EUROGRAPHICS 2025 / A. Bousseau and A. Dai
(Guest Editors)

*Volume 44 (2025), Number 2*

# *HPRO:* Direct Visibility of Point Clouds for Optimization

Sagi Katz and Ayellet Tal[1]

[1]Technion—Israel Institute of Technology, Israel

**Abstract**

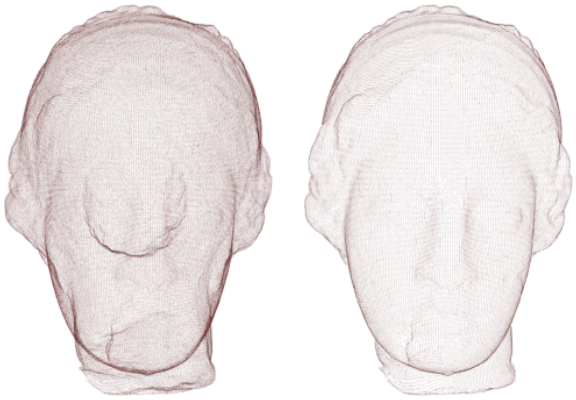
*Given a point cloud, which is assumed to be a sampling of a continuous surface, and a viewpoint, which points are visible from that viewpoint? Since points do not occlude each other, the real question is which points would be visible if the surface they were sampled from were known. While an existing approximation method addresses this problem, it is unsuitable for use in optimization processes or learning models due to its lack of differentiability. To overcome this limitation, the paper introduces a novel differentiable approximation method. It is based on identifying the extreme points of a point set in a differentiable manner. This approach can be effectively integrated into optimization algorithms or used as a layer in neural networks, allowing for the computation and utilization of visible points in various tasks, such as optimal viewpoint selection. The paper also provides theoretical proofs of the operator's correctness in the limit, further validating its effectiveness. The code is available at https://github.com/sagikatz/HPRO*

## 1. Introduction

A point cloud, which is a sampling of a continuous surface, is a set of 3D positions. It is a simple, efficient, and versatile representation of 3D data, which benefits numerous applications. This paper addresses the following task: Given a viewpoint and a point cloud, determine the subset of points that are visible from the viewpoint. In essence, the task is to identify the points that would be visible if the underlying surface, from which the points were sampled, actually existed.

The determination of visibility poses a challenge due to the lack of connectivity between points [AK04, ABCO*03, BW03, KTB07, KT15, MTSM10]. After all, due to sampling there is no inherent occlusion between points unless they coincidentally align along the same ray from the viewpoint. This scenario is illustrated in Figure 1(a), where both the front (e.g., the nose, mouth, scar) and the back (the hairdo) of Igea are visible, as the points do not occlude each other. Consequently, determining whether Igea is facing forward or backward relative to the viewpoint becomes impossible. While precise visibility can be determined on a reconstructed surface, surface reconstruction itself is a complex problem that often requires additional information such as normals and dense input. This leads to a crucial question: Can visibility be directly determined from a point cloud, bypassing the need for reconstruction?

The question of direct visibility was answered affirmatively in prior works [KT15, KTB07] with the introduction of an elegant and theoretically-supported operator called *HPR (Hidden Point Removal)*. HPR consists of two phases: point transformation and computation of the convex hull of the transformed points. It was theoretically proven that the points on the resulting convex hull correspond to the images of the visible points.



(a) Input (b) Visible points

**Figure 1:** ***Point visibility****. (a) When considering a point set sampled from a continuous surface, all points, both fore and hind, are visible. (b) Our* HPRO *operator determines the true visible points—those that would be visible on the continuous surface. Thus, the scar is visible, while the hairdo is not.*

This operator has found uses in a wide range of applications in computer graphics [BTS*17, MeSEMO14, SLT13], vision [HKD21, KKT18, RMS17], robotics [VCBL18], geographic information [SHMZ21], communication [HLQ20] and transportation [HCC*17, RS18]. Specifically, it benefits shadow estimation [KTB07], normal approximation [KT15], document enhancement [KKT18], path planning [SLT13], localization [HKD21], action recognition [RMS17], and more. However, due to its non-

differentiability, the HPR operator is not suitable for direct integration into optimization or deep learning pipelines. It is typically used in classical methods or for offline pre-processing in learning, including training networks using HPR-computed visibility as ground truth [ZH22].

We introduce *HPRO*, a novel operator that approximates the visible subset in a differentiable manner. Since the first step of the original HPR operator is already differentiable, HPRO needs only approximate the convex hull using a differential procedure. Our key idea is to leverage the basic definition of a convex hull as a sequence of *extreme points*, which are farthest in some direction. We show how to approximate their determination. This approach particularly suits point clouds with a near-spherical structure, which is our case after point transformation (the first step). This relaxes the need for an accurate non-differentiable convex hull construction algorithms like Quickhull [BDH96].

Our operator's practicality is demonstrated through its application in optimizing a viewpoint for a given point cloud. The operator's differentiability allows for visibility optimization, whether minimizing or maximizing visibility. This stands in contrast to previous view-selection algorithms that rely on pre-selected candidate viewpoints, generating views, and then comparing their quality.

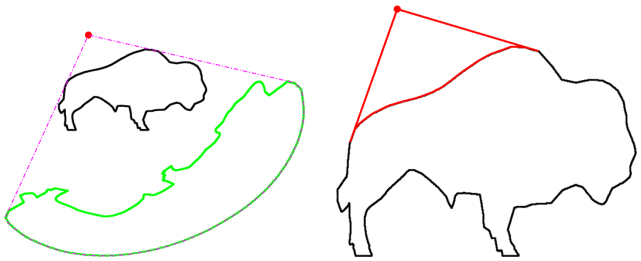
Hence the major contributions of this work are:

1. It introduces a differential, efficient, and theoretically-supported operator to determine the visible points of a point cloud.
2. It presents a simple and differentiable method for approximating the convex hull. Since convex hulls play a crucial role in various applications in computer graphics, computer vision, and robotics [MOMM08], our approximation is expected to have broad utility beyond visibility.
3. The utility of our method is demonstrated in optimizing viewpoint selection in three dimensions. Furthermore, the method can be applied to numerous other applications.

