



# Viewpoint placement for inspection planning

Petra Gospodnetić<sup>1,2</sup> · Dennis Mosbach<sup>1,2</sup> · Markus Rauhut<sup>2</sup> · Hans Hagen<sup>1</sup>

Received: 4 January 2021 / Revised: 10 June 2021 / Accepted: 23 September 2021 / Published online: 30 October 2021  
© The Author(s) 2021

## Abstract

Inspection planning approaches so far have focused on automatically obtaining an optimal set of viewpoints required to cover a given object. While research has provided interesting results, the automatic inspection planning has still not been made a part of the everyday inspection system development process. This is mostly because the plans are difficult to verify and it is impossible to compare them to laboratory-developed plans. In this work, we give an overview of available generate-and-test approaches, evaluate their results for various objects and finally compare them to plans created by inspection system development experts. The comparison emphasizes both benefits and downsides of automated approaches and highlights problems which need to be tackled in the future in order to make the automated inspection planning more applicable.

**Keywords** Visual surface inspection · Inspection planning · Viewpoint placement · Image acquisition optimization

## 1 Introduction

An automated system performing visual surface inspection is an important component in industrial production lines. It is used to inspect a given object for defects which were strictly defined by the manufacturer. Inspected objects have a varying level of geometrical complexities, from simple flat surfaces (e.g., sheet metal or paper production) to more complicated freeform structures containing hard-to-inspect areas such as holes and cavities. In addition to the geometrical complexity, surface of the object changes its appearance depending on the material and surface texture. The development of such systems today is experimentally driven and requires a group of experts. It can roughly be divided into four steps: *product assessment*, *prototype configuration*, *prototype verification* and *configuration adaptation*. The steps include both

hardware (image acquisition) and software (image analysis) aspects of the system development, with both being equally important. A well- designed image acquisition ensures that all necessary parts of the object are imaged in a way which will enable the image analysis to detect defects.

The process of designing the right image acquisition setup can be lengthy and is likely to integrate subjective assumptions made by the expert developing the system. Challenges which are met during the development typically derive from the surface material and the geometrical complexity of the inspected object. The high variation in surface orientation makes not only the reflectance issues more prominent, but also introduces a problem of self-occlusion. Taking it all into consideration, the expert is expected to find an acquisition solution, capable of inspecting the object in its entirety. Therefore, once set up, the inspection system cannot be easily adapted to a new application without repeating the complete prototyping process.

Such inflexibility may be acceptable for production lines which consistently produce a small set of known objects. However, if objects on the production line vary, development of automated inspection system becomes more challenging. It causes particular problems for additive manufacturing industry, as well as factories that aim to follow the Industry 4.0 trend, orienting their production toward serialization and high product customization.

Automated planning of inspection system aims to make the overall design process more flexible and faster by pro-

✉ Petra Gospodnetić  
petra.gospodnetic@itwm.fraunhofer.de

Dennis Mosbach  
dennis.mosbach@itwm.fraunhofer.de

Markus Rauhut  
markus.rauhut@itwm.fraunhofer.de

Hans Hagen  
hagen@cs.uni-kl.de

<sup>1</sup> Computer Graphics and HCI Group, University of Kaiserslautern, 67663 Kaiserslautern, Germany

<sup>2</sup> Image Processing Department, Fraunhofer ITWM, 67663 Kaiserslautern, Germany

viding an expert with a possibility to use a set of automated methods and tools. The approaches offered so far focused on optimization of viewpoints (camera positions) relative to the object, but gave little attention to evaluation of methods used to generate viewpoints before optimization. In this work, we explore different existing generate-and-test techniques used for viewpoint generation and compare their performance in terms of achieved coverage in the resulting set of viewpoints. Additionally, we introduce a straightforward generate-and-use approach and examine its use. Finally, for the first time, we offer a possibility to compare results of automated and manual viewpoint planning for the same objects.

All the results presented in this work have been made available online<sup>1</sup>, as well as the source code for obtaining the graphs and tables presented in this chapter<sup>2</sup>.

## 2 Inspection planning research

The importance of researching inspection planning has been recognized a few decades ago with Cowan and Kovesi [1], where they examined a possibility of using an analytical approach and highlighted the complexity of the task. Since then, the topic received attention sporadically, with most of the works focusing on optimal device placement. Works which did not focus solemnly on device placement were Tarabanis et al. [2], Scott et al. [3], and Gospodnetić et al. [4]. Tarabanis et al. [2] introduced the classification into *generate-and-test* and *synthesis* approach, and made an exhaustive overview of the work done so far. The survey considers objects containing only polyhedral features and the authors state a clear need for the development of planning algorithms capable of handling complex surfaces. They are inclined toward the *synthesis approach*, stating that the viewpoint candidate approach might have the following drawbacks: computational costs of the combinatorial search over finely tessellated (spherical) parameter spaces, use of a tessellated sphere for viewpoint visibility evaluation, and overall scalability of the approach. Besides a survey, Scott et al. [3] introduced 16 requirements a view planning algorithm for range sensing should satisfy and, among others, emphasized the lack of standardized benchmarks. Those requirements have been adapted and expanded by Gospodnetić et al. [4], in order to be suitable for flexible inspection systems. Additionally, they introduced a concept of a modular inspection planning pipeline which is application-flexible and made a clear distinction between an automated and semi-automated inspection planning and posed a list of requirements flexible inspection planning systems must satisfy.

<sup>1</sup> <https://owncloud.fraunhofer.de/index.php/s/H8jV9rwGN84knzP>.

<sup>2</sup> <https://gitlab.itwm.fraunhofer.de/gospodnetic/vob>.

Works which focused on optimal device placement were adopting either a continuous or a discrete approach. The discrete approaches received a lot more attention and will be the focus of our work. However, it is worth mentioning that continuous approaches presented by Mavrinac et al. [5] and Mohammadikaji [6] are a valid and interesting alternative.

Discrete inspection planning offers a set of solutions which are *practically acceptable*, but do not guarantee to produce the global optimum. It resides on a common idea that first a set of viewpoint candidates is created, which is afterward evaluated and used for picking optimal viewpoints. We split the approaches into three categories: Space Sampling, Vertex Sampling and Patch Sampling.

### 2.1 Space sampling

Space Sampling was most prevalent in the early works, but can still be found in more recent works. It is performed independently of the object 3D model, with a set of parameters defining the sampling space. Viewpoint candidates, which are generated based on the sampling, can further be adapted using the actual 3D model, i.e., the 3D model is used *passively* for refinement and evaluation of already created viewpoint candidates.

Sakane et al. [7,8] use uniformly and adaptively tessellated spheres for solution space discretization. The size of the resulting viewpoint candidate sets is briefly commented on, stating that the desired number of viewpoint candidates should be manually chosen to balance the trade-off between planning complexity and accurate inspection representation. Similarly, Tarbox and Gottschlich [9] use a densely sampled viewing sphere, explicitly stating they have no a priori preference on viewpoint placement. The camera viewing distance  $d$  is fixed to an arbitrary number and the viewpoints are uniformly distributed over the sphere's surface. It is assumed that the visible parts of the object are always within the acceptable depth of field. The main incentive for using dense sampling is to increase the likelihood of the discovery and coverage of regions which are difficult to sense by introducing redundancy. For meshes containing 1000–2500 vertices, they produce a set of 15,000 viewpoint candidates which are then used for the evaluation of the proposed optimization algorithms. Jing [10] employs a mesh dilation by a maximum depth field within which a camera can operate. Such an approach resembles the tessellated sphere, but with higher correlation to the model geometry. The viewpoint candidates are obtained by sampling the dilated surface until a predefined number of viewpoint candidates is reached. Their orientation is calculated based on the distance  $d$  to the original object primitives, where each primitive has an attraction force of  $1/d^2$ .

## 2.2 Vertex sampling

Recent contributions tend to use the given 3D model *actively*. Instead of focusing on the space around the object, they directly sample the object mesh and determine camera positions using a predefined distance to the surface points. Such approaches are more appropriate for the computation of feature-sensitive results.

Scott [11] suggests a model-based approach using geometric information of the object. A viewpoint candidate is created for each vertex of the mesh, which is stated to be the optimal point for surface description. Due to the high resolution and geometric complexity of some discrete models, the first step is to perform a curvature-sensitive resampling. After an additional decimation step, the final viewpoint candidate set is created. A good decimation level is heuristically determined to be 32 times lower than the target model. The method achieves good results for a set covering approach (e.g., a model of approx. 36,000 vertices is resampled to 702 vertices and decimated to 72 viewpoint candidates). In his work on underwater inspection path planning, Englot [12] creates a viewpoint candidate set using random sampling of the mesh primitives, where each primitive is used as a pivot point for a viewpoint. The viewpoints are generated until each facet of the model has reached an arbitrary level of redundancy (number of different viewpoint candidates from which a facet can be observed), resulting in ca. 2500 and 1300 viewpoint candidates from approximately 131,000 and 107,000 vertex meshes, respectively. Gronle and Osten [13] agree with [11] on generating one viewpoint candidate per mesh vertex. Also, they agree with [12] that using every vertex is redundant. Instead of mesh decimation or random vertex picking, they reduce the cardinality of their viewpoint candidate sets using an octree. Each leaf contains a group of neighboring candidates. A viewpoint is selected at random and all the neighboring viewpoints in the same leaf are removed if their orientation vector closes an angle with the chosen viewpoint which is smaller than a predefined threshold.

## 2.3 Patch sampling

Like Vertex Sampling, Patch Sampling uses mesh information to subdivide the given 3D model into regions which are afterward sampled using different criteria. Given that geometrically complex features are rarely equally distributed in all areas of the object, subdividing an object into smaller regions makes sense and provides an opportunity to create an unevenly distributed viewpoint candidate set. Such an uneven set of candidates has an opportunity to contain more viewpoints in regions of higher complexity.

Sheng et al. [14,15] focus on inspecting objects consisting of surfaces with prominent differences in orientation or having low degree of waviness. They take the *flat patch grow-*

*ing* approach by first splitting the surface into sub-patches, calculating the bounding box of each sub-patch and finally placing the viewpoint candidate at a fixed distance along the normal of the largest bounding box side. Prieto et al. [16] discretize a 3D model described by NURBS into voxels, split the object into sub-surfaces representing simple patches, and, for each patch, find a viewpoint which can be used on the surface in a sweeping manner (also known as view swaths). While such an approach works well on simple objects, it struggles when applied on free form or highly complex surfaces. Instead of trying to find the optimal positions for the camera placement, Mosbach et al. [17], decided to tackle the problem of geometrical complexity by sampling the surface with various densities. Their approach uses B-splines and a set of so-called feature functionals to analytically measure geometrical features of an object and use them to guide the sampling process (e.g., higher sampling in the curved areas and lower in the flat areas). For discrete mesh representations, they offer a preprocessing step to reconstruct the object first by using a set of B-spline surfaces, while for the B-spline representation the method can be used directly. The approach is independent of the 3D model resolution and generates viewpoints based on values measuring objects geometrical properties.

