General Techniques for Approximate Incidences and Their Application to the Camera Posing Problem

Dror Aiger
Google, Tel Aviv, Israel
https://ai.google/research/people/DrorAiger
aigerd@google.com

Haim Kaplan
School of Computer Science, Tel Aviv University, Tel Aviv, Israel
Google, Tel Aviv, Israel
haimk@tau.ac.il

Efi Kokiopoulou
Google, Zurich, Switzerland
efi@google.com

Micha Sharir
School of Computer Science, Tel Aviv University, Tel Aviv, Israel
michas@tau.ac.il

Bernhard Zeisl
Google, Zurich, Switzerland
bzeisl@google.com

────── **Abstract** ──────

We consider the classical camera pose estimation problem that arises in many computer vision applications, in which we are given n 2D-3D correspondences between points in the scene and points in the camera image (some of which are incorrect associations), and where we aim to determine the camera pose (the position and orientation of the camera in the scene) from this data. We demonstrate that this posing problem can be reduced to the problem of computing ε-approximate incidences between two-dimensional surfaces (derived from the input correspondences) and points (on a grid) in a four-dimensional pose space. Similar reductions can be applied to other camera pose problems, as well as to similar problems in related application areas.

We describe and analyze three techniques for solving the resulting ε-approximate incidences problem in the context of our camera posing application. The first is a straightforward assignment of surfaces to the cells of a grid (of side-length ε) that they intersect. The second is a variant of a primal-dual technique, recently introduced by a subset of the authors [3] for different (and simpler) applications. The third is a non-trivial generalization of a data structure Fonseca and Mount [4], originally designed for the case of hyperplanes. We present and analyze this technique in full generality, and then apply it to the camera posing problem at hand.

We compare our methods experimentally on real and synthetic data. Our experiments show that for the typical values of n and ε, the primal-dual method is the fastest, also in practice.

1 Introduction

Camera pose estimation is a fundamental problem in computer vision, which aims at determining the pose and orientation of a camera solely from an image. This localization problem appears in many interesting real-world applications, such as for the navigation of self-driving cars [6], in incremental environment mapping such as Structure-from-Motion (SfM) [1, 10, 11], or for augmented reality [8, 9, 12], where a significant component are algorithms that aim to estimate an accurate camera pose in the world from image data.

Given a three-dimensional point-cloud model of a scene, the classical, but also state-of-the-art approach to absolute camera pose estimation consists of a two-step procedure. First, one matches a large number of features in the two-dimensional camera image with corresponding features in the three-dimensional scene. Then one uses these putative correspondences to determine the pose and orientation of the camera. Typically, the matches obtained in the first step contain many incorrect associations, forcing the second step to use filtering techniques to reject incorrect matches. Subsequently, the absolute 6 degrees-of-freedom (DoF) camera pose is estimated, for example, with a perspective n-point pose solver [7] within a RANSAC scheme [5].

In this work we concentrate on the second step of the camera pose problem. That is, we consider the task of estimating the camera pose and orientation from a (potentially large) set of n already calculated image-to-scene correspondences.

Further, we assume that we are given a common direction between the world and camera frames. For example, inertial sensors, available on any smart-phone nowadays, allow to estimate the vertical gravity direction in the three-dimensional camera coordinate system. This alignment of the vertical direction fixes two degrees of freedom for the rotation between the frames and we are left to estimate four degrees of freedom out of the general six. To obtain four equations (in the four remaining degrees of freedom), this setup requires two pairs of image-to-scene correspondences[1] for a minimal solver. Hence a corresponding naive RANSAC-based scheme requires $O(n^2)$ filtering steps, where in each iterations a pose hypothesis based on a different pair of correspondences is computed and verified against all other correspondences.

Recently, Zeisl et al. [13] proposed a Hough-voting inspired outlier filtering and camera posing approach, which computes the camera pose up to an accuracy of $\varepsilon > 0$ from a set of 2D-3D correspondences, in $O(n/\varepsilon^2)$ time, under the same alignment assumptions of the vertical direction. In this paper we propose new algorithms that work considerably faster in practice, but under milder assumptions. Our method is based on a reduction of the problem to a problem of counting ε-*approximate incidences* between points and surfaces, where a point p is ε-approximately incident (or just ε-incident) to a surface σ if the (suitably defined) distance between p and σ is at most ε. This notion has recently been introduced by a subset of the authors in [3], and applied in a variety of instances, involving somewhat simpler scenarios than the one considered here. Our approach enables us to compute a camera pose when the number of correspondences n is large, and many of which are expected to be outliers. In contrast, a direct application of RANSAC-based methods on such inputs is very slow, since the fraction of inliers is small. In the limit, trying all pairs of matches involves $\Omega(n^2)$ RANSAC iterations. Moreover, our methods enhance the quality of the posing considerably [13], since each generated candidate pose is close to (i.e., consistent with) with many of the correspondences.

---

[1] As we will see later in detail, each correspondence imposes two constraints on the camera pose.

Our results. We formalize the four degree-of-freedom camera pose problem as an approximate incidences problem in Section 2. Each 2D-3D correspondence is represented as a two-dimensional surface in the 4-dimensional pose-space, which is the locus of all possible positions and orientations of the camera that fit the correspondence exactly. Ideally, we would like to find a point (a pose) that lies on as many surfaces as possible, but since we expect the data to be noisy, and the exact problem is inefficient to solve anyway, we settle for an approximate version, in which we seek a point with a large number of approximate incidences with the surfaces.

Formally, we solve the following problem. We have an error parameter $\varepsilon > 0$, we lay down a grid on $[0, 1]^d$ of side length ε, and compute, for each vertex v of the grid, a count $I(v)$ of surfaces that are approximately incident to v, so that (i) every surface that is ε-incident to v is counted in $I(v)$, and (ii) every surface that is counted in $I(v)$ is $\alpha\varepsilon$-incident to v, for some small constant $\alpha > 1$ (but not all $\alpha\varepsilon$-incident surfaces are necessarily counted). We output the grid vertex v with the largest count $I(v)$ (or a list of vertices with the highest counts, if so desired).

As we will comment later, (a) restricting the algorithm to grid vertices only does not miss a good pose v: a vertex of the grid cell containing v serves as a good substitute for v, and (b) we have no real control on the value of $I(v)$, which might be much larger than the number of surfaces that are ε-incident to v, but all the surfaces that we count are "good" – they are reasonably close to v. In the computer vision application, and in many related applications, neither of these issues is significant.

We give three algorithms for this camera-pose approximate-incidences problem. The first algorithm simply computes the grid cells that each surface intersects, and considers the number of intersecting surfaces per cell as its approximate ε-incidences count. This method takes time $O\left(\frac{n}{\varepsilon^2}\right)$ for all vertices of our ε-size grid. We then describe a faster algorithm using geometric duality, in Section 3. It uses a coarser grid in the primal space and switches to a dual 5-dimensional space (a 5-tuple is needed to specify a 2D-3D correspondence and its surface, now dualized to a point). In the dual space each query (i.e., a vertex of the grid) becomes a 3-dimensional surface, and each original 2-dimensional surface in the primal 4-dimensional space becomes a point. This algorithm takes $O\left(\frac{n^{3/5}}{\varepsilon^{14/5}} + n + \frac{1}{\varepsilon^4}\right)$ time, and is asymptotically faster than the simple algorithm for $n > 1/\varepsilon^2$.