## 2. Related work

**Point cloud visibility.** Visibility determination is a fundamental problem in computer graphics [FVDF*94]. Visibility of point sets is often addressed within rendering [GBP04, RL00, WS05, WK04]. These papers typically assume that the points adhere to certain sampling criteria, such as the Nyquist condition, and are accompanied by normals. This paper builds upon previous research that aims to address the problem of point set visibility determination independently of rendering, without relying on prior assumptions or surface reconstruction [KT15, KTB07, MTSM10]. Although these approaches demonstrated excellent performance, they lack differentiability, preventing seamless integration into optimization or deep learning pipelines. The present paper focuses on resolving this limitation.

**Convex hull approximation.** The convex hull, defined as the smallest convex set encompassing a set of points in $\mathbb{R}^D$, is highly significant not only in geometry but also in various disciplines such as computer vision, computer graphics, datasets, signal processing, robotics, and communication [MOMM08]. The explosion of complexity in higher dimensions led to the development of approximations in high dimensions [KRF13, SV16]. While the objective is to avoid dimension dependence, the complexity remains considerably high, making it suitable for genuinely high dimensions. In [KKZ06] the focus is on 2D, with subsequent extension to 3D by [MeSEMO14]. We propose an efficient differentiable method that can be applied in any dimension, including $\mathbb{R}^3$ (our primary focus). Our proposed approximation is particularly suitable for point clouds that exhibit a near-spherical structure, which is the case following point transformation (Step 1 of the operator).



(a) transformation & CH (b) back projection

**Figure 2: *2D HPR operator illustration.*** *(a) Given a 2D point set (depicted as the black buffalo) and a viewpoint (represented by the red point), the points are transformed into the thin green shape (Step 1), and the convex hull (CH) is constructed (Step 2), shown as a dashed magenta line that partially overlaps with the green shape. (b) The red points, which correspond to the images of the green points lying on the convex hull (CH), are the visible points from the viewpoint. The operator can be applied in any dimension.*

## 3. Background

We are given a point set $P \subset \mathbb{R}^D$, sampled from a continuous surface, and a viewpoint $C$. The goal is to approximate the visibility of the points in $P$ from $C$. The HPR operator of [KT15, KTB07] performs this approximation in two steps, as shown in Fig. 2:

**Step 1—Point transformation:** The point set undergoes a transformation into a dual space, where: (1) Points that were initially closer to the viewpoint are moved farther away. (2) Mildly concave regions are changed into convex regions. Let $f$ be a 1-dimensional continuous kernel function $f : \mathbb{R}^+ \to \mathbb{R}^+$ that takes the distance of $p_i$ from $C$ and generates an updated distance after applying $f$, $\forall p_i \in P$. It is assumed that $f$ is invertible and has a parameter $\gamma$. We define $F_f : \mathbb{R}^D \to \mathbb{R}^D$ as a radial transformation:

$$F_f(p_i, C) = \begin{cases} C + \frac{p_i - C}{\|p_i - C\|} f_\gamma(\|p_i - C\|) & p_i \neq C \\ 0 & p_i = C \end{cases} . \quad (1)$$

There are various options for the kernel $f$, which are discussed in [KT15], along with the three properties that $f$ should satisfy. Briefly, $f(l)$ should be monotonically decreasing, the orientation of the points relative to the viewpoint should be maintained, and the ratio of the application of $f$ to two distances should be bounded between 1 and $1-\epsilon$ for any small $\epsilon$.

**Step 2—convex hull (CH):** A CH algorithm is applied to the transformed points and $C$, $\{F_f(p_i, C) | \forall p_i \in P\} \cup C$. In [KT15], it is proven that if the above properties of $f$ are satisfied, then for a

(a) Input (b) Ground Truth (c) HPR (d) HPRO (e) HPR errors (f) HPRO errors

**Figure 3:** ***Results & errors.*** *Both [KTB07] and* HPRO *may exhibit errors, particularly around the silhouettes.* HPRO *guarantees differentiability at the expense of a slightly higher error rate. In both cases, the value of* $\gamma$ *that minimized the error was used (*$\gamma = 5 \times 10^{-4}$*).*

point $p_i$ to be visible, its image $F_f(p_i, C)$ must lie on the convex hull of the set of transformed points. This is intuitively depicted in Fig. 2, where each black point undergoes transformation along the ray from the viewpoint (Step 1), and the convex hull is constructed in the transformed domain (Step 2) to identify the visible points.

## 4. Visibility determination by *HPRO*

### 4.1. The *HPRO* operator

Let $P = \{p_i \in \mathbb{R}^D\}$, $i \in [1,n]$, be a point cloud with $n$ points. As in [KTB07, KT15], our *HPRO* comprises of two steps: point transformation and convex hull computation. The main distinction lies in the differentiable method employed for the second step. We elaborate on these steps below.

**Step 1—Point Transformation.** The point cloud undergoes a radial flip around $C$ using the function $F_f$, as defined in Eq. 1. While multiple kernels $f$ are available, we employ the *Exponential Inversion Kernel*, defined below for a scalar $l$, which represents the distance from a point to the viewpoint, and a parameter $\gamma < 0$:

$$f_\gamma(l) = l^\gamma. \tag{2}$$

**Step 2—Differentiable convex set computation.** While the previous step is naturally differentiable for any $p_i \neq C$, convex hull computation does not share this property. Hence, rather than computing the precise CH, we resort to an approximation. It is worth noting that the original HPR is already an approximation. Hence, if our CH approximation is reasonably accurate, the resulting visible set will exhibit minimal differences compared to the output of HPR, as shown in Fig. 3.

Our proposed approximation is particularly suitable for point clouds that exhibit a near-spherical structure, which is the case following point transformation (Step 1). Note that in the limit ($\gamma = 0$), the transformed point set resides on a perfect sphere, and the transformed points depend solely on the angle, unaffected by the distance or viewpoint. With smaller $\gamma$ values, the visible points tend to be transformed to a spherical-like shape (Fig. 2). This holds true for any point set and any viewpoint that is not part of the set. It is a direct consequence of one of the sufficient properties of HPR kernels, as described in [KT15]. This property states that for a $\gamma$-controlled kernel $f_\gamma$, it is required that for any $dist_1, dist_2 \in \mathbb{R}_+$ ($dist_i$ represents the distance of point $p_i$ from the viewpoint), such that $dist_1 > dist_2$, and for any $0 < \epsilon < 1$, there exists a value of $\gamma = \Gamma$ such that

$$1 > \frac{f_\Gamma(dist_1)}{f_\Gamma(dist_2)} > 1 - \epsilon.$$

Thus, particularly in the case of the exponential kernel, the transformed point set can approximate a sphere as closely as desired, provided the $\gamma$ value is sufficiently close to 0.