## 2.4 Manual planning

Automatic inspection planning aimed to enable inspection of geometrically complex surfaces, which would otherwise be very difficult to perform by an engineer. Recently, Gospodnetic et al. [18] took a different approach providing an engineer with an interactive surface inspection planning tool capable of replicating the three main parts of the physical inspection planning process: inspection object exploration, inspection region selection on the 3D model and viewpoint selection. Using the tool, an engineer can perform inspection planning digitally and obtain a real-time feedback about inspection coverage of the newly created plan and each of the viewpoints.

## 3 Viewpoint candidate requirements

In this work, we focus on inspecting objects based on their geometrical features, while the material characteristics are yet to be further examined. For that purpose requirements which a good set of viewpoint candidates must satisfy are as follows:

- R1 Provide full coverage
- R2 Exhibit adaptive coverage redundancy
- R3 Minimize the amount of inadmissible viewpoints
- R4 Have scalable coverage redundancy

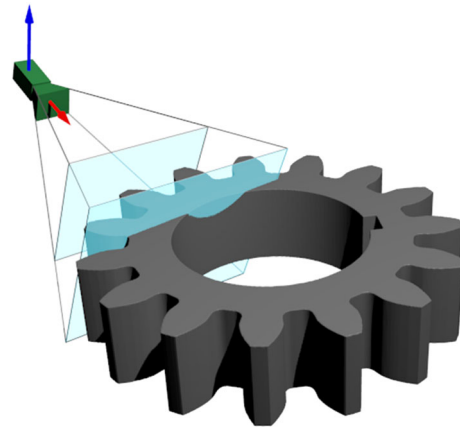
A set of viewpoint candidates must be able to completely cover the inspected object, or its restricted region, i.e., a predefined region on the object which needs to be inspected (3). Otherwise, its subset, the optimal viewpoints, will also not cover all regions which need to be inspected. Geometrical complexities typically vary across the object's surface, and viewpoint candidate placement should reflect that (3). That way more viewpoints should be concentrated in areas which are difficult to inspect (more complex). Furthermore, generate-and-test approaches rely on a visibility matrix, which is computed for every viewpoint-primitive pair and can be costly to compute (see Sect. 5). A primitive is considered visible by a viewpoint if it is observed in such a manner to satisfy planning constraints (e.g., being within a predefined depth of field or observed under an acceptable angle). A viewpoint candidate set is likely to contain viewpoints which are oriented toward the object, but do not satisfy the constraints, making those viewpoints inadmissible. However, the visibility matrix will be computed for those viewpoints as well. In order to avoid unnecessary computations, 3 states that the number of such viewpoints should be kept as low as possible. Finally, optimization based on the visibility matrix is an NP-hard set covering problem. While redundancy is needed to provide possibility to choose the optimal solution, too much redundancy will cause combinatorial explosion. In that sense, 3 emphasizes the importance that the same approach can generate a viewpoint candidate set with different viewpoint densities.

Since inspection planning will always take place prior to an actual implementation of the inspection system, it is safe to assume it can be done offline and poses no additional requirements regarding time restrictions.

## 4 Object space exploration

As can be noticed in the related work, there are various approaches tackling the viewpoint candidate generation. This work aims to reproduce the most distinct ones, compare them and revise based on requirements introduced in Sect. 3.

A viewpoint (Fig. 1) represents a camera position in space, relative to the object's origin. It is defined as a  $9D$  vector  $\mathbf{v} = [\mathbf{p}, \mathbf{d}, \mathbf{u}]$ ,  $\mathbf{p}, \mathbf{d}, \mathbf{u} \in \mathbb{R}^3$ .  $\mathbf{p}$  represents a point in space where the camera is positioned,  $\mathbf{d}$  is the direction of the optical axis of the camera (direction-vector) and  $\mathbf{u}$  represents orientation of the camera around the optical axis (up-vector). The camera is assumed to be a typical industrial area-scan camera producing a 2D image. In order to generate a set of viewpoint candidates, object space exploration must be performed. It is a process used to determine appropriate positions for viewpoint candidates. For the purpose of



**Fig. 1** Viewpoint position denotes the eye of the camera, relative to the object. Its orientation is completely defined by an up-vector (blue) and direction-vector (red). The two blue planes denote the depth-of-field, which depends on the lens settings and describes the space within which the object will be imaged with satisfactory sharpness (color figure online)

this work, we will follow the categorization introduced in Sect. 2.

### 4.1 Space sampling

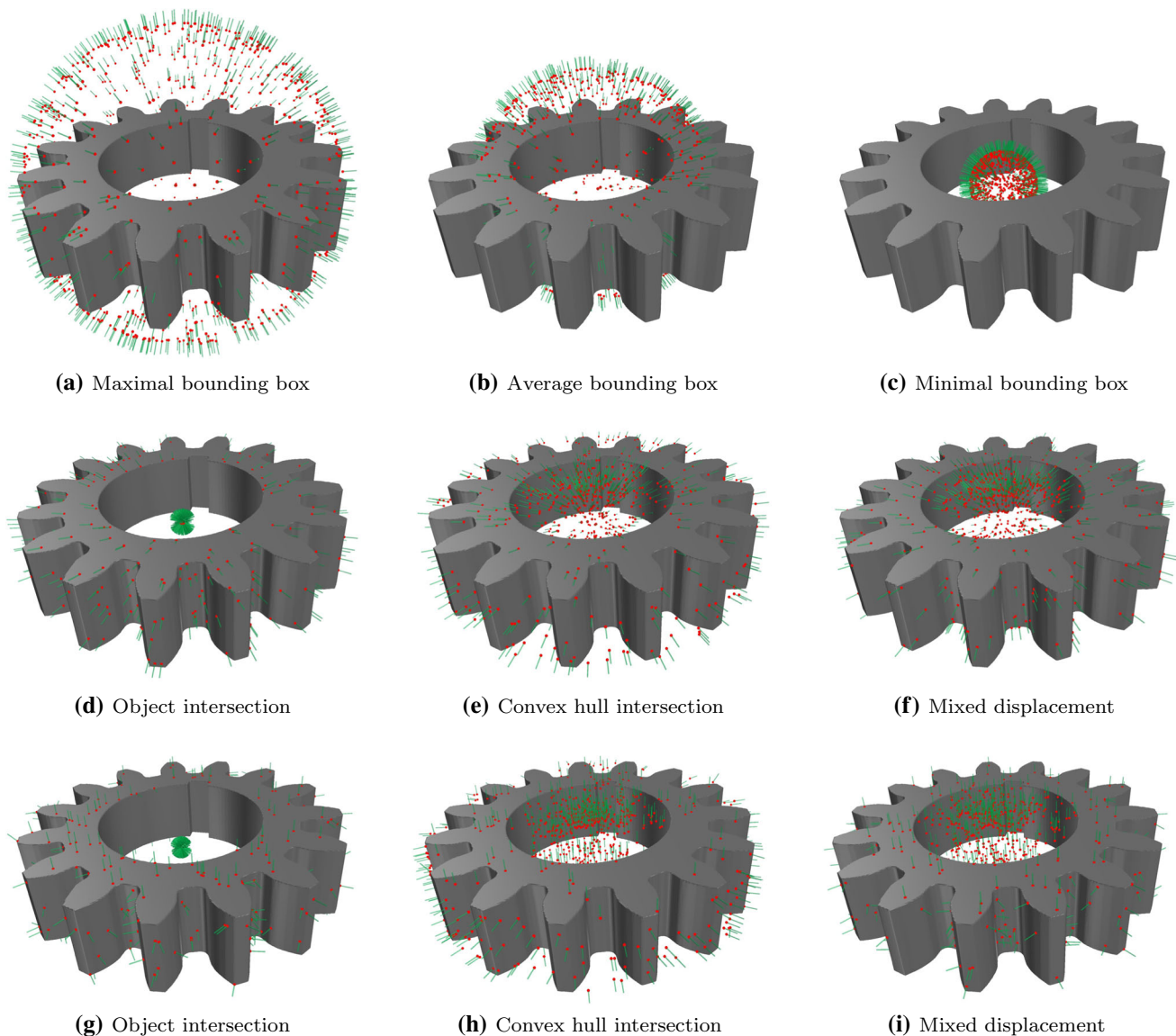
#### 4.1.1 Fixed-distance sphere sampling

The methods presented in [7–9] generate viewpoint candidates by defining a sphere around the object and sampling its surface. The origin of the sphere coincides with the center of the bounding box around the object to be inspected. The sphere is uniformly sampled to obtain a predetermined number of points. For every obtained point, a viewpoint is created by placing the camera at the point and orienting it toward the center of the object. The orientation of the up vector is not discussed. In this work, the up vector is determined using a cross-product of viewpoint direction  $\mathbf{d}$  and a predefined helper vector  $[0, 1, 0]$  (or  $[1, 0, 0]$  in case the first one is parallel with  $\mathbf{d}$ ). Because all the viewpoints are generated strictly on a sphere, they will not follow the geometry of the object and their distance to the object will vary. The sphere radius is initially set to an arbitrary focusing distance from the camera and is afterward expanded by values relating to the bounding box (e.g., half of maximal, average or minimal length of the object's bounding box), as can be seen in Fig. 2a–c.

#### 4.1.2 Constraint-aware sphere sampling

Since the objects are typically not spherical, generating viewpoints on a sphere can produce different results, with a significant number of viewpoints being completely inadmissible, i.e., not satisfying inspection constraints. Works so far did not mention possibilities to tackle this problem. For that





**Fig. 2** 1000 viewpoint candidates generated using fixed-distance ((a)–(c)) and distance-aware Sphere Sampling ((d)–(i)) methods. Viewpoints are visualized using their orientation (green line) and focal point (red). The focal point is placed along the viewpoint optical axis at focusing distance. For fixed radius, a predetermined focusing distance is used and extended with maximum bounding box extent (a), average bounding box extent (b) or minimum bounding box extent (c). (d)–(i) show

distance-aware methods where viewpoints are repositioned to the point of intersection with the surface of the object ((d), (g)), convex hull ((e), (h)), or both combined ((f), (i)). In Figures (d) – (f) viewpoints are repositioned without changing the initial orientation, while in Figures (g)–(i) they are both repositioned and reoriented to match the negative of intersection point normal

purpose, we suggest and test methods which still use the sampling sphere as a starting point but further reposition the viewpoints to satisfy inspection constraints. Repositioning the viewpoints after sampling the sphere will introduce *passive geometry awareness*, meaning that the viewpoint generation itself was not guided by the geometry, but the geometry was used to adapt the viewpoints. The repositioning is performed based on information obtained by casting a ray along the viewpoint direction vector and evaluating the

intersection. If a primitive has been intersected, the viewpoint can be adapted to match any given constraints. For the purpose of this work, we are evaluating *focusing distance* and *angle of incidence* constraints.