Finally, we give a general method for constructing an approximate incidences data structure for general k-dimensional algebraic surfaces (that satisfy certain mild conditions) in $\mathbb{R}^d$, in Section 4. It extends the technique of Fonseca and Mount [4], designed for the case of hyperplanes, and takes $O(n + \text{poly}(1/\varepsilon))$ time, where the degree of the polynomial in $1/\varepsilon$ depends on the number of parameters needed to specify a surface, the dimension of the surfaces, and the dimension of the ambient space. We first present and analyze this technique in full generality, and then apply it to the surfaces obtained for our camera posing problem. In this case, the data structure requires $O(n + 1/\varepsilon^6)$ storage and is constructed in roughly the same time. This is asymptotically faster than our primal-dual scheme when $n \geq 1/\varepsilon^{16/3}$ (for $n \geq 1/\varepsilon^7$ the $O(n)$ term dominates and these two methods are asymptotically the same). Due to its generality, the latter technique is easily adapted to other surfaces and thus is of general interest and potential. In contrast, the primal-dual method requires nontrivial adaptation as it switches from one approximate-incidences problem to another and the dual space and its distance function depend on the type of the input surfaces.

We implemented our algorithms and compared their performance on real and synthetic data. Our experimentation shows that, for commonly used values of n and ε in practical scenarios ($n \in [8K, 32K]$, $\varepsilon \in [0.02, 0.03]$), the primal-dual scheme is considerably faster than

the other algorithms, and should thus be the method of choice. Due to lack of space, the experimentation details are omitted in this version, with the exception of a few highlights. They can be found in the full version of the paper [2].

2 From camera positioning to approximate incidences

Suppose we are given a pre-computed three-dimensional scene and a two-dimensional picture of it. Our goal is to deduce from this image the location and orientation of the camera in the scene. In general, the camera, as a rigid body in 3-space, has six degrees of freedom, three of translation and three of rotation (commonly referred to as the *yaw, pitch and roll*). We simplify the problem by making the realistic assumption, that the vertical direction of the scene is known in the camera coordinate frame (e.g., estimated by en inertial sensor on smart phones). This allows us to rotate the camera coordinate frame such that its z-axis is parallel to the world z-axis, thereby fixing the pitch and roll of the camera and leaving only four degrees of freedom (x, y, z, θ), where $c = (x, y, z)$ is the location of the camera center, say, and θ is its yaw, i.e. horizontal the orientation of the optical axis around the vertical direction. See Figure 1.
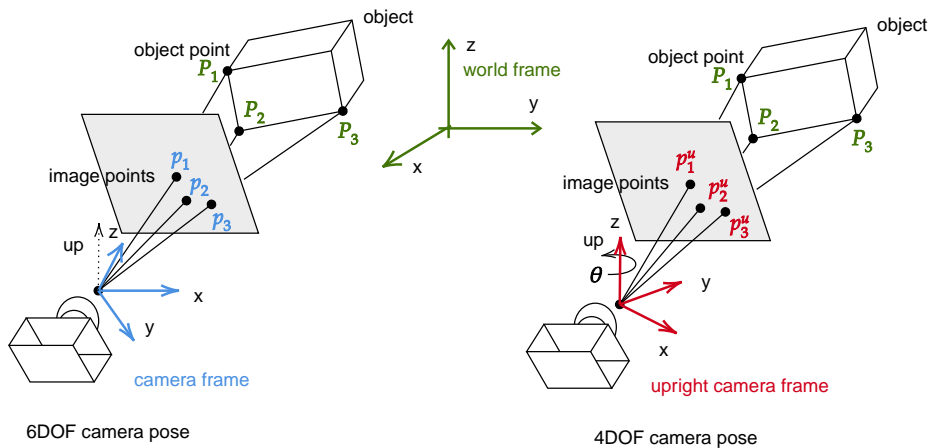
Figure 1 With the knowledge of a common vertical direction between the camera and world frame the general 6DoF camera posing problem reduces to estimating 4 parameters. This is the setup we consider in our work.

By preprocessing the scene, we record the spatial coordinates $w = (w_1, w_2, w_3)$ of a discrete (large) set of salient points. We assume that some (ideally a large number) of the distinguished points are identified in the camera image, resulting in a set of image-to-scene correspondences. Each correspondence $\mathbf{w} = \{w_1, w_2, w_3, \xi, \eta\}$ is parameterized by five parameters, the spatial position w and the position $v = (\xi, \eta)$ in the camera plane of view of the same salient point. Our goal is to find a camera pose (x, y, z, θ) so that as many correspondences as possible are (approximately) *consistent* with it, i.e., the ray from the camera center c to w goes approximately through (ξ, η) in the image plane, when the yaw of the camera is θ.

2.1 Camera posing as an ε-incidences problem

Each correspondence and its 5-tuple $\mathbf{w}$ define a two-dimensional surface $\sigma_{\mathbf{w}}$ in parametric 4-space, which is the locus of all poses (x, y, z, θ) of the camera at which it sees w at coordinates (ξ, η) in its image. For n correspondences, we have a set of n such surfaces. We prove that each point in the parametric 4-space of camera poses that is close to a surface $\sigma_{\mathbf{w}}$, in a suitable metric defined in that 4-space, represents a camera pose where w is projected to a point in the camera viewing plane that is close to (ξ, η), and vice versa (see Section 2.2 for the actual expressions for these projections). Therefore, a point in 4-space that is close to a large number of surfaces represents a camera pose with many approximately consistent correspondences, which is a strong indication of being close to the correct pose.

Extending the notation used in the earlier work [3], we say that a point q is ε-*incident* to a surface σ if $\mathrm{dist}(q, \sigma) \le \varepsilon$. Our algorithms approximate, for each vertex of a grid G^ε of side length ε, the number of ε-incident surfaces and suggest the vertex with the largest count as the best candidate for the camera pose. This work extends the approximate incidences methodology in [3] to the (considerably more involved) case at hand.