To obtain an approximate CH, recall that the CH comprises points that are extreme in some direction. Therefore, we propose examining a condition for each point to determine if it is an extreme point. We will show that this test satisfies two conditions: (1) It is differentiable; (2) The errors are bounded, hence making the approximation useful.

For every point $p_i$, we calculate a *visibility indicator*, $V_i$. The transformed point $F_f(p_i, C)$ is considered an extreme point if it maximizes the projection on the direction $d_i = \frac{F_f(p_i,C) - C}{||F_f(p_i,C) - C||}$. This direction connects the transformed point with the viewpoint. We prove below that the visible points indeed maximize the projection in the limit and illustrate its practical validity. This computation is performed as follows, where $R_{ij}$ is the projection of point $p_j$ on direction $d_i$:

$$R_{ij} = \langle F_f(p_j, C) - C, d_i \rangle, \tag{3}$$

$$V_i = \begin{cases} \max_j(R_{ij}) - R_{ii} = 0 & \text{if } p_i \text{ is an extreme point} \\ \max_j(R_{ij}) - R_{ii} > 0 & \text{otherwise.} \end{cases} \tag{4}$$

The resulting vector of the values $V_i$ for $i \in [1,n]$ will be used in the subsequent practical computation (Eq. 5).

This algorithm is differentiable (except when the inputs to the max function are equal) and it allows gradients to flow through, as we only employ simple differentiable operations. As expected, most of the differences between HPR and *HPRO* are around the silhouettes (Fig. 3), since the condition of Eq. 4 does not hold there and the extreme points might be maxima in different directions. Intuitively, these points are "almost visible", as explained in Fig. 4.

In Section 4.2, we prove that our operator is a good approximation in the sense that, in the limit as the density of the points

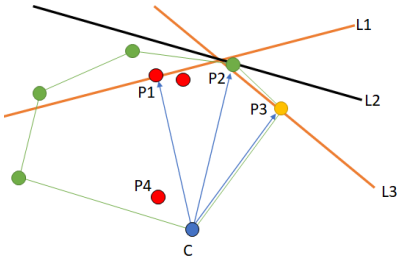



**Figure 4:** ***Convex hull approximation:*** *HPR computes the convex hull of the transformed points and C. Since all the points, except for the red points, fall on the convex hull, their images will be marked as visible. Differently, HPRO tests for each point the emptiness of the half-space defined by the point and its direction from the viewpoint. (The boundaries of the half-spaces associated with $P_{1-3}$ are $L_{1-3}$, respectively.) It will thus correctly mark as visible the green points and as invisible the red points. However, $P_3$ will be marked invisible by HPRO, since $P_2$ maximizes the projection on the vector connecting $P_3$ with C. Such cases are rare, as we will show both theoretically and experimentally.*

(a) Input (b) HPR (c) *HPRO*

**Figure 5:** ***Handling varying sampling rates.*** *Given a point cloud (a), the visible points are determined by the HPR (b) and our* HPRO *operators. Both perform well across different sampling rates, as demonstrated by their performance on the (dense) hair and the (sparse) leg.*

approaches 0: (1) For every $\gamma$, the operator does not miss visible points. (2) There exists a value of $\gamma$ for which the operator is accurate.

Our operator covers the two possible cases, when the viewpoint is located "outside" the point cloud and when it is "within" the point cloud. The case where the viewpoint is within the cloud is naturally handled, as it does not alter the input to the convex hull construction. The case in which the viewpoint is outside the cloud has to be treated especially in [KT15], by including $C$ in the convex hull computation. Nevertheless, no special handling is needed in our method, since the extreme point computation already considers $C$. For instance, in Fig. 4, if $C$ is not included in the convex hull computation, $P_4$ will be on the convex hull and will be considered visible by HPR. But, the *HPRO* condition does not consider $P_4$ as visible since there are farther points in the direction of $P_4$ from $C$.

Furthermore, our *HPRO* inherits some of the desirable properties of *HPR*: First, it is able to handle varying sampling rates. This is attributed to the empty regions associated with each point, which depend on the point's neighborhood. This is evident in Fig. 5, where the hair of the statue has a much higher density than the body. Second, it has only a handful of parameters; in particular, it has 1-4 parameters: $\gamma$ from Equation 2, which is mandatory, and $\alpha$ ,$\delta$ & $k$, which are optional and may be used in the implementation, as discussed in Section 7. Third, it is independent of any dataset for training. Fourth, HPRO works equally well with the viewpoint inside or outside, as it does not distinguish between the two. When the viewpoint is inside the object, the transformed points span all directions around the viewpoint. If the viewpoint is outside, the transformed points cover only a smaller portion of the directions. In both cases, visibility is determined by examining each point's direction relative to the viewpoint. By contrast, HPR requires including the viewpoint in the convex hull computation to ensure the convex hull does not incorrectly enclose the transformed points.

### 4.2. Correctness in the limit

The classical HPR operator [KT15, KTB07] is proved to be correct in the limit, when the density of the points is 0. If the density is not 0, the operator approximates the set of visible points. Our proposed *HPRO* adds an approximation of the convex hull. This section verifies that the theoretical guarantees in the limit still hold. Section 5 shows that this additional approximation does not compromise the accuracy too much. To prove the theoretical guarantees, we follow the definition for sample density:

**Definition 4.1** A sample $P \subseteq S$ is a *$\rho$-sample* from surface $S$ if $\forall q \in S \; \exists p \in P$ s.t. $|q-p| < \rho$.

The following two lemmas formally show that in the limit, *HPRO* provides the same guarantees as HPR. These lemmas assume that $\rho = 0$, implying that $P$ is the surface $S$ from which $P$ was sampled. Intuitively, these lemmas hold true as the transformed point set gradually resembles a sphere when $\gamma$ approaches 0, thereby improving the accuracy of the convex hull approximation.

Let $V \subseteq S$ be the set of true visible points from $C$, *HPRO(P)* be the set of points marked visible by *HPRO*, and *HPR(P)* be the set of points marked visible by *HPR*.