The focusing distance constraint is satisfied by repositioning the viewpoint along its optical axis (direction vector) so that the distance between the viewpoint and intersected object surface matches the focusing distance (Fig. 2d), i.e., the focal point of the viewpoint lies on the surface of the object. If the

optical axis of the viewpoint does not intersect the object, the viewpoint remains unchanged. Because of that, objects which have even slightly more complex topology are likely to suffer if viewpoints are repositioned only based on the intersection of the optical axis and the object. Angle of incidence restricts the maximal angle which the optical axis closes with the normal of the intersected primitive. The constraint is satisfied if the viewpoint is repositioned to be parallel with the normal.

The problem can be alleviated if the convex hull of the object is used. Instead of checking if the optical axis intersects the object itself, it is checked if it intersects the convex hull of the object. If the intersection exists, the viewpoint is repositioned accordingly (Fig. 2e). Repositioning viewpoints in that manner will create *floating viewpoints*, for which the focal point does not lie on the surface object but is close to it, increasing the likelihood of the viewpoint being admissible.

Displacing a viewpoint to the convex hull of an object may cause viewpoints, which would otherwise intersect the object, to be placed at a distance to the surface greater than the camera focusing distance. Mixed displacement method (Fig. 2f) repositions the viewpoint focal point either to the surface of an object, or to its convex hull if no intersection with the object is available.

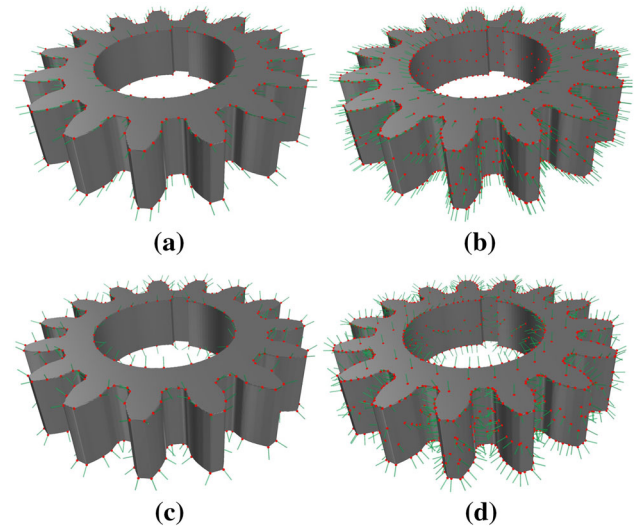
As can be noticed in Fig. 2a–f, repositioning the viewpoints without changing their orientation reveals that many of the viewpoints might be discarded as inadmissible due to their angle of incidence. For that purpose, in addition to focusing distance repositioning, the viewpoints are reoriented to satisfy  $\mathbf{d} = -\mathbf{n}$ , where  $\mathbf{n}$  represents the normal of an intersected primitive (Fig. 2g–i).

## 4.2 Vertex sampling

Using the sampling sphere might be a straightforward approach, but the geometry of the object is only taken into account by the constraint-aware methods. In order to focus more on the geometry when generating viewpoints, using mesh vertices is a natural approach, since vertices can be considered as optimal points required to describe the given geometry [11].

### 4.2.1 Direct vertex sampling

In its simplest form, a viewpoint candidate is created for every vertex of the object and then displaced from the point by the predetermined focusing distance. Direction of the displacement depends on the desired orientation of the viewpoint. Therefore, when using the vertex-based approach, it is important to determine the desired viewpoint orientation and density aspects beforehand. A viewpoint can be oriented toward the center of the object (Fig. 3a, b) or along the negative normal of the vertex used to generate it (Fig. 3c, d). Since



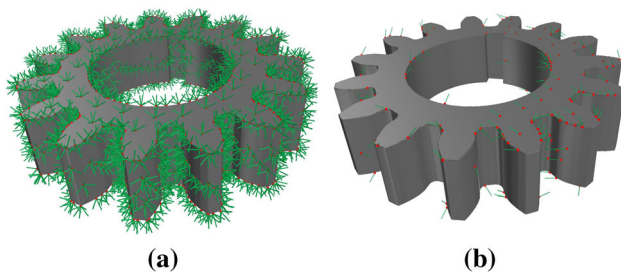
**Fig. 3** Viewpoint candidates generated using Direct Vertex Sampling using two different orientations for two different mesh resolutions. (a) and (b) show viewpoints oriented toward centroid, while (c) and (d) show viewpoints oriented using corresponding vertex normal. Figures (a) and (c) viewpoints are generated on a mesh containing 337 vertices, while the mesh used for (b) and (d) had 1350 vertices. Number of viewpoints for all figures corresponds to the number of vertices

a viewpoint is generated for every vertex, cardinality of the vertex-based methods is directly dependent on the resolution of the mesh. Therefore, Scott et al. [11] suggested to lower the number of viewpoint candidates by reducing mesh resolution. However, the spatial distribution is also dependent on the vertex placement and the resolution reduction may cause complete absence of viewpoints in flat areas, as shown in Fig. 3a, c.

### 4.2.2 Octree filtering

As proposed by Gronle and Osten [13], viewpoints are first generated for every vertex and oriented in the negative direction of the vertex normal. In order to introduce different angles of incidence into the set, additional viewpoints are generated by duplicating and rotating every viewpoint around the corresponding vertex, as shown in Fig. 4a. Maximum rotation of each viewpoint is  $90^\circ$  from the original orientation. All the viewpoints are further stored into an octree, which is subdivided until the child exceeds a predetermined threshold  $t$  and the leaves contain neighboring viewpoints. Finally, the number of viewpoints is reduced by randomly picking a viewpoint in each leaf and removing neighbors whose orientation difference is smaller than the predetermined minimal angle  $\phi$ , as shown in 4b.

For the purpose of this work, we have implemented the proposed method and introduced minor modifications. Since the viewpoint distance has not been explicitly discussed, we assume the viewpoint is displaced from the vertex along the



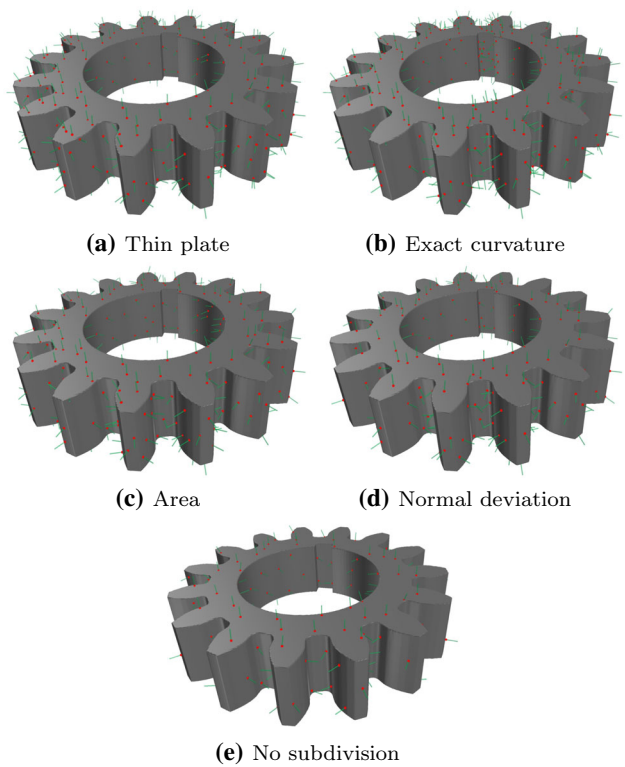
**Fig. 4** Viewpoint candidates generated using octree filtering method for a mesh containing 1350 vertices. **(a)** First, viewpoints are created for each vertex, oriented using the vertex normal and rotated in two planes by  $55^\circ$  from the original viewpoint, creating 6750 viewpoints before filtering. **(b)** After filtering, 218 viewpoints remained as viewpoint candidates

vertex normal for the value of a predetermined focusing distance. In the original work, the authors performed the rotation in a single plane which matched a plane accessible by the available 4-DOF manipulator. We assume a 6-DOF manipulator is available and therefore rotates the viewpoints in two perpendicular rotation planes. Originally, volume fraction of the inspected object is used as the octree subdivision threshold. Since the octree is built around viewpoints after they have been displaced to satisfy the focusing distance, it is possible that the adequate threshold might be greater than the object volume and determining it might be difficult. In this work, we use the volume threshold, but also explore a possibility of using a different threshold, i.e., a percentage of all generated viewpoint candidates contained within the leaf.

### 4.3 Patch sampling

Direct use of the mesh representation of an object restricts the viewpoint generation to the structure of the mesh. Furthermore, it provides little information about geometrical features of the object. As mentioned in Sect. 2.3, subdivision of the object into patches gives an opportunity to generate viewpoints based on features of each separate patch.

An approach by Mosbach et al. [17] utilizes models consisting of multiple B-spline surfaces (patches) and their geometrical features, as shown in Fig. 5. Most discrete meshes can easily be converted into such models. Let  $S : \Omega_0 \rightarrow \mathbb{R}^3$  be a B-spline surface parameterized over a rectangular parameter domain  $\Omega_0 \subset \mathbb{R}^2$ . Viewpoint candidates corresponding to a B-spline surface are computed by a non-uniform surface sampling, guided by a so-called *feature functional*  $E(S, \Omega)$ , which measures how prominent a certain property occurs within a given surface segment  $S(\Omega) = \{S(u, v) \mid (u, v) \in \Omega\}$  for  $\Omega \subseteq \Omega_0$ . The choice of the feature functional controls the way the resulting viewpoints are distributed. Feature functionals considered here are as follows.



**Fig. 5** Viewpoints generated using the patch sampling, based on geometry feature functionals. **(a)** thin plate—475 viewpoints, **(b)** exact curvature—514 viewpoints, **(c)** area—427 viewpoints, **(d)** normal deviation—322 viewpoints and **(e)** no subdivision—196 viewpoints

*Curvature.* By integrating the squared principal curvatures  $\kappa_1$  and  $\kappa_2$ , this term accurately describes surface bending. It only vanishes if and only if the surface is flat.

$$E_\kappa(S, \Omega) = \iint_{\Omega} (\kappa_1^2(u, v) + \kappa_2^2(u, v)) \|\partial_u S(u, v) \times \partial_v S(u, v)\| \, dudv.$$

*Thin-plate energy.* This term is commonly used in variational design. It approximately measures surface bending via second derivatives. It can be computed efficiently but might have small nonzero values in flat regions.

$$E_{TP}(S, \Omega) = \iint_{\Omega} \|\partial_{uu} S(u, v)\|^2 + 2 \|\partial_{uv} S(u, v)\|^2 + \|\partial_{vv} S(u, v)\|^2 \, dudv$$

*Surface area.* By simply measuring surface area of a given segment, this feature functional leads to a close to uniform surface sampling. It can be combined with a curvature-based feature functional to increase the amount of viewpoints in highly curved regions while still ensuring a certain coverage



of large, flat regions.

$$E_A(S, \Omega) = \iint_{\Omega} \|\partial_u S(u, v) \times \partial_v S(u, v)\| \, dudv.$$

*Normal deviation.* This feature functional measures the maximum deviation of any normal vector  $n(u, v)$  within the segment from the average normal vector

$$E_{ND}(S, \Omega) = \max_{(u,v) \in \Omega} \cos^{-1} (n_{\text{avg}}(S, \Omega) \cdot n(u, v))$$

with

$$n_{\text{avg}}(S, \Omega) = \frac{\iint_{\Omega} n(u, v) \|\partial_u S(u, v) \times \partial_v S(u, v)\| \, dudv}{\iint_{\Omega} \|\partial_u S(u, v) \times \partial_v S(u, v)\| \, dudv}.$$

The resulting values are geometrically intuitive, but also more affected by local deformations.

