriptors: We compared our descriptor to SHOT [27] and to Spin Images [13]. Figure 4 shows the low-level distinctness produced using the three descriptors. It can be noticed that FPFH competes favorably with the other descriptors, detecting the fine distinctive features, such as the rosette, the crosses on the top of the towers and the sculptures in the windows. This can be explained by the fact that the other descriptors are highly dependent on accurate normal estimation at the query point, which is not the case for our noisy data. Conversely, in FPFH the normals in the whole neighborhood are considered, reducing the dependency on a specific normal.

An additional advantage of our descriptor is its compactness (33 bins), which makes it applicable to extremely large data sets. The Spin Images descriptor is twice as big as the FPFH (64 bins), while SHOT is ten times bigger than the FPFH (352 bins). The nearest neighbor calculation using FPFH is 20 times faster than using SHOT and 6 times faster than using Spin Images. Therefore, the large size of our data renders the use of SHOT or Spin Images impractical.

Dissimilarity measure: We seek a dissimilarity measure that is robust to noise and to sampling density. In addition, its computation should be fast, due to the large data the algorithm is designated to work on.

To address the performance requirement, we are using the K-D Tree data structure to organize the descriptors. Since cross-bin metrics cannot be used within a K-D Tree, we considered several metrics which can, including L_1, L_2, Manhattan distance and Chi-Square (χ^2). We found that χ^2 is superior to the others, being less affected by noise, while still being sensitive to delicate features.

Formally, given two points, p_i and p_j, and their FPFH

descriptors, the χ^2 dissimilarity measure between them is:

$$D_{\chi^2}(p_i, p_j) = \sum_{n=0}^{N} \frac{\left(FPFH_n(p_i) - FPFH_n(p_j)\right)^2}{FPFH_n(p_i) + FPFH_n(p_j)}, \quad (4)$$

where N is the number of bins in the FPFH and $FPFH_n$ denotes the n^{th} bin of the histogram.

4. Hierarchical Saliency Computation

Our goal is to compute saliency based on the dissimilarity between the descriptors, discussed in the previous section. Following human visual perception, we analyze both the unique delicate features, as well as the large meaningful regions of the scene.

First, we identify the low-level distinctness that highlights the fine details. Then, we use this information in the high-level stage, where entire significant parts of the scene are detected. We use a small neighborhood for the low-level distinctness and a large neighborhood for the high-level distinctness. Without loss of generality, we assume that our point sets are normalized to fit in a unit sphere.

Low-level distinctness: A point p is distinct when it differs from the other points in its appearance. This is usually realized by looking for point descriptors whose dissimilarity to other descriptors is high. However, in the low-level stage, where the point descriptor is calculated on a small neighborhood, this consideration is insufficient. Similar points that are far away indicate the existence of a 3D texture and thus, the distances between the points are important as well. Inspired by [9], a point is distinct when the points similar to it are nearby and less distinct when the resembling points are far away. Therefore, the low-level dissimilarity measure is proportional to the difference in geometry and inverse proportional to the Euclidian distance:

$$d_L(p_i, p_j) = \frac{D_{\chi^2}(p_i, p_j)}{1 + ||p_i - p_j||}. \quad (5)$$

In practice, computing this dissimilarity between all the points of the point set is too expensive. It suffices to consider only points that are highly similar to the query point. Therefore, for every point p_i, we search for a set of points $\mathbb{P}$ for which $D_{\chi^2}(p_i, p_j) \leq d_{min}, \forall j$, using our K-D tree data structure. In our implementation the threshold d_{min} is 1% of the maximal distance between the descriptors.