**Lemma 4.1** HPRO(P) $\subseteq V$, i.e. every point determined as visible by the algorithm is indeed visible.

*Proof* It suffices to prove that $HPRO(P) \subseteq HPR(P)$. Since it was proved in [KT15] that $HPR(P) \subseteq V$, the lemma will follow. Let $p_i \in HPRO(P)$, that is to say the transformed point of $p_i$ is found to be extreme in its direction by computation. As such, it must lie on the convex hull of the transformed set of points and will therefore be found visible by the HPR operator, i.e. $p_i \in HPR(P)$. □

**Lemma 4.2** There exists a value of $\Gamma$ for which $\lim_{\gamma \to \Gamma} HPRO(P) = V$.

*Proof* One side of the equality was proved in Lemma 4.1. To prove the other side, we show below that if $p_i$ is visible (i.e. $p_i \in V$), then $p_i \in \lim_{\gamma \to \Gamma} HPRO(P)$. Recall that the *HPRO* condition for $p_i$ to be considered visible, according to Eq. 4, is that the vector to

the transformed point maximizes the dot product with its direction. Recall that the length of a dot product between given vectors is the product of the vector lengths and the cosine of the angle between them. Therefore, the condition can be expressed as:

$$f(||p_i||) \geq \sup\{f(||p_j||)\cos\alpha_{ij}, p_{j\neq i} \in P\},$$

where $\alpha_{ij}$ is the angle between the vectors $p_i$ and $p_j$. For the Exponential Inversion kernel (Eq. 2) this boils down to

$$||p_i||^\gamma \geq \sup\{||p_j||^\gamma \cos\alpha_{ij}, p_{j\neq i} \in P\}.$$

Suppose that $j = k$ is the index that maximizes the right hand of this equation, then

$$||p_i||^\gamma \geq ||p_k||^\gamma \cos\alpha_{ik}.$$

If $|\alpha_{ik}| \geq \pi/2$ or $||p_i|| \geq ||p_k||$, then the equation holds for any value of $\gamma$. For $|\alpha_{ik}| < \pi/2$ and $||p_i|| < ||p_k||$:

$$\Gamma_i \log||p_i|| = \Gamma \log||p_k|| + \log(\cos\alpha_{ik}),$$

$$\Gamma_i = \log(\cos(\alpha_{ik}))/\log(||p_i||/||p_k||).$$

We can then choose $\Gamma$ that maximizes all $\Gamma_i$. □

### 4.3. Practical computation and visibility scores

In the limit, as $\gamma \to 0^-$, the transformed points converge to a sphere, and invisible points may receive $V_i$ values (Eq. 4) that are nearly 0. Due to computation inaccuracies inherent in floating-point arithmetic, checking for $V_i = 0$ becomes impractical. To address this issue, we compute the maximum in Eq. 4 by using the *top-k* function. This choice is driven by the ease of using the ordering of the $R_{ij}$ values for given $i, j \in [1,n]$ rather than determining a small constant threshold to determine that $V_i$ is close enough to 0. Thus, we define a visibility score by normalizing the $R_{ij}$ values using the $k$ top values. It's worth noting that although *top-k* is generally not differentiable, we make use of a differentiable approximation [XDC*20].

Formally, for a transformed point $F(p_i, C)$, let $R_{i,max} = \max_j(R_{ij})$ and let $R_{i,k}$ be the $k^{th}$ value. The score, $w_i$, for the visibility of $p_i$ is then defined as follows:

$$w_i = \text{elu}(\frac{R_{ii} - R_{i,k}}{R_{i,max} - R_{i,k}}), \quad (5)$$

where *elu* is the Exponential Linear Unit [CUH15]. From this equation, we observe that if $R_{ii}$ is one of the top $k$ values, then $0 \leq w_i \leq 1$; otherwise, $-1 < w_i < 0$. Note that if a point $p_i$ maximizes the projection in its direction, then $w_i = 1$. If $R_{ii}$ does not rank among the top $k$ values, its score becomes negative and it is considered invisible. In our experiments, we used $k = 10$.

**Pseudo code.** In the following, the pseudo-code of our method is given. Note the simplicity of our code: in just a few lines of code, our method can generate excellent visibility results. This is yet another benefit of our method.

**Algorithm 1** HPRO

**Input:** A point set $\{p_i, i \in [1,N]\}$, a viewpoint $C$
**Output:** A visibility vector $\{w_i, i \in [1,N]\}$

```
% Compute the transformed point set :
{p̂_i, i ∈ [1,N]} ← {F_f(p_i, C), i ∈ [1,N]}
for i = 1,2,...N do
    % Compute the projections for all points
    for j = 1,2,...N do
        R_{i,j} ← ⟨p̂_j − C, (p̂_i − C)/||p̂_i − C||⟩
    end for
    % Compute the kth value
    R_{i,k} ← topk_j(R_{i,j})
    % Compute the maximum value
    R_{i,max} ← max_j(R_{i,j})
    % Compute the score
    w_i = elu((R_{i,i} − R_{i,k})/(R_{i,max} − R_{i,k}))
end for
```

| Method | 10% noise | 5% | 2% | no noise |
|---|---|---|---|---|
| HPR (5k pts) | 77.05 | 80.22 | **84.43** | **93.17** |
| *HPRO* (5k pts) | **77.28** | **80.29** | 84.06 | 88.48 |
| HPR (1k pts) | 76.56 | **79.67** | **83.55** | **89.66** |
| *HPRO* (1k pts) | **76.87** | 79.63 | 82.76 | 85.08 |

**Table 1:** ***Accuracy on noisy models.*** HPRO *is more robust to noise compared to HPR. The results are based on an average across* 100 *models, randomly selected from ModelNet40, each resampled to 1k (5k) points.*

## 5. Experimental & ablation results

**Qualitative results.** Figs. 5, 6 demonstrate our results. In the input, points that should be invisible are visible, since points cannot occlude each other. For instance, in Fig. 6 the top or bottom of the column is visible through the column itself, as is the rear upper leg of the dinosaur through the front leg, the back of the Buddha's head through its face, and the rear leg of the chair through the seat. After applying our operator, only the genuinely visible points remain visible. The results closely resemble those produced by HPR.

**Quantitative results.** Table 1 reports mean accuracy results, computed on 100 normalized and uniformly-sampled models, randomly selected from ModelNet40 [CFG*15]. For each model, the visible points are computed from 5 random views. As expected, HPR's accuracy surpasses that of *HPRO* for clean point sets, as it is an approximation (since the convex hull is approximated). The differences, however, are minimal and tolerable. Interestingly, as noise levels increase, HPR's accuracy declines more rapidly than that of *HPRO*. At high noise levels, *HPRO*'s accuracy is slightly higher. The ground truth results are obtained by casting rays from the viewpoint to the sampled points and checking for collisions with the surface itself. Our noise model follows that of [MTSM10], where every point is perturbed within a ball with a pre-defined radius.