2.2 The surfaces $\sigma_{\mathbf{w}}$

Let $w = (w_1, w_2, w_3)$ be a salient point in $\mathbb{R}^3$, and assume that the camera is positioned at $(c, \theta) = (x, y, z, \theta)$. We represent the orientation of the vector $w - c$, within the world frame, by its spherical coordinates (φ, ψ), except that, unlike the standard convention, we take ψ to be the angle with the xy-plane (rather than with the z-axis):

$$\tan \psi = \frac{w_3 - z}{\sqrt{(w_1 - x)^2 + (w_2 - y)^2}} \qquad \tan \varphi = \frac{w_2 - y}{w_1 - x}$$

In the two-dimensional frame of the camera the (ξ, η)-coordinates model the *view* of w, which differs from above polar representation of the vector $w - c$ only by the polar orientation θ of the viewing plane itself. Writing κ for $\tan \theta$, we have

$$\xi = \tan(\varphi - \theta) = \frac{\tan \varphi - \tan \theta}{1 + \tan \varphi \tan \theta} = \frac{(w_2 - y) - \kappa(w_1 - x)}{(w_1 - x) + \kappa(w_2 - y)}, \tag{1}$$

$$\eta = \tan \psi = \frac{w_3 - z}{\sqrt{(w_1 - x)^2 + (w_2 - y)^2}}.$$

We note that using $\tan \theta$ does not distinguish between θ and $\theta + \pi$, but we will restrict θ to lie in $[-\pi/4, \pi/4]$ or in similar narrower ranges, thereby resolving this issue.

We use $\mathbb{R}^4$ with coordinates (x, y, z, κ) as our primal space, where each point models a possible pose of the camera. Each correspondence $\mathbf{w}$ is parameterized by the triple (w, ξ, η), and defines a two-dimensional algebraic surface $\sigma_{\mathbf{w}}$ of degree at most 4, whose equations (in x, y, z, κ) are given in (1). It is the locus of all camera poses $v = (x, y, z, \kappa)$ at which it sees w at image coordinates (ξ, η). We can rewrite these equations into the following parametric representation of $\sigma_{\mathbf{w}}$, expressing z and κ as functions of x and y:

$$\kappa = \frac{(w_2 - y) - \xi(w_1 - x)}{(w_1 - x) + \xi(w_2 - y)} \qquad z = w_3 - \eta\sqrt{(w_1 - x)^2 + (w_2 - y)^2}. \tag{2}$$

For a camera pose $v = (x, y, z, \kappa)$, and a point $w = (w_1, w_2, w_3)$, we write

$$F(v; w) = \frac{(w_2 - y) - \kappa(w_1 - x)}{(w_1 - x) + \kappa(w_2 - y)} \qquad G(v; w) = \frac{w_3 - z}{\sqrt{(w_1 - x)^2 + (w_2 - y)^2}}. \tag{3}$$

In this notation we can write the Equations (1) characterizing $\sigma_{\mathbf{w}}$ (when regarded as equations in v) as $\xi = F(v; w)$ and $\eta = G(v; w)$.

2.3 Measuring proximity

Given a guessed pose $v = (x, y, z, \kappa)$ of the camera, we want to measure how well it fits the scene that the camera sees. For this, given a correspondence $\mathbf{w} = (w, \xi, \eta)$, we define the *frame distance* fd between v and $\mathbf{w}$ as the L_∞-distance between (ξ, η) and (ξ_v, η_v), where, as in Eq. (3), $\xi_v = F(v; w)$, $\eta_v = G(v; w)$. That is,

$$\mathsf{fd}(v, \mathbf{w}) = \max \left\{ |\xi_v - \xi|, \ |\eta_v - \eta| \right\}. \tag{4}$$

Note that (ξ_v, η_v) are the coordinates at which the camera would see w if it were placed at position v, so the frame distance is the L_∞-distance between these coordinates and the actual coordinates (ξ, η) at which the camera sees w; this serves as a natural measure of how close v is to the actual pose of the camera.

We are given a viewed scene of n distinguished points (correspondences) $\mathbf{w} = (w, \xi, \eta)$. Let S denote the set of n surfaces $\sigma_\mathbf{w}$, representing these correspondences. We assume that the salient features w and the camera are all located within some bounded region, say $[0, 1]^3$. The replacement of θ by $\kappa = \tan \theta$ makes its range unbounded, so we break the problem into four subproblems, in each of which θ is confined to some sector. In the first subproblem we assume that $-\pi/4 \le \theta \le \pi/4$, so $-1 \le \kappa \le 1$. The other three subproblems involve the ranges $[\pi/4, 3\pi/4]$, $[3\pi/4, 5\pi/4]$, and $[5\pi/4, 7\pi/4]$. We only consider here the first subproblem; the treatment of the others is fully analogous. In each such range, replacing θ by $\tan \theta$ does not incur the ambiguity of identifying θ with $\theta + \pi$.

Given an error parameter $\varepsilon > 0$, we seek an approximate pose v of the camera, at which many correspondences $\mathbf{w}$ are within frame distance at most ε from v, as given in (4).

The following two lemmas relate our frame distance to the Euclidean distance. Their (rather technical) proofs are given in the full version of this paper [2].

▶ **Lemma 1.** *Let $v = (x, y, z, \kappa)$, and let $\sigma_\mathbf{w}$ be the surface associated with a correspondence $\mathbf{w} = \{w_1, w_2, w_3, \xi, \eta\}$. Let v' be a point on $\sigma_\mathbf{w}$ such that $|v - v'| \le \varepsilon$ (where $|\cdot|$ denotes the Euclidean norm). If*
   (i) $|(w_1 - x) + \kappa(w_2 - y)| \ge a > 0$, and
   (ii) $(w_1 - x)^2 + (w_2 - y)^2 \ge a > 0$, for some absolute constant a,
then $\mathsf{fd}(v, \mathbf{w}) \le \beta\varepsilon$ for some constant β that depends on a.

Informally, Condition (i) requires that the absolute value of the $\xi = \tan(\varphi - \theta)$ coordinate of the position of w in the viewing plane, with the camera positioned at v, is not too large (i.e., that $|(\varphi - \theta)|$ is not too close to $\pi/2$). We can ensure this property by restricting the camera image to some suitably bounded ξ-range.

Similarly, Condition (ii) requires that the xy-projection of the vector $w - c$ is not too small. It can be violated in two scenarios. Either we look at a data point that is too close to c, or we see it looking too much "upwards" or "downwards". We can ensure that the latter situation does not arise, by restricting the camera image, as in the preceding paragraph, to some suitably bounded η-range too. That done, we ensure that the former situation does not arise by requiring that the physical distance between c and w be at least some multiple of a.

The next lemma establishes the converse connection.

▶ **Lemma 2.** *Let $v = (x, y, z, \kappa)$ be a camera pose and $\mathbf{w} = \{w_1, w_2, w_3, \xi, \eta\}$ a correspondence, such that $\mathsf{fd}(v, \mathbf{w}) \le \varepsilon$. Assume that $|(w_1 - x) + \xi(w_2 - y)| \ge a > 0$, for some absolute constant a, and consider the point $v' = (x, y, z', \kappa') \in \sigma_\mathbf{w}$ where (see Eq. (2))*

$$z' = w_3 - \eta\sqrt{(w_1 - x)^2 + (w_2 - y)^2} \qquad \kappa' = \frac{(w_2 - y) - \xi(w_1 - x)}{(w_1 - x) + \xi(w_2 - y)}.$$

Then $|z - z'| \leq \sqrt{2}\varepsilon$ and $|\kappa - \kappa'| \leq c\varepsilon$, for some constant c, again depending on a.