Finally, a point p_i is distinct when $d_L(p_i, p_j)$ is high $\forall p_j \in \mathbb{P}$. Thus, the low-level distinctness value of point p_i is defined as:

$$\mathcal{D}_{low}(p_i) = 1 - \exp\left(-\frac{1}{|\mathbb{P}|}\sum_{p_j \in \mathbb{P}} d_L(p_i, p_j)\right). \quad (6)$$

Point association: Detecting low-level distinctness usually results in isolated points. However, people tend to group close items together, since visual forms possess one or several focus points about which the form is organized. We model this effect, similarly to [9], by defining a fraction (20% in our implementation) of points having the highest low-level distinctness, as focus points. Let p_{fi} be the closest focus point to p_i and $\mathcal{D}_{foci}(p_i)$ be the low-level distinctness of p_{fi}. The association of point p_i is defined as ($\sigma = 0.05$):

$$\mathcal{A}_{low}(p_i) = \mathcal{D}_{foci}(p_i) \cdot \exp\left(-\frac{||p_{fi} - p_i||^2}{2\sigma^2}\right). \quad (7)$$

High-level distinctness: To evaluate the distinctness of entire regions, we compute our descriptors on large neighborhoods. For cases where no prior information on the data is available, we pick the neighborhood to be a sphere with radius 0.1. For city scans, we set the radius to be half the height of the tallest building.

When the descriptors are computed on large neighborhoods, the neighborhoods of close points almost fully overlap, leading to very similar descriptors. Therefore, we would like to decrease the contribution of nearby points and consider a point distinct when it is dissimilar to far points.

This is achieved by logarithmic weighting, which reduces the contribution of very close points, without affecting faraway points. In particular, we define the high-level dissimilarity measure between p_i and p_j as:

$$d_H(p_i, p_j) = D_{\chi^2}(p_i, p_j) \cdot \log(1 + ||p_i - p_j||). \quad (8)$$

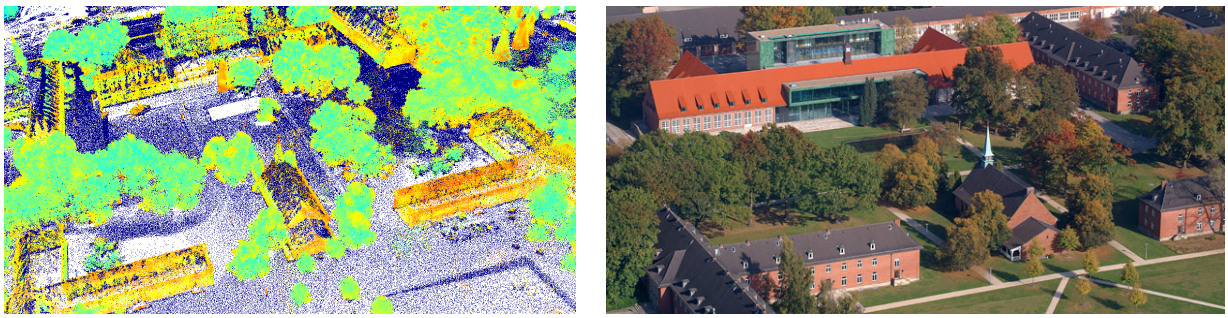
Finally, high-level distinctness is defined as:

$$\mathcal{D}_{high}(p_i) = 1 - \exp\left(-\frac{1}{|\mathbb{P}|}\sum_{p_j \in \mathbb{P}} d_H(p_i, p_j)\right). \quad (9)$$

Since the high-level distinctness depends on the low-level distinctness, the descriptor of each point is computed by considering only 10% of the points with the highest low-level distinctness.

5. Results

Our algorithm is designed to run on extremely large data sets. An example of such data is city scans, which have recently become prevalent. We are not aware of any related work that handles saliency of such huge data. This section demonstrates our results on this type of data. Moreover, to assess the quality of our results, we compare them to those produced by surface saliency detection algorithms. In this case, we show that our results are competitive, despite the lack of connectivity information.



(a) Our saliency for the campus (zoom-in) (b) An image of the campus from Google maps

Figure 5. **The saliency for the Jacobs University campus (15M points)**. The buildings are salient and therefore are colored in orange. The trees, of which there are many, are less salient and are colored in green. The floor is blue.

Saliency in urban scenes: Urban point sets usually consist of millions of noisy points, which are generated by merging multiple range scans. We ran our algorithm on two such point sets, the city center of Bremen and the Jacobs University campus (Figures 1, 5), which were scanned by a Riegl VZ-400 laser scanner [3].