Each surface of the model gets recursively subdivided into smaller segments, until the feature functional values, evaluated on each segment, are lower than a user-defined subdivision threshold  $t$ . This threshold controls the amount of resulting viewpoint candidates. It is generally chosen as a percentage of the total or average feature functional value across the entire model. For feature functions that compute geometrically intuitive values (e.g., angles) the threshold can be also given directly as an absolute value.

## 5 Next best view optimization

The next best view is a frequently used method to solve the planning problem [10,11,13]. Since the goal of this paper is to evaluate performance of different viewpoint candidate sets using a common method, the same method is used here. The optimization consists of building the visibility matrix and using it further to determine the next viewpoint to provide the biggest coverage. The problem at hand is equivalent to a minimum set coverage problem, which is an NP-hard problem.

The visibility matrix describes the correlation between the mesh primitives and viewpoints which can observe them. It is built using ray tracing, for a set of predefined parameters describing camera settings and setup constraints. Camera settings include sensor resolution, pixel size and focal length, while inspection constraints include depth of field and maximum angle of incidence. The ray-tracing method is computationally expensive, but is crucial for inspection planning because of its capability to produce realistic image simulations.

Once the visibility matrix is built, the optimization (see Algorithm 1) starts by finding the primitive covered by the least amount of viewpoints. From the set of viewpoints that

observe this primitive, the viewpoint which observes the largest part of the object is chosen. Further, viewpoints are chosen in an iterative way by always choosing the viewpoint that observes the most uncovered primitives. The process is repeated until all the primitives are covered or there are no more viewpoints available.

### 5.1 Modifications

Next best view optimization, as described above, has two challenges: (1) ray tracing is computationally expensive and (2) optimization has a so-called long-tail effect. The long-tail effect means that, as the coverage converges toward maximum, every additional viewpoint will have strongly decreasing coverage impact, introducing only a small number of newly observed primitives. Thus a significant number of viewpoints will be added to the optimal solution which have no significant impact and a high degree of redundancy. Here we propose modifications to the next best view algorithm, which can help tackle the described challenges

Ray tracing is performed for every viewpoint in the candidate set. As can be seen from Fig. 2, it is possible that some viewpoints do not satisfy the inspection constraints along the optical axis. For example, intersection point, if it even exists, is not within the depth of field, or is observed under the wrong angle. By assuming that the viewpoint is admissible only if it satisfies the inspection constraints in the center of the image, it is possible to filter out inadmissible viewpoints before ray tracing takes place. That way the time required to build the visibility matrix is reduced.

As can be seen in Fig. 10, 80% of the coverage will typically be covered within the first 30% of viewpoints. Further, viewpoints are likely to introduce a significant amount of coverage redundancy with continuously smaller amount of newly covered surface. For that reason, a so-called *stalling threshold* is introduced. The stalling threshold is set to an arbitrary percentage of primitives which are newly observed by the second used viewpoint (the first viewpoint depends on the least observed primitive and, as such, must not necessarily introduce the most newly covered surface). That way, once the next best viewpoint introduces less newly observed primitives than the threshold, the optimization is stopped. Such practice is more likely to result in 98% or 99% coverage, rather than 100%. However, the missing coverage can be examined by the user later on using coverage visualization and fixed if necessary, as described in Sect. 7.

## 6 Generate-and-use

Generate-and-test approach relies on a set of pregenerated viewpoint candidates which are further tested and optimized. However, it is also possible to generate, test and use a view-



**Algorithm 1** Next Best View

---

```

 $P \leftarrow$  list of all mesh primitives
 $\mathcal{V} \leftarrow$  list of all viewpoint candidates
 $l \leftarrow$  primitive covered by least viewpoints
 $V_l \leftarrow$  viewpoints covering  $l$ 
 $v \leftarrow$  viewpoint  $v' \in V_l$  that covers most primitives
while  $P \neq \emptyset$  and  $\mathcal{V} \neq \emptyset$  do
   $O \leftarrow O \cup \{v\}$ 
   $P \leftarrow P \setminus$  primitives covered by  $v$ 
   $\mathcal{V} \leftarrow \mathcal{V} \setminus$  viewpoints not seeing uncovered primitives
   $v \leftarrow$  viewpoint covering most uncovered primitives
end while
return  $O$ 

```

---

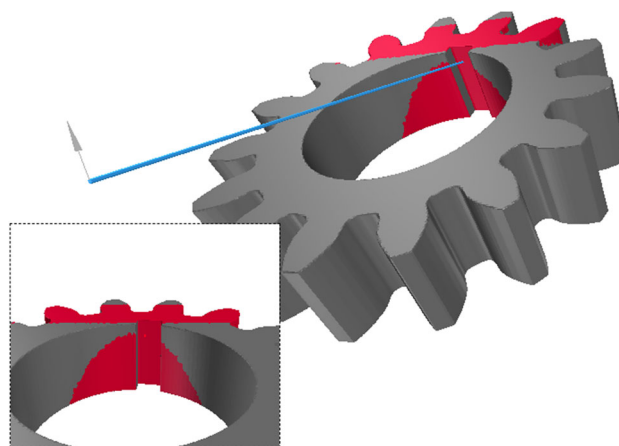
point right away, thus avoiding a separate generation step of viewpoint candidates.

In order to give an overview of the idea, this paper introduces a straightforward solution. A viewpoint is created by choosing a random uncovered primitive of the mesh and displacing the viewpoint to a predefined focusing distance in the direction of the primitive normal. After creating the viewpoint, if it satisfies all the constraints along the optical axis, it is considered admissible and ray tracing is performed to compute its coverage. Otherwise, the viewpoint is rejected and the primitive is dismissed from the set of uncovered primitives. The process is repeated until a predetermined coverage threshold  $t$  is exceeded.

## 7 Manual planning

So far no study has made a direct comparison of automatic viewpoint planning results and a plan manually created by an expert. Mostly because the physical nature of expert planning makes it impractical to precisely record each chosen viewpoint. Furthermore, the set of parameters used for viewpoint evaluation in automatic planning is only a subset of the parameters used by an expert in the laboratory, making the results even harder to compare. In this work, the viewpoint planning interface introduced by Gospodnetic et al. [18] was extended to provide the possibility of precise manual viewpoint planning.

The interface consists of an interactive 3D editor, viewpoint preview window, parameter editor and plan information. Planning parameters include camera focal length, sensor specifications (width, height and pixel size), maximal allowed incidence angle, focusing distance and depth-of-field interval within which the sharpness is acceptable. Upon determining the planning parameters, an expert can use the editor to plan the viewpoints. As shown in Fig. 6, for every created viewpoint, coverage is computed based on the chosen parameters and displayed on the surface of the object. Area which is not colored does not satisfy the constraints (e.g., acceptable sharpness or maximal angle of incidence). The



**Fig. 6** In the inspection planning interface surface of the object which is covered by the created viewpoint is highlighted in red and the image preview is shown in the lower left corner. Viewpoint is denoted by the blue line, which spans from the camera position to the focal point and is oriented to match the viewpoint direction vector. Camera up orientation is denoted by the gray arrow

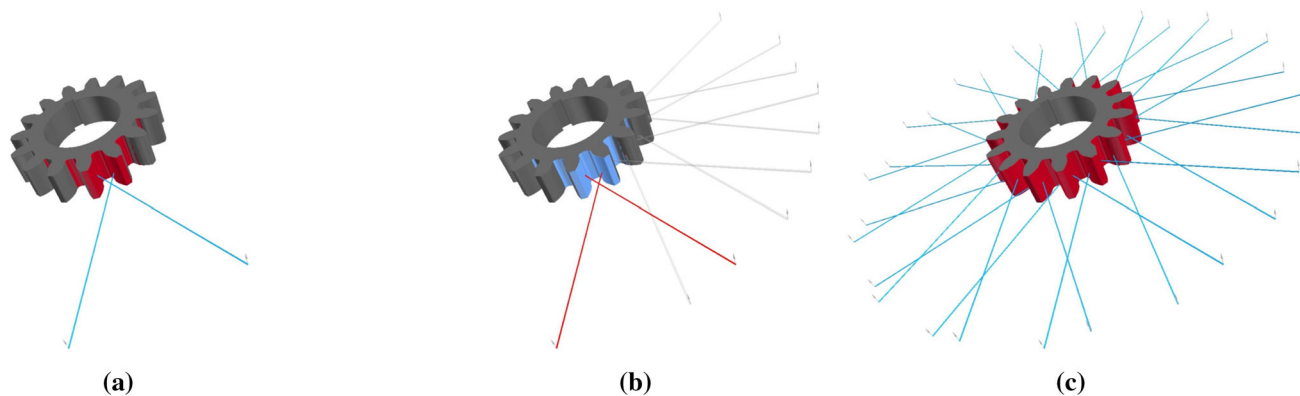
preview window displays the object geometry as it would appear in an undistorted image, together with coverage visualization.

A user can create a so-called *anchored viewpoints*, where its focal point lies on the surface of the object, or a *floating viewpoints*, where the focal point does not touch the surface of the object. Anchored viewpoints are created by selecting a point on the surface of the object which should lie in the center of the image. Floating viewpoints, on the other hand, are created by selecting a point on the convex hull of the object.

Every created viewpoint can be selected, copied, moved, deleted or rotated. Rotation can be performed in three different ways: (1) around its optical axis, thus adjusting the up vector of the camera, (2) around its focal point, thus adjusting the viewing angle, or (3) around the origin of the object, which coincides with the world coordinate system. Finally, when experts plan viewpoints, they always keep in mind a possibility to reuse the same hardware setup for different viewpoints, i.e., obtain different viewpoints by rotating the object using a turntable (rotary table). The approach is particularly useful for symmetrical or repetitive geometries. For that purpose, the interface provides a *turntable option*, where a user can select viewpoints and replicate them with a predefined rotation in world coordinate system, as can be seen in Fig. 7.