Informally, the condition $|(w_1 - x) + \xi(w_2 - y)| \geq a > 0$ means that the orientation of the camera, when it is positioned at (x, y) and sees w at coordinate ξ of the viewing plane is not too close to $\pm\pi/2$. This is a somewhat artificial constraint that is satisfied by our restriction on the allowed yaws of the camera (the range of κ).

A Simple algorithm. Using Lemma 2 and Lemma 1 we can derive a simple naive solution which does not require any of the sophisticated machinery developed in this work. We construct a grid G over $Q = [0, 1]^3 \times [-1, 1]$, of cells τ, each of dimensions $\varepsilon \times \varepsilon \times 2\sqrt{2}\varepsilon \times 2a\varepsilon$, where a is the constant of Lemma 2. We use this non-square grid G since we want to find ε-approximate incidences in terms of frame distance. For each cell τ of G we compute the number of surfaces $\sigma_{\mathbf{w}}$ that intersect τ. This gives an approximate incidences count for the center of τ. Further details and a precise statement can be found in the full version [2].

3 Primal-dual algorithm for geometric proximity

Following the general approach in [3], we use a suitable duality, with some care. We write $\varepsilon = 2\gamma\delta_1\delta_2$, for suitable parameters γ, and $\varepsilon/(2\gamma) \leq \delta_1, \delta_2 \leq 1$, whose concrete values are fixed later, and apply the decomposition scheme developed in [3] tailored to the case at hand. Specifically, we consider the coarser grid G_{δ_1} in the primal space, of cell dimensions $\delta_1 \times \delta_1 \times \sqrt{2}\delta_1 \times c\delta_1$, where c is is the constant from Lemma 2, that tiles up the domain $Q = [0, 1]^3 \times [-1, 1]$ of possible camera positions. For each cell τ of G_{δ_1}, let S_τ denote the set of surfaces that cross either τ or one of the eight cells adjacent to τ in the (z, κ)-directions.[2] The duality is illustrated in Figure 2.
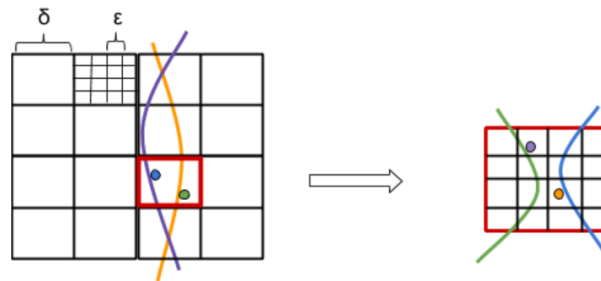
Figure 2 A schematic illustration of our duality-based algorithm.

We discretize the set of all possible positions of the camera by the vertices of the finer grid G_ε, defined as G_{δ_1}, with ε replacing δ_1, that tiles up Q. The number of these candidate positions is $m := O(1/\varepsilon^4)$. For each vertex $q \in G_\varepsilon$, we want to approximate the number of surfaces that are ε-incident to q, and output the vertex with the largest count as the best candidate for the position of the camera. Let V_τ be the subset of G_ε contained in τ. We ensure that the boxes of G_{δ_1} are pairwise disjoint by making them half open, in the sense that if $(x_0, y_0, z_0, \kappa_0)$ is the vertex of a box that has the smallest coordinates, then the box is defined by $x_0 \leq x < x_0 + \delta_1$, $y_0 \leq y < y_0 + \delta_1$, $z_0 \leq z < z_0 + \sqrt{2}\delta_1$, $\kappa_0 \leq \kappa < \kappa_0 + c\delta_1$. This makes the sets V_τ pairwise disjoint as well. Put $m_\tau = |V_\tau|$ and $n_\tau = |S_\tau|$. We have

---

[2] The choice of z, κ is arbitrary, but it is natural for the analysis, given in the full version [2].

$m_\tau = O\left((\delta_1/\varepsilon)^4\right)$ for each τ. Since the surfaces $\sigma_\mathbf{w}$ are two-dimensional algebraic surfaces of constant degree, each of them crosses $O(1/\delta_1^2)$ cells of G_{δ_1}, so we have $\sum_\tau n_\tau = O(n/\delta_1^2)$.

We now pass to the dual five-dimensional space. Each point in that space represents a correspondence $\mathbf{w} = (w_1, w_2, w_3, \xi, \eta)$. We use the first three components (w_1, w_2, w_3) as the first three coordinates, but modify the ξ- and η-coordinates in a manner that depends on the primal cell τ. Let $c_\tau = (x_\tau, y_\tau, z_\tau, \kappa_\tau)$ be the midpoint of the primal box τ. For each $\sigma_\mathbf{w} \in S_\tau$ we map $\mathbf{w} = (w, \xi, \eta)$, where $w = (w_1, w_2, w_3)$, to the point $\mathbf{w}_\tau = (w_1, w_2, w_3, \xi_\tau, \eta_\tau)$, where $\xi_\tau = \xi - F(c_\tau; w)$ and $\eta_\tau = \eta - G(c_\tau; w)$, with F and G as given in (3). We have

▶ **Corollary 3.** *If $\sigma_\mathbf{w}$ crosses τ then $|\xi_\tau|, |\eta_\tau| \le \gamma\delta_1$, for some absolute constant γ, provided that the following two properties hold, for some absolute constant $a > 0$ (the constant γ depends on a).*
  (i) $|(w_1 - x_\tau) + \kappa_\tau(w_2 - y_\tau)| \ge a$, *and*
  (ii) $(w_1 - x_\tau)^2 + (w_2 - y_\tau)^2 \ge a$, *where (x_τ, y_τ) are the (x, y)-coordinates of the center of τ.*

Proof. If $\sigma_\mathbf{w} \in S_\tau$ then it contains a point v' such that $|v' - c_\tau| \le c'\delta_1$, for a suitable absolute constant c' (that depends on c). We now apply Lemma 1, recalling (4). ◀

We take the γ provided by Corollary 3 as the γ in the definition of δ_1 and δ_2. We map each point $v \in V_\tau$ to the dual surface $\sigma_v^* = \sigma_{v;\tau}^* = \{\mathbf{w}_\tau \mid v \in \sigma_\mathbf{w}\}$. Using (3), we have

$$\sigma_{v;\tau}^* = \{(w, \ F(v; w) - F(c_\tau; w), G(v; w) - G(c_\tau; w)) \mid w = (w_1, w_2, w_2) \in [0, 1]^3\}.$$

By Corollary 3, the points $\mathbf{w}_\tau$, for the surfaces $\sigma_\mathbf{w}$ that cross τ, lie in the region $R_\tau = [0, 1]^3 \times [-\gamma\delta_1, \gamma\delta_1]^2$. We partition R_τ into a grid G_{δ_2} of $1/\delta_2^5$ small congruent boxes, each of dimensions $\delta_2 \times \delta_2 \times \delta_2 \times (2\gamma\delta_1\delta_2) \times (2\gamma\delta_1\delta_2) = \delta_2 \times \delta_2 \times \delta_2 \times \varepsilon \times \varepsilon$.