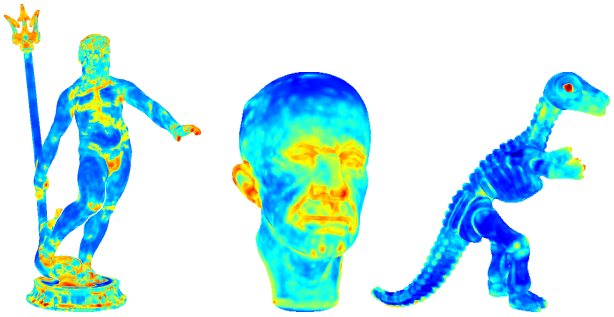
Figure 1 shows our saliency map for the city center of Bremen. Our high-level distinctness identifies the entire facades of the most interesting buildings: the St. Peter's Cathedral and Bremen's town hall. The low-level distinctness highlights the fine details of the buildings, such as the rosette on the Cathedral, the crosses on the towers, and the small statues on the roof.

Figure 5 shows our saliency for the Jacobs University campus. The buildings are found salient and therefore are colored in orange. The trees, which are similar in their appearance and of which there are many, are less salient and are thus colored in green. The floor appears in blue.

General 3D objects: Hereafter, we demonstrate the results of our algorithm on 3D objects that contain up to millions of points. In the case where the objects are represented as surfaces, we use only their vertices as the input to our algorithm, ignoring the connectivity. Figure 6 demonstrates that our algorithm detects the "expected" salient regions, such as the fork of Neptune and the fish next to his feet, and the facial features of Max Planck and the dinosaur.

Qualitative evaluation: In order to assess the quality of our algorithm, we compare our results to those of saliency detection of surfaces. Unlike the case of surfaces, we do not have information regarding the connectivity between the points, the exact normals, the ability to compute geodesic distances, etc. Nevertheless, we show below that our approach is competitive, while being more efficient.

In particular, we compare our method to the recent work of [18]. As can be seen in Figure 7, for small models, like the frog, the results are similar, as both methods detect the facial features and the limbs. However, when running on models with hundreds of thousands of points, our method



2M points 204K points 129K points

Figure 6. **Saliency for large point sets.** Our algorithm detects the "expected" salient regions, e.g., the fork of Neptune and the fish near his feet and the facial features.

produces better saliency maps, detecting fine features, such as the delicate relief features on the bowl and fins of the fish. This can be explained by the fact that in [18] the models are simplified as a preprocessing step, as their algorithm cannot run on large data. Conversely, for our algorithm, these models are considered small.

Quantitative evaluation: We evaluated our algorithm on the benchmark of [6], whose goal is to evaluate the detection of interest points on surfaces vs. human-marked points. Our algorithm outperforms other methods [4, 8, 17, 20, 21, 26] for two error measures out of three. For the third error measure, our method is similar to the results of the other methods. The full description, settings and results are available in the supplementary material.

Complexity analysis: The complexity of the distinctness computation depends on that of the FPFH and on that of finding the K-nearest neighbors. It is $O(kn \log n)$, where n is the number of points and k is the number of neighbors used. For instance, the actual running time on Igea (134K points) is 2 minutes. On the Bremen cathedral (627K points), the running time is $\sim$1 hour, of which the FPFH calculation takes 13 minutes and the distinctness computation takes 48 minutes.