## 8 Experiments

All of the methods described in Sects. 4 through 7 have been applied to three different objects: Gear, Hirth and



**Fig. 7** Viewpoint generation using the turntable option. (a) viewpoints which can be applied on other parts of the object are created. (b) viewpoints which should be rotated and replicated are selected (their coverage is highlighted as blue). As the rotation and replication param-

eters change, the user instantly obtains a visual feedback where the new viewpoints would be placed. (c) rotation and replication are confirmed and coverage is computed for the newly created viewpoints

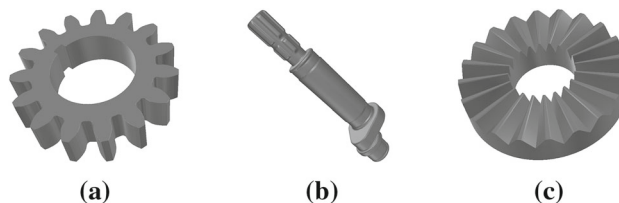
Crankshaft. The comparison pipeline can be seen in Fig. 9. To achieve maximal comparability, all tested methods used the same inspection configuration parameters. Camera parameters were defined to match a physical camera setup with a sensor size of  $2448 \times 2050$  px, focal length of 12.93, focusing distance 140 mm and depth-of-field interval ranging from 135 to 155 mm. The maximum incidence angle was set to 75 deg.

The objects resemble typical objects to be expected in surface inspection systems and are of different complexities. The Gear object (Fig. 8a) has a combination of free form (teeth and inside of a hole) and flat surfaces. The geometry of the teeth geometry has a repetitive pattern, but the areas between the teeth are frequently self-occluded. Furthermore, the surface on the inside of the hole must be observed at an angle due to self-occlusion. The Crankshaft object (Fig. 8b) has an oriented volume (elongated), it is round and can be logically split into three sections: a section which lies on the central axis, a section which is displaced from the central axis but still parallel with it, and a section which is perpendicular to the central axis. The surface is not equally smooth in all regions, adding a potential need for more viewpoints in some areas. Finally, the Hirth object (Fig. 8c) resembles the Gear in terms of the central hole, but has different surfaces. Geometry on the top surface has repetitive pattern consisting of locally flat surfaces, while the side and the bottom of the object are smooth.

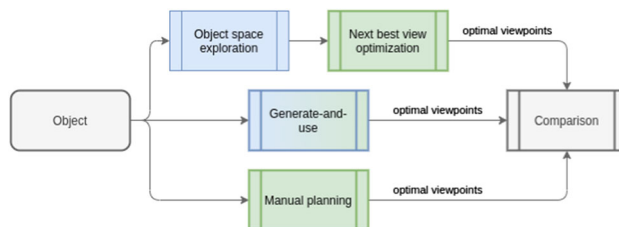
All computations have been performed on a PC with AMD FX-8320 Eight-Core Processor and 16 GB RAM.

### 8.1 Generate-and-test

The object space exploration is performed first, generating a list of viewpoint candidates which are further optimized



**Fig. 8** Testing objects (a) Gear (99,998 faces, 50,000 vertices), (b) Crankshaft (74,018 faces, 37,011 vertices) and (c) Hirth (87,360 faces, 43,680 vertices)



**Fig. 9** Comparison pipeline applied to an object. For every object 3 different types of approaches have been used. (1) Object space exploration step (blue) is performed separately from optimization (green). (2) Object space exploration is intertwined with optimal viewpoint selection. (3) Manual planning performed by an expert which is expected to output only optimal viewpoints (color figure online)

using the next best view approach. Computing the visibility matrix required the most computational time because of the underlying ray-tracing process which is performed for every viewpoint candidate. The time required to trace one viewpoint varies per image and depends mostly on the portion of the object which is visible in the image. This took 3.81 s on average. Therefore, the maximum size of the viewpoint candidate set was aimed not to exceed 1000 viewpoints where number of viewpoints could be used as a direct threshold (i.e., Sphere Sampling) and 1500 viewpoints otherwise, if

**Table 1** Performance comparison using the complete viewpoint candidate set

Approach	Method	Gear			Hirth			Crankshaft		
		#VPC	#OVP	%	#VPC	#OVP	%	#VPC	#OVP	%
Space sampling	Avg bbox	1000	31	79.6	500	19	90.15	1000	34	29.14
	Max bbox	1000	44	48.74	500	10	81.69	500	33	30.38
	Min bbox	1000	26	53.33	1000	13	86.76	1000	20	34.63
	Convex hull	1000	38	85.55	1000	17	98.69	1000	38	71.79
	Object intersection	1000	41	97.47	100	16	98.74	1000	28	65.98
	Mixed intersection	1000	41	87.84	500	16	98.08	1000	32	66.74
	Convex hull norm.	1000	29	70.02	1000	7	95.56	1000	29	86.4
	Object intersection norm.	1000	45	99.05	100	15	98.46	1000	29	90.16
	Mixed intersection norm.	1000	36	87.43	1000	16	98.86	1000	28	91.16
Vertex sampling	Direct bbox center	1350	36	82.08	1336	32	86.47	2002	34	68.14
	Direct norm.	337	35	99.0	125	18	98.39	502	24	99.67
	Octree	<b>120</b>	35	98.68	71	17	98.21	182	29	98.98
Patch sampling	Thin plate	475	35	98.85	151	13	99.65	276	<b>23</b>	99.69
	Exact curvature	514	<b>33</b>	99.03	199	<b>12</b>	99.02	327	<b>23</b>	99.55
	Area	427	34	99.09	169	15	99.79	300	24	99.38
	Normal deviation	322	35	98.21	151	13	98.67	123	24	99.02
	No subdivision	196	35	95.14	<b>70</b>	15	98.79	<b>120</b>	24	99.02

VPC viewpoint candidates, OVP - optimal viewpoints, % coverage

possible. The next best view duration was negligible in comparison, lasting under 10 seconds for a viewpoint candidate set of 1000 viewpoints. In total, we have generated and tested 292 viewpoint candidate sets. 98 for Gear, 107 for Hirth and 87 for Crankshaft. The number of generated sets per object varies because of the aforementioned aimed number of viewpoints to be contained within the set. The time required to generate viewpoints by any of the analyzed approaches was in the magnitude of seconds and can therefore be considered negligible.

Sphere Sampling methods were tested for 10, 100, 500 and 1000 generated viewpoint candidates, using full resolution models. Vertex Sampling methods depend on the resolution of the model used to generate the viewpoint candidates. For that purpose, a Quadratic Edge Collapse Strategy, implemented in MeshLab [19], was used to obtain lower resolutions of the model. The resolution was reduced as long as distinct geometrical features were preserved (e.g., edges, holes, cavities). Gear model was reduced to contain 4000, 1350, 675 and 337 vertices, Hirth model 3750, 1336, 800, 250 and 125 vertices and Crankshaft model to 3752, 2002, 1002 and 502 vertices.

The Octree Filtering approach has a set of parameters which can be adapted. For the purpose of this paper, the parameters were adjusted heuristically, aiming to reduce the number of viewpoint candidates as much as possible. In that sense, systematic testing to explore the optimal combination of parameters lies outside of the scope of the paper.

Viewpoints have been generated for different resolutions with rotation angle being 35°, 45°, 55° and 60° and filtering angle being 40°, 50°, 60° and 85°. The same angles were used for both volume and viewpoint threshold type. The thresholds values for both types were found heuristically, with viewpoint threshold being chosen as  $1/k \times 8$ , where  $k$  represents the aimed octree depth. Complete generation results can be seen in Appendix, Tables 5, 6 and 7.

B-spline Patch Sampling has been conducted using the subdivision methods described in Sect. 4.3. For area, exact curvature and thin plate, feature functional threshold values were 1, 0.75 and 0.5, while normal deviation used 45°, 60° and 90° angle thresholds. Additionally, a no-subdivision run was performed, which placed a viewpoint in the middle of every surface patch. This results in the lowest number of viewpoint candidates, see Fig. 5e.

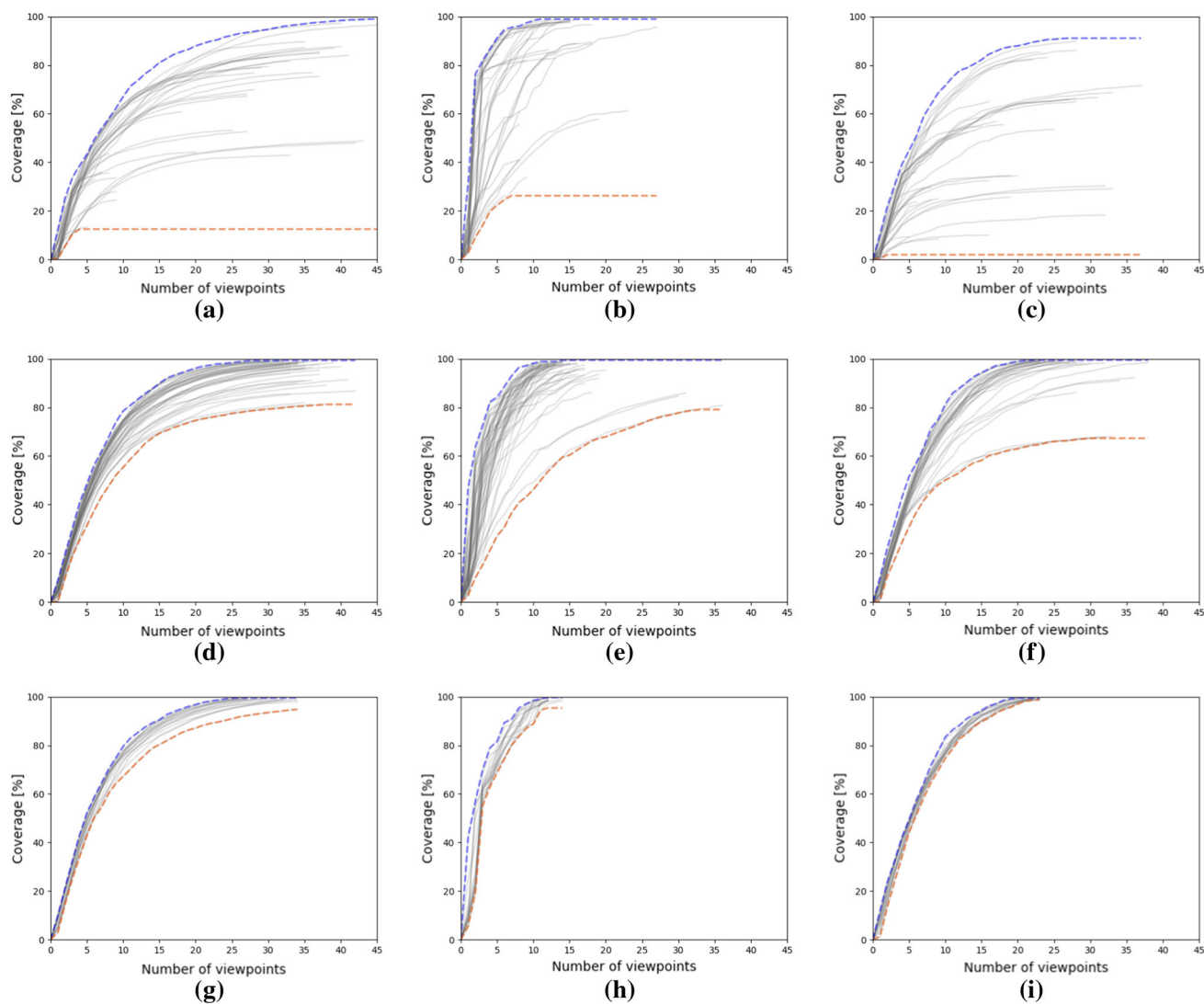
The performance of each method has been measured in terms of surface coverage using the next best view optimization with stalling threshold set to 2% of the best viewpoint coverage contribution. The methods were tested separately without (Table 1) and with (Table 2) the use of filtering described in Sect. 5.1. For each method, the result with the lowest number of viewpoint candidates to reach viewpoint coverage over 98% was considered to be the best performing. If a method could not produce coverage over 98%, the result with the highest achieved coverage was considered as best instead. Among the methods which achieved coverage of 98%, those obtaining the lowest number of viewpoint