Exactly as in the primal setup, we make each of these boxes half-open, thereby making the sets of dual vertices in the smaller boxes pairwise disjoint. We assign to each of these dual cells τ^* the set $S_{\tau^*}^*$ of dual points that lie in τ^*, and the set $V_{\tau^*}^*$ of the dual surfaces that cross either τ^* or one of the eight cells adjacent to τ^* in the (ξ_τ, η_τ)-directions. Put $n_{\tau^*} = |S_{\tau^*}^*|$ and $m_{\tau^*} = |V_{\tau^*}^*|$. Since the dual cells are pairwise disjoint, we have $\sum_{\tau^*} n_{\tau^*} = n_\tau$. Since the dual surfaces are three-dimensional algebraic surfaces of constant degree, each of them crosses $O(1/\delta_2^3)$ grid cells, so $\sum_{\tau^*} m_{\tau^*} = O\left(m_\tau/\delta_2^3\right)$.

We compute, for each dual surface σ_v^*, the sum $\sum_{\tau^*} |S_{\tau^*}^*|$, over the dual cells τ^* that are either crossed by σ_v^* or that one of their adjacent cells in the (ξ_τ, η_τ)-directions is crossed by σ_v^*. We output the vertex v of G_ε with the largest resulting count, over all primal cells τ.

The following theorem establishes the correctness of our technique. Its proof is given in Appendix B of the full version [2].

▶ **Theorem 4.** *Suppose that for every cell $\tau \in G_{\delta_1}$ and for every point $v = (x, y, z, \kappa) \in V_\tau$ and every $\mathbf{w} = ((w_1, w_2, w_3), \xi, \eta)$ such that $\sigma_\mathbf{w}$ intersects either τ or one of its adjacent cells in the (ξ_τ, η_τ)-directions, we have that, for some absolute constant $a > 0$,*
  (i) $|(w_1 - x) + \kappa(w_2 - y)| \ge a$,
  (ii) $(w_1 - x)^2 + (w_2 - y)^2 \ge a$, *and*
  (iii) $|(w_1 - x) + \xi(w_2 - y)| \ge a$.
Then (a) For each $v \in V$, every pair $(v, \mathbf{w})$ at frame distance $\le \varepsilon$ is counted (as an ε-incidence of v) by the algorithm. (b) For each $v \in V$, every pair $(v, \mathbf{w})$ that we count lies at frame distance $\le \alpha\varepsilon$, for some constant $\alpha > 0$ depending on a.

3.1 Running time analysis

The cost of the algorithm is clearly proportional to $\sum_\tau \sum_{\tau^*} (m_{\tau^*} + n_{\tau^*})$, over all primal cells τ and the dual cells τ^* associated with each cell τ. We have

$$\sum_\tau \sum_{\tau^*} (m_{\tau^*} + n_{\tau^*}) = O\left(\sum_\tau \left(m_\tau / \delta_2^3 + n_\tau \right) \right) = O\left(m/\delta_2^3 + n/\delta_1^2 \right).$$

Optimizing the choice of δ_1 and δ_2, we choose $\delta_1 = \left(\frac{\varepsilon^3 n}{m} \right)^{1/5}$ and $\delta_2 = \left(\frac{\varepsilon^2 m}{n} \right)^{1/5}$. These choices make sense as long as each of δ_1, δ_2 lies between $\varepsilon/(2\gamma)$ and 1. That is, $\frac{\varepsilon}{2\gamma} \leq \left(\frac{\varepsilon^3 n}{m} \right)^{1/5} \leq 1$ and $\frac{\varepsilon}{2\gamma} \leq \left(\frac{\varepsilon^2 m}{n} \right)^{1/5} \leq 1$, or $c' \varepsilon^2 m \leq n \leq \frac{c'' m}{\varepsilon^3}$, where c' and c'' are absolute constants (that depend on γ).

If $n < c' \varepsilon^2 m$, we use only the primal setup, taking $\delta_1 = \varepsilon$ (for the primal subdivision). The cost is then $O\left(n/\varepsilon^2 + m \right) = O\left(m \right)$. Similarly, if $n > \frac{c'' m}{\varepsilon^3}$, we use only the dual setup, taking $\delta_1 = 1$ and $\delta_2 = \varepsilon/(2\gamma)$, and the cost is thus $O\left(n + m/\varepsilon^3 \right) = O(n)$. Adding everything together, to cover all three subranges, the running time is then $O\left(\frac{m^{2/5} n^{3/5}}{\varepsilon^{6/5}} + n + m \right)$. Substituting $m = O\left(1/\varepsilon^4 \right)$, we get a running time of $O\left(\frac{n^{3/5}}{\varepsilon^{14/5}} + n + \frac{1}{\varepsilon^4} \right)$. The first term dominates when $n = \Omega(\frac{1}{\varepsilon^2})$ and $n = O(\frac{1}{\varepsilon^7})$. In conclusion, we have the following result.

▶ **Theorem 5.** *Given n data points that are seen (and identified) in a two-dimensional image taken by a vertically positioned camera, and an error parameter $\varepsilon > 0$, where the viewed points satisfy the assumptions made in Theorem 4, we can compute, in $O\left(\dfrac{n^{3/5}}{\varepsilon^{14/5}} + n + \dfrac{1}{\varepsilon^4} \right)$ time, a vertex v of G_ε that maximizes the approximate count of ε-incident correspondences, where "approximate" means that every correspondence $\mathbf{w}$ whose surface $\sigma_{\mathbf{w}}$ is at frame distance at most ε from v is counted and every correspondence that we count lies at frame distance at most $\alpha\varepsilon$ from v, for some fixed constant α.*

Restricting ourselves only to grid vertices does not really miss any solution. We only lose a bit in the quality of approximation, replacing ε by a slightly large constant multiple thereof, when we move from the best solution to a vertex of its grid cell.

4 Geometric proximity via canonical surfaces

In this section we present a general technique to preprocess a set of algebraic surfaces into a data structure that can answer approximate incidences queries. In this technique we round the n original surfaces into a set of canonical surfaces, whose size depends only on ε, such that each original surface has a canonical surface that is "close" to it. Then we build an octree-based data structure for approximate incidences queries with respect to the canonical surfaces. However, to reduce the number of intersections between the cells of the octree and the surfaces, we further reduce the number of surfaces as we go from one level of the octree to the next, by rounding them in a coarser manner into a smaller set of surfaces.

This technique has been introduced by Fonseca and Mount [4] for the case of hyperplanes. We describe as a warmup step, in Appendix C of the full version [2], our interpretation of their technique applied to hyperplanes. We then extend here the technique to general surfaces, and apply it to the specific instance of 2-surfaces in 4-space that arise in the camera pose problem.