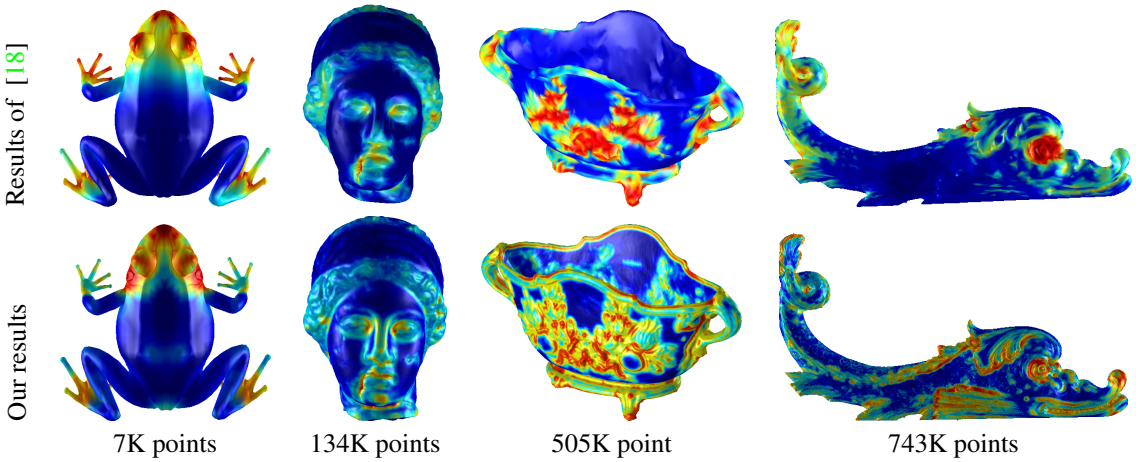


Figure 7. **Comparison with [18].** Our results (bottom row) compete favorably with those of [18] (top row). Our saliency is computed based on the vertices, ignoring connectivity information, which is used by [18]. For small data, like the frog, the results are similar, detecting the facial features and the limbs. However, for larger models, our method produces better saliency maps, detecting fine features, such as delicate relief features on the bowl, and the fins of the fish. This is due to the fact that our algorithm is efficient enough to work on the full objects without simplifying them, losing the fine details.

6. Applications

We demonstrate the utility of our saliency in two applications. First, we propose a technique for producing a set of the most informative viewpoints of the data. Second, given urban data, we construct an informative tour of the city, which maximizes the saliency viewed along the path.

Viewpoint selection: Given a point set, our goal is to automatically produce a set of the most informative viewpoints. The idea is to maximize the accumulative saliency viewed by the set of viewpoints. These are generated iteratively, adding each time a viewpoint with the maximal additional saliency.

Our algorithm proceeds as follows. First, a set of candidate camera locations is produced. For city scans, we use the scanning locations when available. Otherwise, the candidates are generated by uniformly sampling the city at an average human height.

Next, the viewpoints are produced by placing a camera at each candidate location and rotating it to cover all the viewing directions. Each viewpoint is associated with a set V_i containing the points visible by the camera. The visible points are found by performing frustum culling and then removing the hidden points using the HPR operator of [15].

For each candidate viewpoint and its associated set V_i, we calculate the amount of saliency it views by:

$$\bar{S}(V_i) = \sum_{p_j \in V_i} \mathcal{S}(p_j) \cdot w_i(p_j), \tag{10}$$

where $\mathcal{S}(p_j)$ is the saliency of p_j, computed by Equation (1). It is weighted according to:

$$w_i(p_j) = \frac{\cos(\beta_{ij})}{(1 + ||L_i - p_j||)}, \tag{11}$$

where L_i is the camera location and β_{ij} is the angle between the normal at p_j and the viewing direction $L_i - p_j$.

The first viewpoint selected is the one having the maximal saliency (Equation 10). Then, similarly to [18], we add a viewpoint, which jointly with the previously-selected viewpoints, maximizes the viewed saliency. We define the added visible saliency contributed by the viewpoint V_i as:

$$\delta(V_i) = \sum_{p_j \in V_i} \mathcal{S}(p_j) \max\left(w_i(p_j) - w_{max}(p_j), 0\right), \tag{12}$$

where $w_{max}(p_j)$ is the maximal weight assigned to p_j by any of the viewpoints selected so far. The viewpoint that maximizes δ is added to the previously-selected viewpoints.

We keep adding viewpoints until the accumulated viewed saliency is at least 30% of the saliency viewed by all the viewpoints. The final number of viewpoints is dynamic and depends on the city's geometry.

Figure 8 shows the viewpoints found by our algorithm. They indeed capture the most famous buildings of the city. The St. Peter's Cathedral and Bremen's town hall. For an additional example, see the supplementary material.