**Complexity and computation time.** HPR's complexity is that of the CH construction. Utilizing QHull, it is $O(n\log n)$ on average and $O(n^2)$ in the worst case. On the other hand, HPRO's complexity is $O(n^2)$ due to independent visibility computation for each point. Note that the independent computation allows easy utilization of GPU parallelism, while the standard CH algorithm used by HPR is computed on a CPU and is challenging to parallelize for the GPU.


**Figure 6:** ***Results.*** *Given a point cloud (a), the visible points are determined by the HPR (b) and our* HPRO *(c) operators. While the rear parts of the objects are visible through the front in (a), our operator correctly detects the front visible points. This is evident when looking at the top/bottom of the column, the upper leg of the dinosaur, the face of Buddha, and the rear leg of the chair. Qualitatively, the results are indistinguishable from those obtained using HPR.*

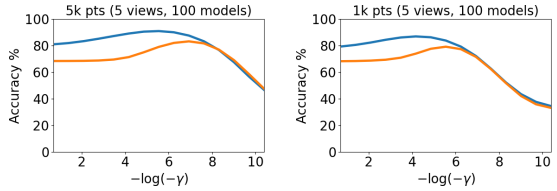
Concerning computation time: HPRO computation time is independent of the specific input and $\gamma$, whereas HPR's time depends on the structure of the input. The worst case occurs when the transformed point set is perfectly spherical and $\gamma \to 0$. Table 2 presents the runtime measurements, with HPRO running on an RTX 4090 GPU and HPR on an i9-13900K CPU. A notable advantage of HPRO is its ability to utilize the GPU, a feature not available for HPR. Importantly, the asymptotic complexity affects larger point sets. However, the advantage on smaller point clouds is significant, as most point-based networks, e.g. [CSKG17], typically work with small point clouds (1K).

| #points | HPRO | HPR (worst) | (HPR average) |
|---|---|---|---|
| 1K | 0.3 | 2.1 | 1.7 |
| 5K | 3.0 | 9.8 | 5.1 |
| 10K | 11.0 | 19.8 | 8.8 |

**Table 2:** ***Running times (in ms).*** HPRO *is faster than HPR on small data points due to its ability to run on the GPU. This is particularly significant because point-based models, such as [CSKG17], typically operate on small point clouds (1K in this case).*

**$\gamma$ selection.** Fig. 7 illustrates the accuracy as a function of $\gamma$. As expected, the optimal $\gamma$ for *HPRO* is closer to 0 (higher $-\log(-\gamma)$) compared to HPR. This difference arises because, inherently, for the same $\gamma$ value, *HPRO* creates a subset of the visible points marked by HPR, resulting in a larger false negative count. Consequently, to optimize its accuracy, one should use a $\gamma$ value closer to 0, as it better balances between false negatives and false positives. This behavior is consistent across kernels. We refer readers to [KTB07] for a discussion on the factors to consider when selecting $\gamma$ (including density, distance to points, and curvature, with density being the most critical), as well as on automatic $\gamma$ selection.

**Kernel selection.** Table 3 compares the effect of changing the kernel from Exponential Inversion with $\gamma < 0$ to *Linear*: $f(l) = \gamma - l$, with $\gamma \geq \max_{p_i \in P} ||p_i - C||$, exhibiting a slight decrease in accuracy.




(a) 5K points (b) 1K points

**Figure 7:** ***Accuracy as a function of*** $\gamma$. *The results are computed for* 5 *views per model, for* 100 *models, where HPRO is in orange and HPR is in blue.* $\gamma$ *should be set slightly closer to 0 to account for the increase in false negatives.*

| #pts | Exponential (Eq. 2) | Linear $f(l) = \gamma - l$ |
|---|---|---|
| 1k | **85.08** | 84.52 |
| 5k | **88.48** | 87.90 |

**Table 3:** ***The impact of the kernels on accuracy.*** *The exponential kernel is slightly better than the linear kernel.*

## 6. Application: Optimal viewpoint selection

Viewpoint selection is crucial for effectively visualizing 3D objects to enhance understanding and recognition. The fundamental question is what constitutes a good view. Some studies have focused on saliency [KTL*17, LST16, SLT13, SZZL22], while others have considered aesthetics [ZFY20]. In all cases, the optimal view is determined from a set of pre-selected candidate views. For a comprehensive overview, please refer to [BFS*18].

Our objective is to demonstrate that, due to the differentiability of our operator, it can be used for optimization. Specifically, by maximizing the sum of visibility scores, we can obtain a viewpoint that captures as much of the object as possible. Unlike previous approaches, which select possible viewpoints (e.g., randomly) and then evaluate their quality, our differentiable method enables optimization to determine the optimal viewpoint. As demonstrated in Fig. 8, as well as in Appendix A, *HPRO* yields slightly better results than a state-of-the-art approach. This is because the optimal view does not need to be one of the pre-selected views.

Our method is inherently simple and operates in an unsupervised manner. Given a point cloud, the method functions as follows: It first normalizes the points to fit within a unit sphere and sets an initial viewpoint at a distance of 4 from the origin. Starting from this initial viewpoint, it employs optimization (Adam), where the viewpoint serves as the learnable parameter, leveraging the differentiability of *HPRO* to determine the final viewpoint. In practice, we keep the distance from the center of the object constant and optimize only the angles. The optimization process aims to maximize the following loss:

$$Loss = -\sum_{i=1}^{n} w_i, \tag{6}$$

where $w_i$ is $V_i$ from Eq. 4 or $w_i$ from Eq. 5, which is derived from a more refined computation that takes into account practical issues. In order for the gradients to flow to non-visible points, we set $k =$ 10, so that $w_i$ represents a notion of closeness to visibility and even points that are not visible affect the gradients.

(a) [LST16] results (b) Our results

**Figure 8:** ***Viewpoint comparison.*** *Thanks to the optimization process, our viewpoints are slightly better than those of [LST16], revealing more of the rear handle and seat of the motorcycle, the rear arm of the dinosaur, and the back cloth of the statue.*

The loss in Eq. 6 can be minimized instead of maximized. By minimizing the sum of visibility scores, we anticipate obtaining a viewpoint that could be regarded as 'accidental.' Fig. 9 illustrates our results, showing both the maximization and minimization of Eq. 6. For instance, while the initial view of the airplane in Fig. 9 seems satisfactory, our maximal view reveals the double wings and displays both them and the plane's body, providing a more comprehensive perspective. Conversely, the minimal view shows very little of the airplane's surface. Similarly, while the bunny's input view is adequate, our optimal view displays more of the ear and the far front leg. The minimal view shows the bunny from underneath, exposing very few details of its features.