We have a set S of n k-dimensional surfaces in $\mathbb{R}^d$ that cross the unit cube $[0,1]^d$, and a given error parameter ε. We assume that each surface $\sigma \in S$ is given in parametric form, where the first k coordinates are the parameters, so its equations are

$$x_j = F_j^{(\sigma)}(x_1, \ldots, x_k), \qquad \text{for } j = k+1, \ldots, d.$$

Moreover, we assume that each $\sigma \in S$ is defined in terms of ℓ *essential parameters* $\mathbf{t} = (t_1, \ldots, t_\ell)$, and $d-k$ additional *free additive parameters* $\mathbf{f} = (f_{k+1}, \ldots, f_d)$, one free parameter for each dependent coordinate. Concretely, we assume that the equations defining the surface $\sigma \in S$, parameterized by $\mathbf{t}$ and $\mathbf{f}$ (we then denote σ as $\sigma_{\mathbf{t},\mathbf{f}}$), are

$$x_j = F_j(\mathbf{x}; \mathbf{t}) + f_j = F_j(x_1, \ldots, x_k; t_1, \ldots, t_\ell) + f_j, \qquad \text{for } j = k+1, \ldots, d.$$

For each equation of the surface that does not have a free parameter in the original expression, we introduce an *artificial* free parameter, and initialize its value to 0. (We need this separation into essential and free parameters for technical reasons that will become clear later.) We assume that $\mathbf{t}$ (resp., $\mathbf{f}$) varies over $[0,1]^\ell$ (resp., $[0,1]^{d-k}$).

Remark.   The distinction between free and essential parameters seems to be artificial, but yet free parameters do arise in certain basic cases, such as the case of hyperplanes discussed in Appendix C of [2]. In the case of our 2-surfaces in 4-space, the parameter w_3 is free, and we introduce a second artificial free parameter into the equation for κ. The number of essential parameters is $\ell = 4$ (they are w_1, w_2, ξ, and η).

We assume that the functions F_j are all continuous and differentiable, in all of their dependent variables $\mathbf{x}$, $\mathbf{t}$ and $\mathbf{f}$ (this is a trivial assumption for $\mathbf{f}$), and that they satisfy the following two conditions.

(i) Bounded gradients. $|\nabla_{\mathbf{x}} F_j(\mathbf{x}; \mathbf{t})| \leq c_1$, $\quad |\nabla_{\mathbf{t}} F_j(\mathbf{x}; \mathbf{t})| \leq c_1$, for each $j = k+1, \ldots, d$, for any $\mathbf{x} \in [0,1]^k$ and any $\mathbf{t} \in [0,1]^\ell$, where c_1 is some absolute constant. Here $\nabla_{\mathbf{x}}$ (resp., $\nabla_{\mathbf{t}}$) means the gradient with respect to only the variables $\mathbf{x}$ (resp., $\mathbf{t}$).

(ii) Lipschitz gradients. $|\nabla_{\mathbf{x}} F_j(\mathbf{x}; \mathbf{t}) - \nabla_{\mathbf{x}} F_j(\mathbf{x}; \mathbf{t}')| \leq c_2 |\mathbf{t} - \mathbf{t}'|$, for each $j = k+1, \ldots, d$, for any $\mathbf{x} \in [0,1]^k$ and any $\mathbf{t}, \mathbf{t}' \in [0,1]^\ell$, where c_2 is some absolute constant. This assumption is implied by the assumption that all the eigenvalues of the mixed part of the Hessian matrix $\nabla_{\mathbf{t}} \nabla_{\mathbf{x}} F_j(\mathbf{x}; \mathbf{t})$ have absolute value bounded by c_2.

4.1   Canonizing the input surfaces

We first replace each surface $\sigma_{\mathbf{t},\mathbf{f}} \in S$ by a canonical "nearby" surface $\sigma_{\mathbf{s},\mathbf{g}}$. Let $\varepsilon' = \frac{\varepsilon}{c_2 \log(1/\varepsilon)}$ where c_2 is the constant from Condition (ii). We get $\mathbf{s}$ from $\mathbf{t}$ (resp., $\mathbf{g}$ from $\mathbf{f}$) by rounding each coordinate in the essential parametric domain L (resp., in the parametric domain Φ) to a multiple of $\varepsilon'/(\ell+1)$. Note that each of the artificial free parameters (those that did not exist in the original equations) has the initial value 0 for all surfaces, and remains 0 in the rounded surfaces. We get $O\left((1/\varepsilon')^{\ell'}\right)$ *canonical* rounded surfaces, where $\ell' \geq \ell$ is the number of *original* parameters, that is, the number of essential parameters plus the number of non-artificial free parameters; in the worst case we have $\ell' = \ell + d - k$.

For a surface $\sigma_{\mathbf{t},\mathbf{f}}$ and its rounded version $\sigma_{\mathbf{s},\mathbf{g}}$ we have, for each j,

$$|(F_j(\mathbf{x}; \mathbf{t}) + f_j) - (F_j(\mathbf{x}; \mathbf{s}) + g_j)| \leq |\nabla_{\mathbf{t}} F_j(\mathbf{x}; \mathbf{t}')| \cdot |\mathbf{t} - \mathbf{s}| + |f_j - g_j|$$
$$\leq c_1 |\mathbf{t} - \mathbf{s}| + |f_j - g_j| \leq (c_1 + 1)\varepsilon',$$

where $\mathbf{t}'$ is some intermediate value, which is irrelevant due to Condition (i).

We will use the ℓ_2-norm of the difference vector $((F_j(\mathbf{x}; \mathbf{t}) + f_j) - (F_j(\mathbf{x}; \mathbf{s}) + g_j))_{j=k+1}^{d}$ as the measure of proximity between the surfaces $\sigma_{\mathbf{t},\mathbf{f}}$ and $\sigma_{\mathbf{s},\mathbf{g}}$ at $\mathbf{x}$, and denote it as $\mathsf{dist}(\sigma_{\mathbf{t},\mathbf{f}}, \sigma_{\mathbf{s},\mathbf{g}}; \mathbf{x})$. The maximum $\mathsf{dist}(\sigma_{\mathbf{t},\mathbf{f}}, \sigma_{\mathbf{s},\mathbf{g}}) := \max_{\mathbf{x} \in [0,1]^k} \mathsf{dist}(\sigma_{\mathbf{t},\mathbf{f}}, \sigma_{\mathbf{s},\mathbf{g}}; \mathbf{x})$ measures the global proximity of the two surfaces. (Note that it is an upper bound on the Hausdorff distance between the two surfaces.) We thus have $\mathsf{dist}(\sigma_{\mathbf{t},\mathbf{f}}, \sigma_{\mathbf{s},\mathbf{g}}) \leq (c_1 + 1)\varepsilon'$ when $\sigma_{\mathbf{s},\mathbf{g}}$ is the canonical surface approximating $\sigma_{\mathbf{t},\mathbf{f}}$.