We are not aware of any previous work that generates informative viewpoints for urban scans. Therefore, we compare our results to those of the viewpoint selection algorithm of surfaces [18]. As demonstrated in Figure 9, in most of the cases the resulting viewpoints are similar. However, for some cases our results are better. For example, for the head of Igea, both algorithms choose a side-view, but our view presents the side with the salient scar near the mouth. For the bowl, our viewpoint is more natural.

Producing the most informative tour: Given a point set of an urban scene and its saliency map, our aim is to suggest

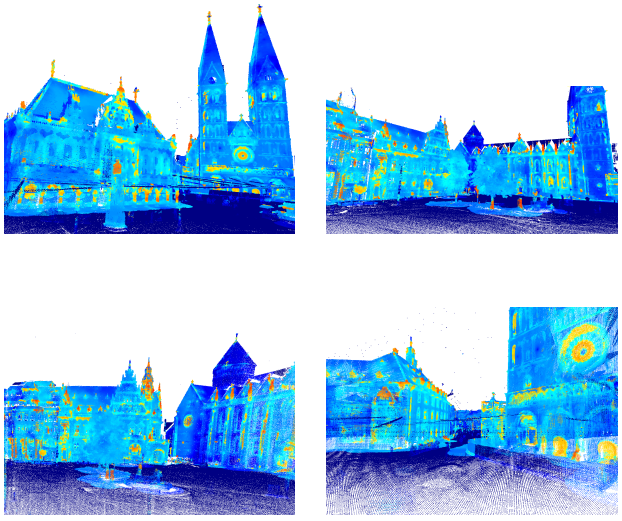


Figure 8. **Viewpoint selection.** The most informative viewpoints generated by our algorithm indeed capture the most interesting buildings of Bremen from various angles.

an informative tour of the city. The idea is to maximize the area of the viewed salient regions along a path.

Our algorithm consists of five steps. First, we compute a set of candidate locations and pick a subset, L_s, of the most salient locations, similarly to viewpoint selection. We stop when at least 75% of the total saliency is viewed by the candidates. Second, as shown in Figure 10(a), we find the shortest closed path PL passing through the points in L_s, creating segments $e_i = (l_i, l_{i+1})$. Third, we generate a path from l_i to l_{i+1}, by replacing e_i as follows. For a small step size, we consider three directions to advance: one towards l_{i+1} and two rotated by $\pm 30°$ from this direction. We choose the one that maximizes the viewed saliency. Finally, since the resulting path is closed, we remove the segment of the path that views the least average saliency. See Figure 10(b).

We note that even though we do not consider obstacle avoidance explicitly, our algorithm avoids obstacles in practice. This can be explained by the fact that we weigh our saliency according to the viewing angle. Consequently, when approaching an obstacle, the value of the cosine in Equation 11 decreases, thus reducing the saliency of viewed points. Moreover, as we get closer to an obstacle, more points become hidden by it, which also decreases the viewed saliency. Hence, our algorithm usually avoids proceeding in the direction of the obstacles.

7. Conclusion

This paper has studied saliency detection for 3D point sets. Our saliency detection algorithm is based on finding the distinct points, using a multi-level approach. It is very efficient and therefore can be applied to huge point sets, such as urban scenes. The efficiency is achieved without sacrificing the detection quality. Our approach is competitive, and often outperforms, approaches that handle surfaces, though it does not rely on connectivity information.

Finally, we demonstrate the utility of our saliency in two applications: selecting a set of informative viewpoints and producing an informative tour in an urban environment.