**Table 2** Performance comparison with filtering the initial viewpoint candidate list

Approach	Method	Gear				Hirth				Crankshaft			
		#VPC	#VPC used	#OVP	%	#VPC	#VPC used	#OVP	%	#VPC	#VPC used	#OVP	%
Space sampling	Avg bbox	1000	229	28	73.02	1000	122	17	76.21	1000	223	28	26.68
	Max bbox	1000	142	35	42.69	1000	190	23	52.07	1000	9	4	3.54
	Min bbox	1000	0	0	0.0	1000	0	0	0.0	1000	691	20	34.63
	Convex hull	1000	269	38	80.92	1000	299	25	79.72	1000	773	38	70.18
	Object intersection	500	188	33	80.91	1000	350	34	88.06	1000	993	28	65.32
	Mixed intersection	1000	363	39	83.13	1000	329	33	88.75	1000	991	31	66.69
Vertex sampling	Convex hull norm.	1000	292	31	70.37	1000	408	9	96.59	1000	853	27	85.4
	Object intersection norm.	500	169	40	84.82	1000	369	16	98.5	1000	999	29	90.16
	Mixed intersection norm.	1000	329	41	88.0	1000	338	16	98.29	1000	995	28	91.16
Patch sampling	Direct bbox center	1350	945	37	81.75	1336	422	28	80.46	2002	1152	33	68.18
	Direct norm	337	317	35	98.95	125	106	18	98.32	502	493	24	99.73
	Octree	120	<b>82</b>	34	98.37	94	<b>43</b>	14	99.16	182	156	28	98.72
Patch sampling	Thin plate	475	361	34	99.11	151	130	13	99.65	276	275	23	99.69
	Exact curvature	514	381	<b>33</b>	99.03	199	142	<b>12</b>	99.02	327	326	<b>23</b>	99.55
	Area	427	293	34	99.1	169	150	15	99.75	300	299	24	99.38
	Normal deviation	322	214	35	98.22	151	114	13	98.37	123	122	24	99.02
	No subdivision	196	125	34	94.81	70	54	15	98.48	120	<b>119</b>	24	99.02

Best performing results are shown in bold  
VPC viewpoint candidates, OVP optimal viewpoints, % - coverage





**Fig. 10** Coverage convergence of object space exploration methods for each of the three models. The graphs show the number of viewpoints selected as optimal ( $x$ -axis) and the coverage they have achieved ( $y$ -axis). Rows in the Figure represent Space, Vertex and Patch Sampling approaches, respectively, while columns represent Gear, Hirth

and Crankshaft models, respectively. A single method belonging to an object space exploration approach is represented as a gray line on the corresponding graph. The dotted lines represent the envelope behavior, with blue being the best and orange being the worst performing method (color figure online)

candidates and lowest number of optimal viewpoints were emphasized (bolded). In Table 2, the number of used viewpoints after filtering was the number of evaluated viewpoint candidates.

The graphs presented in Fig. 10 show coverage convergence as optimal viewpoints are selected. They provide an overview of the overall behavior of each object space exploration approach, when applied to different models. The gray lines on the graphs represent tested methods with different parameters and show total coverage for each new viewpoint chosen by the next best view algorithm. Maximum and minimum performance per number of chosen viewpoints creates an envelope. Ideally, all the runs should be coherent and have

similar convergence, with no outliers causing the envelope to spread. Such behavior shows the consistency of an approach.

The coverage convergence graphs show how fast will the next best view algorithm converge for a different set of viewpoint candidates. Beside convergence, the number of discarded viewpoints gives additional insight into the performance of different viewpoint candidate sets by highlighting how many viewpoint candidates do not satisfy the inspection constraints in the very center of the image. For that purpose, we have computed an average of discarded viewpoints for each model. The average rejection rate per approach can be seen in Table 3. Low performance of the Space Sampling approach can be observed in Tables 1, 2 and 3. For that reason, and for the sake of brevity, Table 4 contains only

**Table 3** Average discarded viewpoints [%] per object space exploration approach against each of the models

	Gear	Hirth	Crankshaft
Space sampling	61.87		
	75.29	77.04	33.29
Vertex sampling	39.51		
	45.28	46.8	26.47
Patch sampling	15.86		
	26.59	20.7	0.3

values for methods belonging to Vertex and Patch Sampling approaches.

## 8.2 Generate-and-use

The method explained in Sect. 6 can be expected to suffer from slower convergence rate and problems in achieving full coverage. Nevertheless, we were interested to explore its behavior. For each model, the coverage threshold was set to 100% and repeated 100 times. The general coverage convergence behavior is shown in Fig. 11.

## 8.3 Manual planning

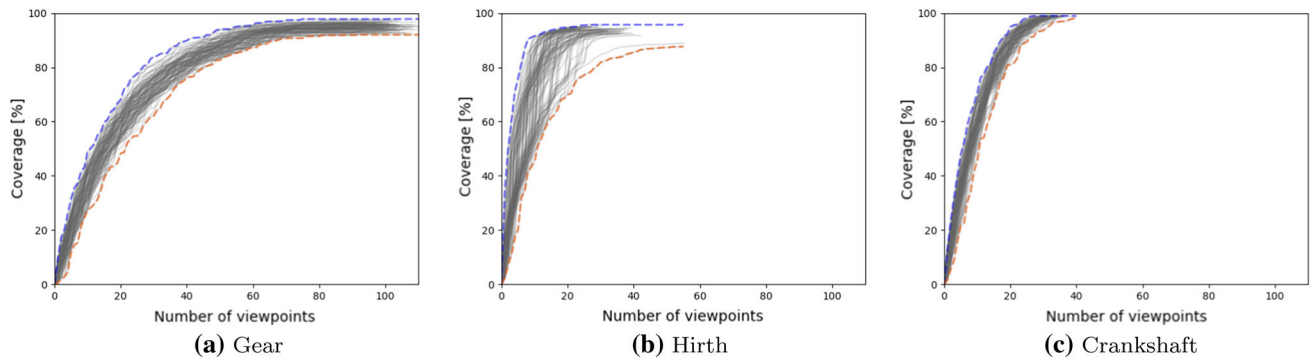
For each of the used models, two experts were given a task to use the inspection planning interface, as described in Sect. 7. Goal of each expert was to place the camera in such a way to obtain 100% coverage and the results can be seen in Fig. 12. Given that the focus of this part of the study was to gain insight into viewpoint placement comparisons, the time required for an expert to create a plan was not measured.

## 9 Discussion

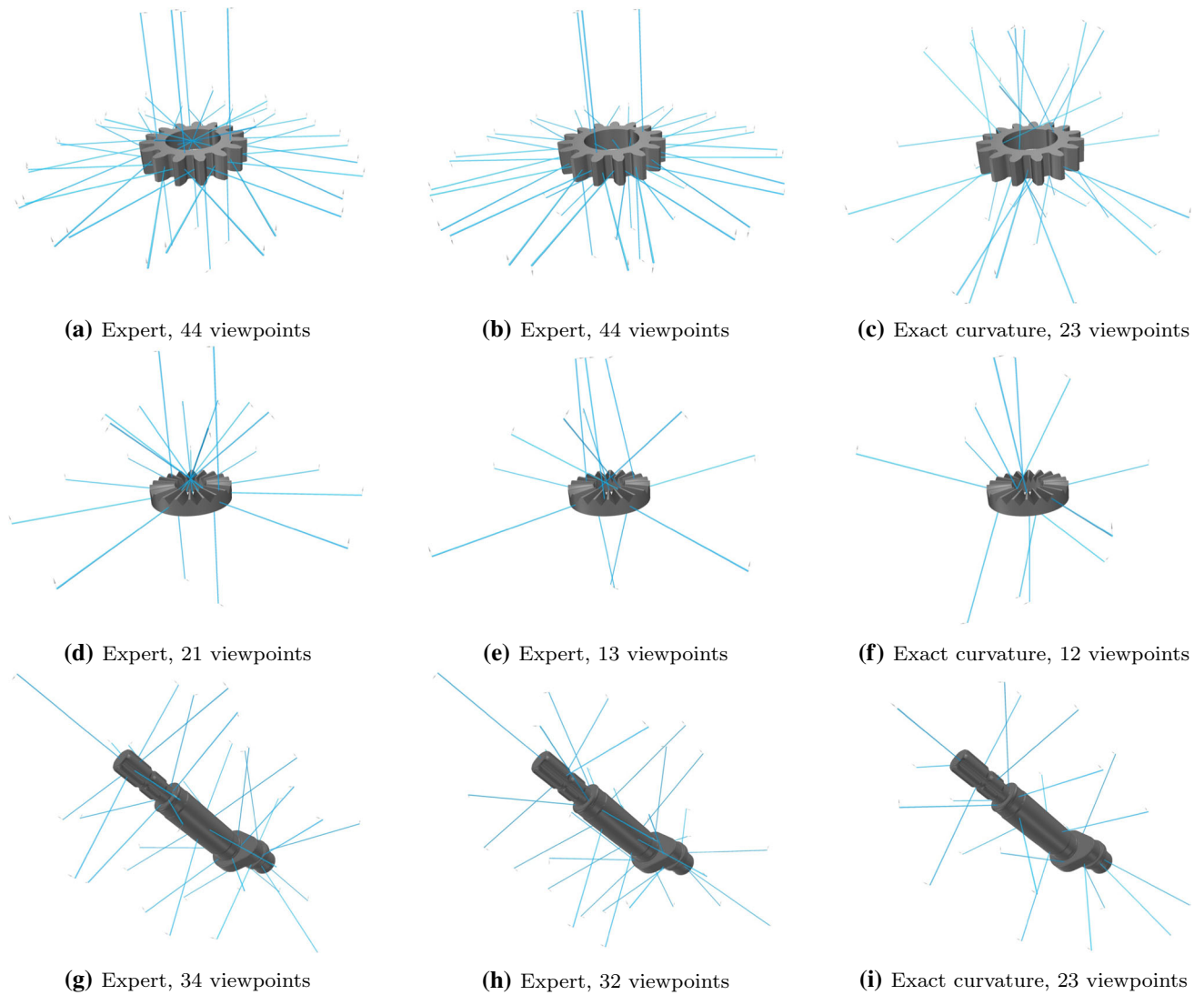
Experiments described in Sect. 8 provided insights into various viewpoint placement methods. The generate-and-test approach highlighted the importance of performing object space exploration and viewpoint generation based on the geometry of the model; rather than trying to place it in the space surrounding the object and adapting it further. The generate-and-use technique indicated that it is possible to achieve coverage results over 90% in a straightforward manner. While it may be useful for rapid prototyping, it will not provide a minimal set of viewpoints required to achieve the obtained coverage. Finally, the manual planning revealed differences in number and placement of viewpoints when compared to automated viewpoint placement. While the number of viewpoints was higher, their distribution showed characteristic patterns which indicate expert's attention to easier viewpoint traversal during inspection.