## 7. Additional improvements

**Handling noise** The original HPR operator did not account for noise [MTSM10]. If noise causes a point to be incorrectly detected as invisible, then moving it closer to the camera should likely make

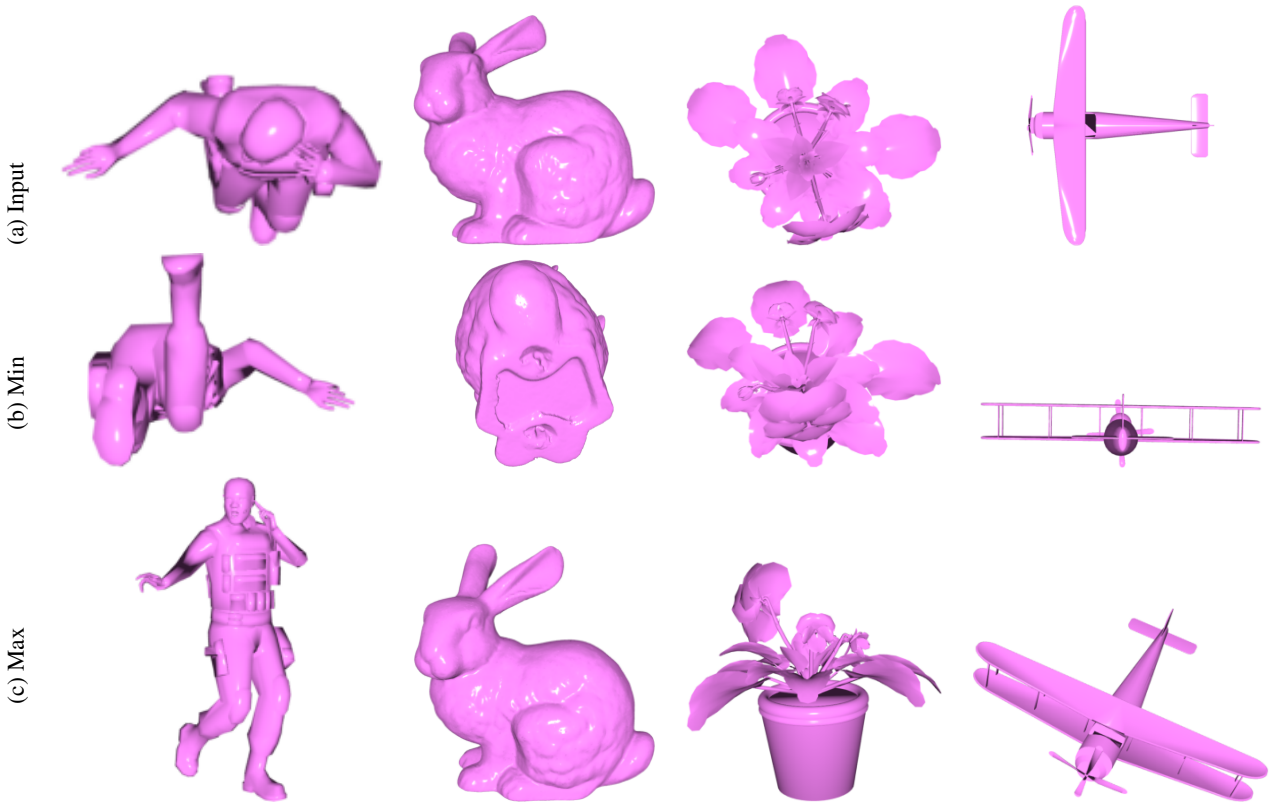



**Figure 9:** ***Viewpoint optimization.*** *Given an initial view (a), our method optimizes it, producing either a minimal (accidental) view (b) or a maximal view (c). Specifically, the minimal view of the human is from the bottom, whereas the maximal view is a $3/4$ angle, considered excellent for humans. The minimal view of the bunny is from the bottom, making it challenging to recognize the object, whereas the maximal view resembles the input but reveals more of the rear ear and the far front leg. The minimal view of the plant is from the top, not showing the pot, whereas the maximal view displays the entire object. The maximal view of the airplane reveals a double set of wings while showing much of the airplane, whereas the minimal view shows very few surfaces.*

it visible again. Therefore, we modify $R_{ii}$ in Eq. 5 as follows. Recall that $R_{ii} = ||F_f(p_i, C) - C|| = f_\gamma(||p_i - C||)$ (Eqs. 1,3). To effectively move $p_i$ closer to $C$ by a distance $\delta$, we adjust $R_{ii}$ to become $R^\delta_{ii} := f_\gamma(||p_i - C|| - \delta)$. An essential assumption here is that $||p_i - C|| >> \delta$, which is typically valid since the viewpoint should be sufficiently distant from any specific point. For the Exponential Inversion Kernel, $R^\delta_{ii} = (||p_i - C|| - \delta)^\gamma$. Thus, Eq. 5 becomes:

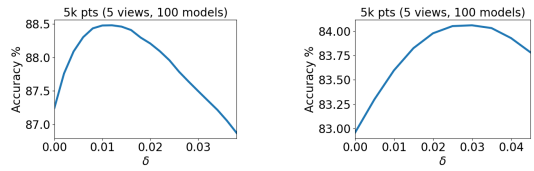
$$w^\delta_i = \text{elu}(\frac{R^\delta_{ii} - R_{i,k}}{R_{i,max} - R_{i,k}}). \tag{7}$$

Note that we only move the tested transformed point along its direction from the viewpoint, while keeping the other transformed points unchanged. Intuitively, when noise is present, slightly adjusting the tested point towards the viewpoint improves detection chances, a mechanism HPR cannot be efficiently applied since it computes the CH for all points together. Effectively handling noisy point clouds is crucial, as many real-world models contain noise.

Fig. 10 illustrates the effect of choosing different $\delta$ values on accuracy for various added noise levels. Intuitively, for a higher noise level, it is expected that the optimal $\delta$ value will be higher. Interestingly, as shown in the figure, for a given noise radius $\Delta$, the optimal value is $\delta = \Delta + 1\%$. This implies that even when no noise is added, a small value of $\delta$ is desirable. The rationale behind this is that the operator tends to generate false negatives around silhouettes, and slightly moving a tested point toward the viewpoint improves its chance of being detected.