Acknowledgements: We thank Dorit Borrman from Bremen Jacobs University for providing the 3D scans of the cities and for her technical support. We used the PCL library [23]. This research was supported in part by the Is-

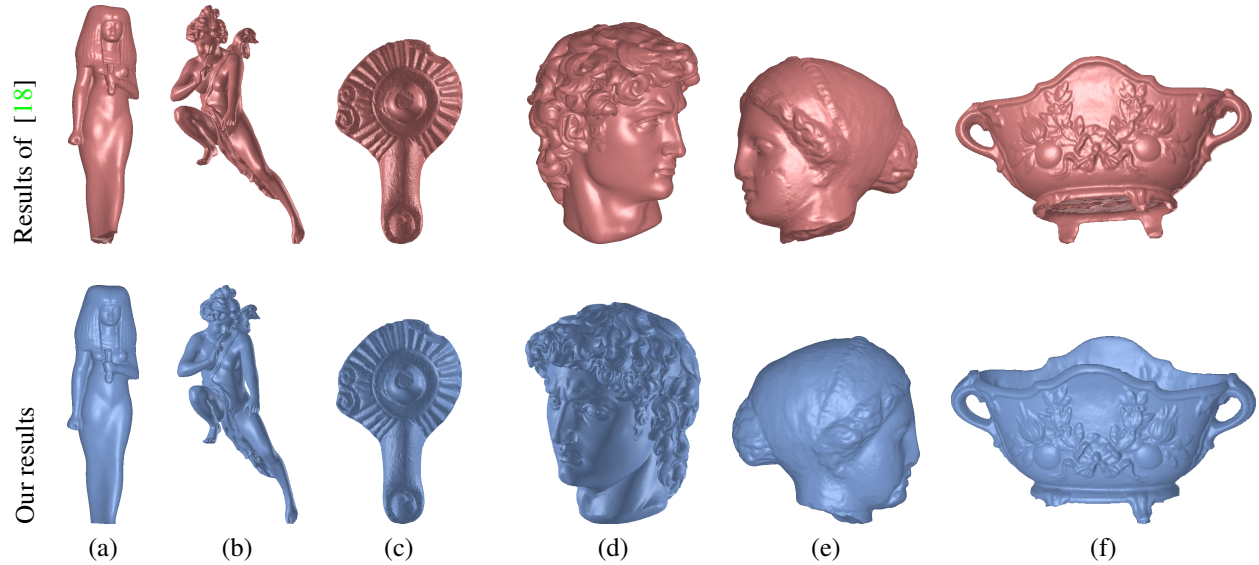
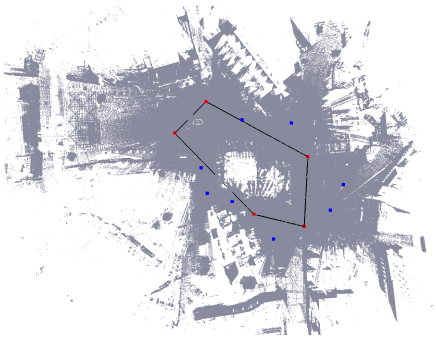
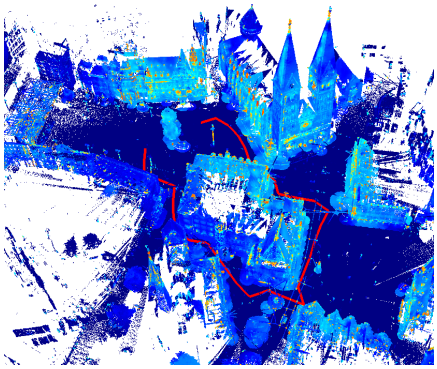


Figure 9. **Comparison of our viewpoints to those of [18].** In most cases, (a-c), our viewpoints (bottom row) are similar to those of [18] (top row). In some cases, (d-f), our results are superior. For the head of Igea, our viewpoint shows the scar near the mouth. For the bowl, our viewpoint is more natural. For David, similarly to [18] we get the 3/4 view, but we see also the other eye.



(a) The candidate points (red and blue) and the shortest path passing through the points in L_s (colored in red)



(b) The generated path

Figure 10. **The recommended tour for Bremen city center.** Walking along the suggested path leads the tourist through all the interesting sites of the center without passing through the walls.

rael Science Foundation (ISF) 1420/12, the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI), Minerva, and the Ollendorff foundation.