**Table 4** Average discarded viewpoints [%] per object space exploration method against each of the models

	75Direct bbox center	Direct normal	Octree	Area	Exact curvature	Thin plate	Normal deviation	No subdivision
Gear	29.02	13.9	49.25	30.97	24.02	22.65	26.24	36.22
Hirth	62.89	28.03	47.05	11.39	31.93	15.22	24.71	22.85
Crankshaft	42.6	0.66	27.63	0.08	0.19	0.17	0.63	0.83
Average	44.84	14.2	41.31	14.15	18.71	12.68	17.19	19.97



**Fig. 11** Convergence graphs showing the behavior of generate-and-use implementation for each of the models



**Fig. 12** Side-by-side comparison of manual inspection planning results made by two experts and the best performing inspection result for each of the models

## 9.1 Object space exploration

The generate-and-test results are highly sensitive to the method used to generate viewpoint candidate lists, which can be observed in Tables 1 and 2, as well as in Fig. 10.

### 9.1.1 Space sampling

In sphere sampling, viewpoints are generated in a space around the object which is not derived from the geometry of the object. The number of created viewpoints depends exclusively on the user. Reaching full coverage is difficult due to the high rate of inadmissible viewpoints (Table 3). Therefore, the user is incentivized to raise the number of generated viewpoints, e.g., see [9], which is overall counterproductive. The approach itself has shown very diverse behavior; between methods, as well as between models. The most notable difference can be observed between fixed-distance and distance-aware methods. Fixed-distance methods are highly dependent on distance parameters and are in general too rigid to be used on more complex geometries. This is mostly because of the focusing distance issues as well as the restricted angle of incidence. The focal point rarely lies on the surface of the object, making it very unlikely that the surface will be observed within the depth-of-field constraint. Furthermore, consistent viewpoint orientation toward the center of the object and varying surface orientation makes it difficult for the method to ensure that a viewpoint will observe the object under proper angle. Even if the geometry is not as complex (e.g., Crankshaft), the planning results will be affected by the volume orientation of the object.

The introduction of passive geometry awareness in distance-aware methods provides a way to increase the chance of observing the object's surface in a way that satisfies the inspection constraints. Repositioning viewpoints to the surface of the object or its convex hull guarantees that all repositioned viewpoints can observe a portion of the object within the required depth-of-field. It does not, however, guarantee that other constraints, such as maximal incidence angle are satisfied. Introducing viewpoint reorientation in addition to repositioning will provide control over the incidence angle. It would be expected that reoriented viewpoints will always perform better than just repositioned. While that might be true for topologically simple object such as Crankshaft, holes may require to be observed under a specific angle, which is different from the primitive normal and hard to generalize. Therefore, for Gear and Hirth objects, an improvement in coverage can be observed for the object intersection method (Table 1), but not necessarily for the convex hull and mixed methods. The reason lies in the fact that viewpoints with steady orientation toward the center of the object are more

likely to observe the inside of the centrally positioned hole, while reorienting viewpoints based on the normal is likely to introduce self-occlusion. Tables 1 and 2 show that distance-aware methods can achieve significantly better results than fixed-distance methods. Which method performs best depends on the geometrical features of the object.

In general, sphere sampling methods show that they are struggling in terms of achievable coverage (3) and computing a good number of admissible viewpoints (3). The distribution of the viewpoints in distance-aware methods shows that more viewpoints will observe large flat areas, while less will observe complex areas, which is contradicting 3. This approach only seems to satisfy the scalability requirement 3 by changing the number of viewpoints one wishes to generate. Even if sphere sampling was exchanged by a different shape, it is unlikely it would bring any significant improvements. Therefore, Space Sampling methods show no benefits and their use is not recommended.

### 9.1.2 Vertex sampling

In contrast to Space Sampling, Vertex Sampling directly uses the mesh representation of the object. Using vertices as representation of optimal points to describe the mesh has shown to be capable of providing coverage over 98% when using vertex normals for viewpoint orientation, even for the lowest provided mesh resolution. Orienting viewpoints toward the center of the object performed worse than the other two Vertex Sampling methods, both in terms of overall performance (Tables 1, 2) and number of inadmissible viewpoints (Table 4) in all cases. While the Direct Vertex Sampling had a significantly smaller rate of inadmissible viewpoints and achieved a slightly better coverage performance in comparison to the Octree Filtering method, its major drawback is that the object resolution plays a key role and must be adapted beforehand. Adapting the resolution of the object might cause finer geometrical features to be removed. Also it can lead to representation of flat areas using large primitives whose vertices are positioned only on edges of the areas, thus having no vertices within the flat areas. Result of having no vertices in certain areas of the object can be observed in Fig. 3a, c, where viewpoints are concentrated only around edges of the object. While reducing the number of viewpoints in flat areas is favorable, omitting them completely is not. This behavior can be avoided by increasing the resolution, and thus increasing the number of generated viewpoints.

The Octree Filtering method too depends on the initial mesh resolution for the purpose of viewpoint placement. It has the advantage that the minimal number of viewpoints is not restricted by the number of vertices, thus allowing the use of higher resolution models. However, finding a *good set*



of parameters can be a tedious process. During testing, the following behavior has been observed: higher rotation granularity will increase the likelihood of observing geometrically complex areas under different angles. At the same time, it will increase the initial number of viewpoints before filtering. Therefore, a lower volume or viewpoint threshold might be needed. Due to varying surface orientation and viewpoint focusing distance, the viewpoints originating from different sections of the object can end up in the same leaf, causing varying viewpoint density in different regions after filtering, see Fig. 4b. Like with Direct Vertex Sampling, geometrically complex regions of the object are likely to be represented with more vertices and will thus produce more viewpoint candidates. Because of random neighbor picking during filtering, the cardinality of the resulting viewpoint candidate set might vary. Finally, both volume and viewpoint threshold types showed to be able to produce viewpoint candidate sets of comparable sizes. However, the threshold using viewpoint percentage has showed to be more intuitive to use because its value will not change with the change of viewpoint focusing distance, i.e., changing the volume of the viewpoint candidate set.

In terms of requirements, both Direct Vertex Sampling (using vertex normals) and Octree Filtering method were able to cover the object completely and hence fulfill 3. Adaptive coverage redundancy is not controlled and is strictly dependent on the underlying mesh. In that sense, 3 is possible, but not guaranteed. Overall, Vertex Sampling generates under 40% of inadmissible viewpoints (Table 3), thus satisfying 3. It is also worth emphasizing that, even though it produces an overall smaller set of viewpoint candidates, the Octree Filtering method on average has a higher share of inadmissible viewpoints than the Direct Vertex Sampling, as shown in Table 4. Finally, scalable redundancy requirement 3 can only be fulfilled by the Octree Filtering method through reduction in the filtering angle parameter.

### 9.1.3 Patch sampling

Very good results were obtained via the B-spline Patch Sampling method as described in Sect. 4.3. In contrast to a discrete mesh, B-spline surfaces are described by continuous mathematical functions. This eliminates the possibility for undesired effects caused by low mesh resolution or bad vertex distribution. The discussed feature functionals make it possible to precisely measure the occurrence of features within specific regions of the object. Consequentially, the distribution of the resulting viewpoints resembles the distribution of the specified features.

While the feature functionals provide an intuitive way to control the distribution, the subdivision threshold allows users to adjust the amount of viewpoints. With these parameters, the method can be configured to produce a low number of

viewpoint candidates that sufficiently cover the entire object. As Tables 3 and 4 show, this method has a low rate of inadmissible viewpoints. To this end, it is worth pointing out that the Crankshaft object shows low number of discarded viewpoints for all methods which use inverted surface normal for determining viewpoint candidate direction. Such behavior can be expected since the Crankshaft, unlike Gear or Hirth, has no holes and cavities which would cause the viewpoint to be discarded due to self-occlusion. This emphasizes the influence of geometrical complexity in terms of topology rather than in terms of shape and symmetry.

Finally, the plots in Fig. 10 indicate that the resulting viewpoints for patch sampling are well distributed which leads to a good convergence behavior of the next best view algorithm. Together, these observations mean that all requirements (3–3) to the viewpoint candidates are fulfilled.

## 9.2 Next best view modifications

As can be seen in Table 2, upon filtering, the best performing viewpoint candidate set stayed predominantly the same. Change of the chosen set occurred most frequently in Space Sampling approach, which can be expected due to the very high number of inadmissible viewpoints. In case of the Vertex and Patch Sampling approaches, the chosen set changed only for the Octree Filtering method applied to the Hirth object. Octree Filtering is capable of producing viewpoint candidate sets which are comparably smaller but at the cost of large portion of inadmissible viewpoints. Therefore, it is possible to deduce that the use of filtering will make the next best view results more unstable for methods with high rate of inadmissible viewpoints, further highlighting the importance of requirement 3. For all other cases, both achieved coverage and the number of optimal viewpoints remained comparable between results with and without the filtering, while reducing the number of required computations.

Introduction of the stalling threshold stopped the next best view from expanding the final set of optimal viewpoints with viewpoints which introduce little coverage and a lot of redundancy. Figure 10 supports that assumption by showing that the coverage convergence in all the runs slowed down before being stopped. However, the use of the stalling threshold will also mean that complete coverage is very unlikely and the user will have to keep that in mind.

## 9.3 Generate-and-use

The introduced technique was able to achieve  $\geq 90\%$  convergence for all the models, as shown in Fig. 11. The advantage of the method is that no parameters are needed, but it is also possible to restrict it by setting the lower aimed coverage threshold. The threshold would in this case behave similar to the stalling threshold modification of the next

best view—it would cut off the convergence curve. The technique might be useful for the purpose of fast prototyping, but shows no other benefits over the generate-and-test approach.

#### 9.4 Automated versus manual planning

Manual viewpoint planning revealed a clear difference in spatial distribution of viewpoints required to perform a full coverage inspection. As can be seen in Fig. 12, while the number of manually created viewpoints was typically higher, the resulting set of viewpoints exhibits a placement pattern. The trend is most notable for the Gear, where majority of the viewpoints were created on the side of the object by replicating the rotation of two viewpoints. The benefit of such approach is that it allows a simpler physical implementation of the acquisition hardware. For example, in order to perform the side inspection, cameras could be placed on two fixed positions and a turntable can be used to rotate the object in order to inspect the side completely. In contrast to that, automatically created viewpoints exhibit no particular pattern in spatial distribution. Therefore, the inspection can only be implemented using a manipulator which entails other challenges such as path planning or timing restriction. In that case, the benefit of having a smaller set of viewpoints becomes questionable and highlights the overall inspection planning question which is yet to be answered—What makes an inspection plan good or adequate?