We define the *weight* of each canonical surface to be the number of original surfaces that got rounded to it, and we refer to the set of all canonical surfaces by S^c.

4.2 Approximately counting ε-incidences

We describe an algorithm for approximating the ε-incidences counts of the surfaces in S and the vertices of a grid G of side length 4ε.

We construct an octree decomposition of $\tau_0 := [0,1]^d$, all the way to subcubes of side length 4ε such that each vertex of G is the center of a leaf-cube. We propagate the surfaces of S^c down this octree, further rounding each of them within each subcube that it crosses.

The root of the octree corresponds to τ_0, and we set $S_{\tau_0} = S^c$. At level $j \geq 1$ of the recursion, we have subcubes τ of τ_0 of side length $\delta = 1/2^j$. For each such τ, we set $\tilde{S}_\tau$ to be the subset of the surfaces in $S_{p(\tau)}$ (that have been produced at the parent cube $p(\tau)$ of τ) that intersect τ. We now show how to further round the surfaces of $\tilde{S}_\tau$, so as to get a coarser set S_τ of surfaces that we associate with τ, and that we process recursively within τ.

At any node τ at level j of our rounding process, each surface σ of S_τ is of the form $x_j = H_j(\mathbf{x}; \mathbf{t}) + f_j$, for $j = k+1, \ldots, d$ where $\mathbf{x} = (x_1, \ldots, x_k)$, and $\mathbf{t} = (t_1, \ldots, t_\ell)$.

(a) For each $j = k+1, \ldots, d$ the function H_j is a translation of F_j. That is $H_j(\mathbf{x}; \mathbf{t}) = F_j(\mathbf{x}; \mathbf{t}) + c$ for some constant c. Thus the gradients of H_j also satisfy Conditions (i) and (ii).

(b) $\mathbf{t}$ is some vector of ℓ essential parameters, and each coordinate of $\mathbf{t}$ is an integer multiple of $\frac{\varepsilon'}{(\ell+1)\delta}$, where $\delta = 1/2^j$.

(c) $\mathbf{f} = (f_{k+1}, \ldots, f_d)$ is a vector of free parameters, each is a multiple of $\varepsilon'/(\ell+1)$.

Note that the surfaces in $S_{\tau_0} = S^c$, namely the set of initial canonical surfaces constructed in Section 4.1, are of this form (for $j = 0$ and $H_j = F_j$). We get S_τ from $\tilde{S}_\tau \subseteq S_{p(\tau)}$ by the following steps. The first step just changes the presentation of τ and $\tilde{S}_\tau$, and the following steps do the actual rounding to obtain S_τ.

1. Let $(\xi_1, \ldots, \xi_k, \xi_{k+1}, \ldots, \xi_d)$ be the point in τ of smallest coordinates and set $\xi = (\xi_1, \ldots, \xi_k)$. We rewrite the equations of each surface of $\tilde{S}_\tau$ as follows: $x_j = G_j(\mathbf{x}; \mathbf{t}) + f_j'$, for $j = k+1, \ldots, d$, where $G_j(\mathbf{x}; \mathbf{t}) = H_j(\mathbf{x}; \mathbf{t}) - H_j(\xi; \mathbf{t}) + \xi_j$, and $f_j' = f_j + H_j(\xi; \mathbf{t}) - \xi_j$, for $j = k+1, \ldots, d$. Note that in this reformulation we have not changed the essential parameters, but we did change the free parameters from f_j to f_j', where f_j' depends on f_j, $\mathbf{t}$, ξ, and ξ_j. Note also that $G_j(\xi; \mathbf{t}) = \xi_j$ for $j = k+1, \ldots, d$.

2. We replace the essential parameters $\mathbf{t}$ of a surface $\sigma_{\mathbf{t},\mathbf{f}}$ by $\mathbf{s}$, which we obtain by rounding each coordinate of $\mathbf{t}$ to the nearest integer multiple of $\frac{\varepsilon'}{(\ell+1)\delta}$. So the rounded surface has the equations $x_j = G_j(\mathbf{x}; \mathbf{s}) + f_j'$, for $j = k+1, \ldots, d$. Note that we also have that $G_j(\xi; \mathbf{s}) = \xi_j$, for $j = k+1, \ldots, d$.

3. For each surface, we round each free parameter f_j', $j = k+1, \ldots, d$, to an integral multiple of $\frac{\varepsilon'}{\ell+1}$, and denote the rounded vector by $\mathbf{g}$. Our final equations for each rounded surface that we put in S_τ are $x_j = G_j(\mathbf{x}; \mathbf{s}) + g_j$ for $j = k+1, \ldots, d$.

By construction, when $\mathbf{t}_1$ and $\mathbf{f}_1'$ and $\mathbf{t}_2$ and $\mathbf{f}_2'$ get rounded to the same vectors $\mathbf{s}$ and $\mathbf{g}$ then the corresponding two surfaces in $\tilde{S}_\tau$ get rounded to the same surface in S_τ. The weight of each surface in S_τ is the sum of the weights of the surfaces in $S_{p(\tau)}$ that got rounded to it, which, by induction, is the number of original surfaces that are recursively rounded to it. In the next step of the recursion the H_j's of the parametrization of the surfaces in S_τ are the functions G_j defined above.

The total weight of the surface in S_τ for a leaf cell τ is the approximate ε-incidences count that we associate with the center of τ.

4.3 Error analysis

We now bound the error incurred by our discretization. We start with the following lemma, whose proof is given in Appendix A of the full version [2].

▶ **Lemma 6.** *Let τ be a cell of the octtree and let $x_j = G_j(\mathbf{x}; \mathbf{t}) + f_j'$, for $j = k+1, \ldots, d$ be a surface obtained in Step 1 of the rounding process described above. For any $\mathbf{x} = (x_1, \ldots, x_k) \in [0, \delta]^k$, for any $\mathbf{t}, \mathbf{s} \in [0, 1]^\ell$, and for each $j = k+1, \ldots, d$, we have*

$$|G_j(\mathbf{x}; \mathbf{s}) - G_j(\mathbf{x}; \mathbf{t})| \le c_2 |\mathbf{x} - \xi| \cdot |\mathbf{t} - \mathbf{s}|, \tag{5}$$

where c_2 is the constant of Condition (ii), and $\xi = (\xi_1, \ldots, \xi_k)$ consists of the first k coordinates of the point in τ of smallest coordinates.

▶ **Lemma 7.** *For any $\mathbf{x} = (x_1, \ldots, x_k) \in [0, \delta]^k$, for any $\mathbf{t}, \mathbf{s} \in [0, 1]^\ell$, and for each $j = k+1, \ldots, d$, we have*

$$\left| G_j(\mathbf{x}; \mathbf{s}) + g_j - (G_j(\mathbf{x}; \mathbf{t}) + f_j') \right| \le c_2 \varepsilon' \le \frac{\varepsilon}{\log(1/\varepsilon)}, \tag{6}$$

where c_2 is the constant of Condition (ii).