References

[1] O. Akman and P. Jonker. Computing saliency map from spatial information in point cloud data. In *Advanced Concepts for Intelligent Vision Systems*, pages 290–299, 2010. 1

[2] A. Borji, S. D. N., and L. Itti. Salient object detection: A benchmark. In *ECCV*, pages 414 –429, 2012. 1

[3] D. Borrmann and J. Elseberg. http://kos.informatik.uni-osnabrueck.de/3Dscans. 1, 5

[4] U. Castellani, M. Cristani, S. Fantoni, and V. Murino. Sparse points matching by combining 3d mesh saliency with statistical descriptors. *CGF*, 27(2):643–652, 2008. 5

[5] X. Chen, A. Saparov, B. Pang, and T. Funkhouser. Schelling points on 3D surface meshes. *ACM Transactions on Graphics*, 31(4):29, 2012. 1

[6] H. Dutagaci, C. Cheung, and A. Godil. Evaluation of 3D interest point detection techniques via human-generated ground truth. *The Visual Computer*, 28:901–917, 2012. 5

[7] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics*, 25(1):150, 2006. 1

[8] A. Godil and A. I. Wagan. Salient local 3d features for 3d shape retrieval. In *IS&T/SPIE Electronic Imaging*, pages 78640S–78640S, 2011. 5

[9] S. Goferman, L. Zelnik-Manor, and A. Tal. Context-Aware Saliency Detection. In *CVPR*, pages 2376–2383, 2010. 1, 4

[10] K. Grill-Spector and R. Malach. The human visual cortex. *Annu. Rev. Neurosci.*, 27:649–677, 2004. 2

[11] X. Hou and L. Zhang. Dynamic visual attention: Searching for coding length increments. *Advances in neural information processing systems*, 21:681–688, 2008. 1

[12] L. Itti, C. Koch, and E. Niebur. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *PAMI*, pages 1254–1259, 1998. 1

[13] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *PAMI*, 21(5):433–449, 1999. 3

[14] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *ICCV*, pages 2106–2113, 2010. 1

[15] S. Katz, A. Tal, and R. Basri. Direct Visibility of Point Sets. *ACM Transactions on Graphics*, 26(3):24:1–11, 2007. 6

[16] G. Kim, D. Huber, and M. Hebert. Segmentation of salient regions in outdoor scenes using imagery and 3D data. In *Workshop on Applications of Computer Vision*, 2008. 1

[17] C. Lee, A. Varshney, and D. Jacobs. Mesh saliency. *ACM Transactions on Graphics*, 24(3):659–666, 2005. 1, 5

[18] G. Leifman, E. Shtrom, and A. Tal. Surface regions of interest for viewpoint selection. In *CVPR*, pages 414 –421, 2012. 5, 6, 7

[19] V. Mahadevan and N. Vasconcelos. Spatiotemporal saliency in dynamic scenes. *PAMI*, 32(1):171–177, 2010. 1

[20] J. Novatnack and K. Nishino. Scale-dependent 3d geometric features. In *ICCV*, pages 1–8. IEEE, 2007. 5

[21] I. Pratikakis, M. Spagnuolo, T. Theoharis, and R. Veltkamp. A robust 3d interest points detector based on harris operator. In *Eurographics Workshop on 3D Object Retrieval*, 2010. 5

[22] D. Rudoy, D. Goldman, E. Shechtman, and L. Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *CVPR*, 2013. 1

[23] R. Rusu and S. Cousins. 3D is here: Point cloud library (PCL). In *ICRA*, pages 1–4, 2011. 7

[24] R. B. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D Registration. In *ICRA*, 2009. 3

[25] H. J. Seo and P. Milanfar. Static and space-time visual saliency detection by self-resemblance. *Journal of vision*, 9(12):171–177, 2009. 1

[26] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. *CGF*, 28(5):1383–1392, 2009. 5

[27] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, 2010. 3

[28] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural Networks*, 19(9):1395–1407, 2006. 1

[29] Y. Yeshurun, R. Kimchi, G. Sha'shoua, and T. Carmel. Perceptual objects capture attention. *Vision research*, 49(10):1329–1335, 2009. 2