An automated approach, as currently formulated, will always start the evaluation of a good viewpoint placement on the surface of the object. That way, an object will surely be observed in the center of the image, however, it does not necessarily maximize the surface which the viewpoint could cover if it was placed nearby without touching the surface. A need for such viewpoints is likely to be higher for topologically complex objects. The automated viewpoint generation provides no possibility to include such reasoning, mostly because it is difficult to sample the space or the object in a manner to generate a meaningful viewpoint. On the other side, for an expert, such reasoning is natural and is more likely to be included in the manual planning process.

## 10 Conclusion

Inspection planning is a challenging topic and, like with any challenging topic, it is important to have an insight into different approaches and their qualities. The set of requirements introduced in Sect. 3 have shown to provide a good way to evaluate different object space exploration results. Classification of the object space exploration approaches into Space, Vertex and Patch Sampling made it possible to make a more meaningful comparison of their strengths and weaknesses.

It is clear that Space Sampling is unlikely to be useful due to its unstable results and high rate of inadmissible viewpoints. Vertex and Patch Sampling are both good approaches, providing viewpoint positions based on geometry characteristics, which has shown to be important. Octree Filtering and Patch Sampling using feature functionals provide competitive results, with feature functionals having an advantage of being more intuitive for parameter adjustment. While the Octree Filtering is able to create smaller viewpoint candidate sets, feature functional methods have the capacity of providing control of both the size of the viewpoint candidate set and viewpoint distribution over the object surface.

For the sake of completeness, testing models, generated data sets and dataset comparison pipeline have been made publicly available.

The methods presented in this work focus on viewpoint placement based on the geometrical features, which is a significant challenge, but it is not alone. For that purpose, we recommend further research of viewpoint placement methods base on material characteristics of the inspected object and expected illumination response.

For the first time, results of automated viewpoint placement methods have been compared to manually placed viewpoints. Such comparison made it possible to provide insight into importance of viewpoint placement in terms of spatial patterns, and not only in terms of achieving maximal coverage. As such, it has become apparent that evaluation of an inspection plan as adequate is still not possible to be made and must be explored further. Therefore, digitalization of manual inspection planning might be as important as inspection planning automation.

**Funding** Open Access funding enabled and organized by Projekt DEAL

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## Appendix

Viewpoint candidate generation results for Octree method of the vertex Sampling approach using different parameters.  $\alpha$ —viewpoint rotation angle,  $\phi$ —filter angle,  $T$ —threshold type,  $t$ —threshold value. Resolution is displayed in vertex count (Tables 5, 6, 7).

**Table 5** Viewpoint candidate generation results for Octree method on object Gear

Gear				#VPC (for 3 different mesh resolutions)		
Parameters				337	1350	4000
$\alpha$	$\phi$	$T$	$t$			
35	40	Volume	0.3	671	1369	3669
45	50	Volume	0.3	449	752	1339
60	50	Volume	0.3	401	550	1003
55	60	Volume	0.3	366	413	516
55	60	Volume	2.0	138	218	970
45	85	Volume	2.0	87	95	471
55	85	Volume	2.0	66	85	97
35	40	Viewpoint	0.125	116	4396	13524
35	40	Viewpoint	0.0156	566	1889	5668
45	50	Viewpoint	0.0156	464	812	2648
60	50	Viewpoint	0.0156	325	536	1128
55	60	Viewpoint	0.0156	276	242	457
45	85	Viewpoint	0.0156	273	165	328
45	85	Viewpoint	0.125	122	855	1041
55	85	Viewpoint	0.0156	249	118	129
55	85	Viewpoint	0.125	62	235	629

**Table 6** Viewpoint candidate generation results for Octree method on object Hirth

Hirth				#VPC (for 3 different mesh resolutions)		
Parameters				125	1002	3750
$\alpha$	$\phi$	$T$	$t$			
35	40	Volume	0.3	802	2135	2185
35	40	Volume	0.5	309	551	598
35	40	Volume	10	160	1176	927
45	50	Volume	10	94	379	682
60	50	Volume	10	71	378	186
55	60	Volume	10	59	128	105
45	85	Volume	10	63	81	65
55	85	Volume	10	58	79	72
35	40	Viewpoint	0.125	313	3576	12,535
35	40	Viewpoint	0.003125	164	982	3340
45	50	Viewpoint	0.003125	111	438	1748
60	50	Viewpoint	0.003125	93	345	632
55	60	Viewpoint	0.003125	82	108	274
45	85	Viewpoint	0.003125	97	103	102
45	85	Viewpoint	0.125	40	548	1952
55	85	Viewpoint	0.125	42	234	260
55	85	Viewpoint	0.003125	82	105	108

**Table 7** Viewpoint candidate generation results for Octree method on object Crankshaft

Crankshaft				#VPC (for 3 different mesh resolutions)		
Parameters				502	1002	3752
$\alpha$	$\phi$	$T$	$t$			
35	40	Volume	0.2	886	1437	3111
45	50	Volume	0.2	738	1132	2309
60	50	Volume	0.2	552	739	1887
55	60	Volume	0.2	494	670	1686
45	85	Volume	0.2	562	613	1208
55	85	Volume	0.2	439	545	1095
35	40	Viewpoint	0.125	1338	2658	9042
35	40	Viewpoint	0.003125	796	1737	4836
45	50	Viewpoint	0.003125	703	1143	2686
60	50	Viewpoint	0.003125	343	723	2340
55	60	Viewpoint	0.003125	251	476	1429
45	85	Viewpoint	0.003125	245	399	1044
45	85	Viewpoint	0.125	308	724	2064
55	85	Viewpoint	0.125	192	408	1426
55	85	Viewpoint	0.003125	182	276	908

## References

- Cowan, C.K., Kovesi, P.D.: Automatic sensor placement from vision task requirements. *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(3), 407–416 (1988)
- Tarabanis, K.A., Allen, P.K., Tsai, R.Y.: A survey of sensor planning in computer vision. *IEEE Trans. Robot. Autom.* **11**(1), 86–104 (1995)
- Scott, W.R., Roth, G., Rivest, J.-F.: View planning for automated three-dimensional object reconstruction and inspection. *ACM Comput. Surv.* **35**(1), 64–96 (2003)
- Gospodnetić, P., Mosbach, D., Rauhut, M., Hagen, H.: Flexible surface inspection planning pipeline. In: 2020 6th International Conference on Control, Automation and Robotics (ICCAR) (in press)
- Mavrinac, A., Chen, X., Alarcon-Herrera, J.L.: Semiautomatic model-based view planning for active triangulation 3-d inspection systems. *IEEE/ASME Trans. Mechatron.* **20**(2), 799–811 (2015)
- Mohammadikaji, M.: Simulation-based planning of machine vision inspection systems with an application to laser triangulation. Ph.D. dissertation, Karlsruhe Institute of Technology, 7 (2019)
- Sakane, S., Niepold, R., Sato, T., Shirai, Y.: Illumination setup planning for a hand-eye system based on an environmental model. *Adv. Robot.* **6**(4), 461–482 (1991)
- Sakane, S., Sato, T.: Automatic planning of light source and camera placement for an active photometric stereo system. In: Proceedings. 1991 IEEE International Conference on Robotics and Automation, pp. 1080–1087 (1991)
- Tarbox, G.H., Gottschlich, S.N.: Planning for complete sensor coverage in inspection. *Comput. Vis. Image Underst.* **65**(1), 84–111 (1995)
- Jing, W.: Coverage planning for robotic vision applications in complex 3d environment, Ph.D. dissertation, Carnegie Mellon University (2017)
- Scott, W.R.: Model-based view planning. *Mach. Vis. Appl.* **20**, 47–69 (2009)
- Englot, B.J.: Sampling-based coverage path planning for complex 3d structures, Ph.D. dissertation, Massachusetts Institute of Technology, 9 (2012)
- Gronle, M., Osten, W.: View and sensor planning for multi-sensor surface inspection. In: *Surface Topography: Metrology and Properties*, vol. 4, no. 2, (2016)
- Sheng, W., Xi, N., Song, M., Chen, Y., Rankin, J.S.: Automated cad-guided automobile part dimensional inspection. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, pp. 1157–1162 (2000)
- Sheng, W., Xi, N., Tan, J., Song, M., Chen, Y.: Viewpoint reduction in vision sensor planning for dimensional inspection. In: *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003*, vol. 1, pp. 249–254. IEEE (2003)
- Prieto, F., Lepage, R., Boulanger, P., Redarce, T.: A cad-based 3d data acquisition strategy for inspection. *Mach. Vis. Appl.* **15**, 76–91 (2003)
- Mosbach, D., Gospodnetić, P., Rauhut, M., Hammand, B., Hagen, H.: Feature driven viewpoint placement for model based surface inspection (2020)
- Gospodnetić, P., Rauhut, M., Hagen, H.: Surface Inspection Planning Using 3D Visualization. In *LEVIA 2019: Leipzig Symposium on Visualization in Applications* (2019)
- Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: MeshLab: an open-source mesh processing tool. In: Scarano, V., Chiara, R.D., Erra, U. (eds) *Eurographics Italian Chapter Conference*. The Eurographics Association (2008)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Dr. Petra Gospodnetić** has received her Master degree from Faculty of Electrical Engineering and Computing, University of Zagreb in 2016, and PhD from University of Kaiserslautern. She is leading an inspection planning research group, working on development of visual inspection systems and performance optimization. Her research focus lies on inspection planning automation through serialization of the involved processes and virtualization of the everyday planning procedures performed by an expert. In order to make it possible, she combines multiple fields such as computer vision, computer graphics, robotics and machine learning.

**Dennis Mosbach** is a researcher at the image processing department of Fraunhofer ITWM in Kaiserslautern, Germany. His main research areas are geometric modeling, computer graphics, and 3D image processing. He holds a PhD in computer science from the University of Kaiserslautern, Germany.

**Markus Rauhut** Dipl. Inform. Markus Rauhut studied computer science and mathematics at the University of Kaiserslautern. After working as a software developer in industry for a few years, he joined the Fraunhofer ITWM in 2001. Since 2014, he is head of the department Image Processing at Fraunhofer ITWM. He has many years of experience in processing and managing projects at the interface between research and application. His research interests are in the fields of algorithms for image processing and the development of complete solutions for image processing systems.

**Prof. Dr. Hans Hagen** is a computer science professor at University of Kaiserslautern and an adjunct professor at University of California, Davis. He received a Bachelor's degree in computer science, a Master degree in mathematics from the University of Freiburg and a PhD in mathematics (geometry) from the University of Dortmund. His main research interests are scientific visualization and geometric modeling. He is a member of the IEEE Visualization Academy of Science, and he got the IEEE Visualization Career Award, the ACM Solid Modeling Pioneer Award and the John Gregory Memorial Award among others.