**Improving the convex set approximation.** As shown in Fig. 3, the majority of *HPRO* approximation errors are concentrated around silhouette regions. This is expected, considering that transformed points originating from silhouette regions tend to deviate significantly from a sphere, especially when $\gamma$ is far from its limit $\Gamma$. This implies that relying on a single direction to test the extremity may not always be sufficient for silhouette points. To enhance accuracy, exploring extra directions may be beneficial. This approach is justified by the fact that a point can be considered extreme if it exhibits extremity in at least one direction.

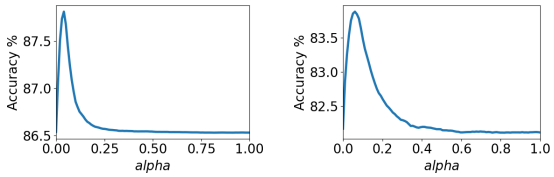
For instance, in Fig. 4 $P_3$ was not detected as visible due to $P_2$ maximizing the projection on the vector from $C$ to $P_3$. To rectify this situation, a slight rotation of this vector, or equivalently, rotation of the line $L_3$, could be applied. To achieve this, we could include a second direction to the transformed $P_3$ originating from




(a) No noise (b) 2% noise

(c) 5% noise (d) 10% noise

**Figure 10:** ***Accuracy as a function of*** $\delta$ ***for different noise levels.*** *A small added value of* $\delta$ *improves the overall accuracy.*




(a) 5K points (b) 1K points

**Figure 11:** ***Accuracy as a function of*** $\alpha$. *The more points, the smaller* $\alpha$ *should be. In the extreme case of infinitesimal density and the optimal* $\gamma$ *approaches zero, points transform into a sphere, eliminating the need for an additional direction.*

(a) Input (b) HPR [KTB07] (c) Our result

**Figure 12:** ***Limitations.*** *Both operators exhibit reduced accuracy around silhouettes and deep concavities.*

a slightly closer position to the point set. Effectively, this operation would rotate $L_3$ and potentially address the visibility issue, providing a more accurate assessment of extremity for $P_3$.

To incorporate an additional direction, we propose the following procedure: Let $M = \frac{1}{n}\sum_i F_f(p_i, C)$ be the center of mass of the transformed point set and let $C^\star = \alpha M + (1-\alpha)C$ be a point on the line between $C$ and $M$, where $\alpha \in [0,1]$ is a parameter. For each point $p_i$, compute $d_i^\star = \frac{p_i - C^\star}{||p_i - C^\star||}$, which provides a second direction to examine. Similar to Eq. 3, we calculate $R_{ij}^\star$ as follows:

$$R_{ij}^\star = \langle F_f(p_j, C) - C^\star, d_i^\star \rangle \tag{8}$$

and $V_i^\star = \max_j(R_{ij}^\star) - R_{ii}^\star = 0$, for extreme points and larger otherwise. Note that while $C$ is changed to $C^\star$ in this equation, the transformed points are still computed using $C$. Next, we proceed to compute $w_i^\star$ as in Eq. 5. Finally, a point is determined as an extreme point if it is maximal in either of the two directions. Consequently, the final visibility score for the point is given by $\max(w_i, w_i^\star)$.

Fig. 11 shows the impact of different $\alpha$ values, which control the second direction, on accuracy. For denser point sets, more aggressive $\gamma$ values are needed (closer to 0), causing the transformed points to be closer to lying on a sphere. Consequently, the source point of the second direction should be computed using a lower $\alpha$ value and be closer to the viewpoint. In the limit, the source point of the second direction and the viewpoint coincide, making $\alpha = 0$ the optimal choice.

The introduction of a second direction led to an overall accuracy improvement of around 1%. This is also evident in Fig. 11, as using one direction is equivalent to setting $\alpha = 0$.

**Limitations.** The main limitation of this operator is its nature of approximating the HPR operator, which itself provides an approximation of visibility. Consequently, it inherits the same limitations: reduced accuracy around silhouettes and deep concavities for a given $\gamma$ value, as shown in Fig. 3 and Fig. 12. It is worth noting that accuracy tends to improve with denser point sets.

## 8. Conclusion

This paper explores the determination of a visible subset of a point cloud from a given viewpoint. It introduces a novel operator called *HPRO*, which, unlike previous operators, is differentiable. The differentiability enables the use of the operator within optimization processes and learning models. The operator is based on approximating the extreme points of a set in a differentiable manner, thereby eliminating the necessity to compute the exact convex hull. Theoretical proofs establish the operator correctness in the limit.

An additional contribution of this work is demonstrating the use of our visibility operator to compute an optimal 3D viewpoint. This optimal viewpoint can maximize visibility, making it suitable for presenting 3D data, or minimize visibility, making it advisable to avoid such viewpoints.

The primary focus of the paper is on the HPRO operator itself. The viewpoint selection application demonstrates the benefits of HPRO. We hope that, similar to HPR, which found numerous uses after its publication, this will also be the case here.

**Acknowledgments.** We gratefully acknowledge the support of the Israel Science Foundation 2329/22

## References

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE Transactions on visualization and computer graphics 9*, 1 (2003), 3–15. 1

[AK04] AMENTA N., KIL Y. J.: Defining point-set surfaces. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 264–270. 1

[BDH96] BARBER C. B., DOBKIN D. P., HUHDANPAA H.: The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS) 22*, 4 (1996), 469–483. 2

[BFS*18] BONAVENTURA X., FEIXAS M., SBERT M., CHUANG L., WALLRAVEN C.: A survey of viewpoint selection methods for polygonal models. *Entropy 20*, 5 (2018), 370. 7

[BTS*17] BERGER M., TAGLIASACCHI A., SEVERSKY L. M., ALLIEZ P., GUENNEBAUD G., LEVINE J. A., SHARF A., SILVA C. T.: A survey of surface reconstruction from point clouds. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 301–329. 1

[BW03] BITTNER J., WONKA P.: Visibility in computer graphics. *Environment and Planning B: Planning and Design 30*, 5 (2003), 729–755. 1

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., ET AL.: Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015). 5

[CSKG17] CHARLES R. Q., SU H., KAICHUN M., GUIBAS L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 77–85. doi:10.1109/CVPR.2017.16. 6

[CUH15] CLEVERT D.-A., UNTERTHINER T., HOCHREITER S.: Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289* (2015). 5

[FVDF*94] FOLEY J. D., VAN DAM A., FEINER S. K., HUGHES J. F., PHILLIPS R. L.: *Introduction to computer graphics*, vol. 55. Addison-Wesley Reading, 1994. 2

[GBP04] GUENNEBAUD G., BARTHE L., PAULIN M.: Deferred splatting. In *Computer Graphics Forum* (2004), vol. 23, Wiley Online Library, pp. 653–660. 2

[HCC*17] HUANG P., CHENG M., CHEN Y., LUO H., WANG C., LI J.: Traffic sign occlusion detection using mobile laser scanning point clouds. *IEEE Transactions on Intelligent Transportation Systems 18*, 9 (2017), 2364–2376. 1

[HKD21] HYEON J., KIM J., DOH N.: Pose correction for highly accurate visual localization in large-scale indoor spaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 15974–15983. 1

[HLQ20] HAN B., LIU Y., QIAN F.: Vivo: Visibility-aware mobile volumetric video streaming. In *Proceedings of the 26th annual international conference on mobile computing and networking* (2020), pp. 1–13. 1

[KKT18] KLIGLER N., KATZ S., TAL A.: Document enhancement using visibility detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2374–2382. 1

[KKZ06] KAVAN L., KOLINGEROVA I., ZARA J.: Fast approximation of convex hull. *ACST 6* (2006), 101–104. 2

[KRF13] KHOSRAVANI H. R., RUANO A. E., FERREIRA P. M.: A simple algorithm for convex hull determination in high dimensions. In *2013 IEEE 8th international symposium on intelligent signal processing* (2013), IEEE, pp. 109–114. 2

[KT15] KATZ S., TAL A.: On the visibility of point clouds. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1350–1358. 1, 2, 3, 4

[KTB07] KATZ S., TAL A., BASRI R.: Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers*. 2007, pp. 24–es. 1, 2, 3, 4, 6, 9

[KTL*17] KIM S.-H., TAI Y.-W., LEE J.-Y., PARK J., KWEON I. S.: Category-specific salient view selection via deep convolutional neural networks. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 313–328. 7

[LST16] LEIFMAN G., SHTROM E., TAL A.: Surface regions of interest for viewpoint selection. *IEEE transactions on pattern analysis and machine intelligence 38*, 12 (2016), 2544–2556. 7

[MeSEMO14] MACHADO E SILVA R., ESPERANÇA C., MARROQUIM R., OLIVEIRA A. A.: Image space rendering of point clouds using the hpr operator. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 178–189. 1, 2

[MOMM08] MARK D. B., OTFRIED C., MARC V. K., MARK O.: *Computational geometry algorithms and applications*. Spinger, 2008. 2

[MTSM10] MEHRA R., TRIPATHI P., SHEFFER A., MITRA N. J.: Visibility of noisy point cloud data. *Computers & Graphics 34*, 3 (2010), 219–230. 1, 2, 5, 7

[RL00] RUSINKIEWICZ S., LEVOY M.: Qsplat: A multiresolution point rendering system for large meshes. In *SIGGRAPH* (July 2000), pp. 343–352. 2

[RMS17] RAHMANI H., MIAN A., SHAH M.: Learning a deep model for human action recognition from novel viewpoints. *IEEE transactions on pattern analysis and machine intelligence 40*, 3 (2017), 667–681. 1

[RS18] ROZSA Z., SZIRANYI T.: Obstacle prediction for automated guided vehicles based on point clouds measured by a tilted lidar sensor. *IEEE Transactions on Intelligent Transportation Systems 19*, 8 (2018), 2708–2720. 1

[SHMZ21] SUN Y., HUA Y., MOU L., ZHU X. X.: Cg-net: Conditional gis-aware network for individual building segmentation in vhr sar images. *IEEE Transactions on Geoscience and Remote Sensing 60* (2021), 1–15. 1

[SLT13] SHTROM E., LEIFMAN G., TAL A.: Saliency detection in large point sets. In *Proceedings of the IEEE international conference on computer vision* (2013), pp. 3591–3598. 1, 7, 10, 11

[SV16] SARTIPIZADEH H., VINCENT T. L.: Computing the approximate convex hull in high dimensions. *arXiv preprint arXiv:1603.04422* (2016). 2

[SZZL22] SONG R., ZHANG W., ZHAO Y., LIU Y.: Unsupervised multi-view cnn for salient view selection and 3d interest point detection. *International Journal of Computer Vision 130*, 5 (2022), 1210–1227. 7

[VCBL18] VECHERSKY P., COX M., BORGES P., LOWE T.: Colourising point clouds using independent cameras. *IEEE Robotics and Automation Letters 3*, 4 (2018), 3575–3582. 1

[WK04] WU J., KOBBELT L.: Optimized sub-sampling of point sets for surface splatting. *Computer Graphics Forum 23* (2004), 643–652. 2

[WS05] WALD I., SEIDEL H.-P.: Interactive ray tracing of point-based models. In *Eurographics Symposium on Point-Based Graphics* (2005), pp. 1–8. 2

[XDC*20] XIE Y., DAI H., CHEN M., DAI B., ZHAO T., ZHA H., WEI W., PFISTER T.: Differentiable top-k with optimal transport. In *Advances in Neural Information Processing Systems* (2020), Larochelle H., Ranzato M., Hadsell R., Balcan M., Lin H., (Eds.), vol. 33, Curran Associates, Inc., pp. 20520–20531. 5

[ZFY20] ZHANG Y., FEI G., YANG G.: 3d viewpoint estimation based on aesthetics. *IEEE Access 8* (2020), 108602–108621. 7

[ZH22] ZHANG Q., HOU J.: Self-supervised pre-training for 3d point clouds via view-specific point-to-image translation, 2022. doi:10.48550/ARXIV.2212.14197. 2

## **Appendix A:** Additional viewpoints comparisons

Figure 13 presents additional examples and compares our results with those of [SLT13]. HPRO often produces slightly better results than [SLT13], as demonstrated by the increased visibility of the gargoyle's rear wing and the camel's rear legs. This improvement is attributed to the fact that the optimal view is not constrained to the pre-selected views. In cases where the pre-selected views are already effective, the results are comparable, as illustrated by the examples of the star and the rocker.

(a) Initial viewpoint (b) [SLT13]'s results (c) Our results

**Figure 13:** ***Additional viewpoint comparisons.***