Proof. Using the triangle inequality and Lemma 6, we get that

$$\left| G_j(\mathbf{x}; \mathbf{s}) + g_j - (G_j(\mathbf{x}; \mathbf{t}) + f_j') \right|$$

$$\le |G_j(\mathbf{x}; \mathbf{s}) - G_j(\mathbf{x}; \mathbf{t})| + \left| g_j - f_j' \right| \le c_2 |\mathbf{x} - \xi| |\mathbf{t} - \mathbf{s}| + \frac{\varepsilon'}{\ell + 1} .$$

Since $|\mathbf{x} - \xi| \le \delta$, $|\mathbf{t} - \mathbf{s}| \le \frac{\ell \varepsilon'}{(\ell+1)\delta}$, and $|g_j - f_j'| \le \frac{\varepsilon'}{\ell+1}$, the lemma follows. ◀

We now bound the number of surfaces in S_τ. Since $\mathbf{s} \in [0, 1]^\ell$ and each of its coordinates is a multiple of $\frac{\varepsilon'}{(\ell+1)\delta}$, we have at most $(\frac{\delta}{\varepsilon'})^\ell$ different values for $\mathbf{s}$. To bound the number of possible values of $\mathbf{g}$, we prove the following lemma (see [2] for the proof).

▶ **Lemma 8.** *Let $x_j = G_j(\mathbf{x}; \mathbf{t}) + f_j'$, for $j = k+1, \ldots, d$, be a surface $\sigma_{\mathbf{t},\mathbf{f}'}$ in $\tilde{S}_\tau$. For each $j = k+1, \ldots, d$, we have $|f_j'| \le (c_1 + 1)\delta$, where c_1 is the constant of Condition (i).*

Lemma 8 implies that each g_j, $j = k+1, \ldots, d$, has only $O(\frac{\delta}{\varepsilon'})$ possible values, for a total of at most $O((\frac{\delta}{\varepsilon'})^{d-k})$ possible values for $\mathbf{g}$. Combining the number of possible values for $\mathbf{s}$ and $\mathbf{g}$, we get that the number of newly discretized surfaces in S_τ is

$$O\left(\left(\frac{\delta}{\varepsilon'}\right)^\ell \cdot \left(\frac{\delta}{\varepsilon'}\right)^{d-k} \right) = O\left(\left(\frac{\delta}{\varepsilon'}\right)^{\ell+d-k} \right). \tag{7}$$

It follows that each level of the recursive octree decomposition generates

$$O\left(\left(\frac{1}{\delta}\right)^d \cdot \left(\frac{\delta}{\varepsilon'}\right)^{\ell+d-k}\right) = O\left(\frac{\delta^{\ell-k}}{(\varepsilon')^{\ell+d-k}}\right)$$

re-discretized surfaces, where the first factor in the left-hand side expression is the number of cubes generated at this recursive level, and the second factor is the one in (7).

Summing over the recursive levels $j = 0, \ldots, \log\frac{1}{\varepsilon}$, where the cube size δ is $1/2^j$ at level j, we get a total size of $O\left(\frac{1}{(\varepsilon')^{\ell+d-k}} \sum_{j=0}^{\log\frac{1}{\varepsilon}} \frac{1}{2^{j(\ell-k)}}\right)$. We get different estimates for the sum according to the sign of $\ell - k$. If $\ell > k$ the sum is $O(1)$. If $\ell = k$ the sum is $O\left(\log\frac{1}{\varepsilon}\right)$. If $\ell < k$ the sum is $O\left(2^{j_{\max}(k-\ell)}\right) = O\left(\frac{1}{(\varepsilon')^{k-\ell}}\right)$. Accordingly, the overall size of the structure, taking also into account the cost of the first phase, is

$$\begin{cases} O\left(\frac{1}{(\varepsilon')^{\ell+d-k}}\right) & \text{for } \ell > k \\ O\left(\frac{1}{(\varepsilon')^d} \log\frac{1}{\varepsilon}\right) & \text{for } \ell = k \\ O\left(\frac{1}{(\varepsilon')^d}\right) & \text{for } \ell < k. \end{cases} \tag{8}$$

The following theorem summarizes the result of this section. Its proof follows in a straightforward way from the preceding discussion from Lemma 7, analogously to the proof of Lemma 8 in the appendix of the full version [2].

▶ **Theorem 9.** *Let S be a set of n surfaces in $\mathbb{R}^d$ that cross the unit cube $[0,1]^d$, given parametrically as $x_j = F_j(\mathbf{x};\mathbf{t}) + f_j$ for $j = k+1, \ldots, d$, where the functions F_j satisfy conditions (i) and (ii), and $\mathbf{t} = (t_1, \ldots, t_\ell)$. Let G be the (4ε)-grid within $[0,1]^d$. The algorithm described above reports for each vertex v of G an approximate ε-incidences count that includes all surfaces at distance at most ε from v and may include some surfaces at distance at most $(2\sqrt{d}+1)\varepsilon$ from v. The running time of this algorithm is proportional to the total number of rounded surfaces that it generates, which is given by Equation (8), plus an additive $O(n)$ term for the initial canonization of the surfaces.*

We can modify our data structure so that it can answer approximate or exact ε-incidence queries as we describe in Appendix C of [2] for the case of hyperplanes.

4.4   Experimental Results: Summary

We implemented our algorithms and compared their performance on real and synthetic data. Some of our results are shown in Figures 3 and 4. They show that in practical scenarios ($n \in [8K, 32K]$, $\varepsilon \in [0.02, 0.03]$), the primal-dual scheme is considerably faster than the other algorithms. See full details in the full version of the paper [2].
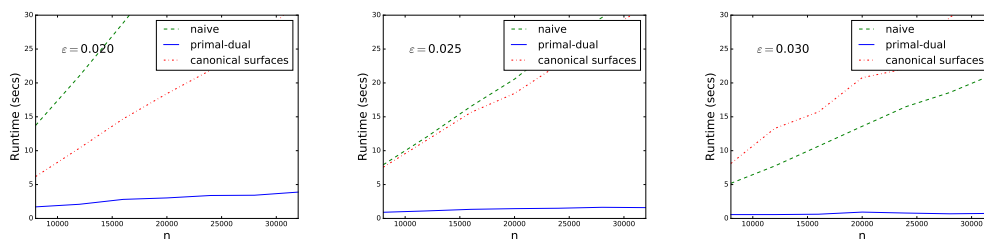
Figure 3 The run-time of the three methods on synthetic data for various values of ε.
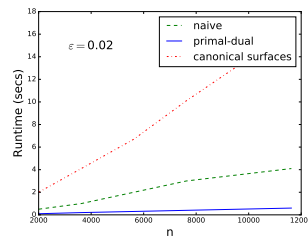
■ **Figure 4** Run-time for real-world data.

──── **References** ────

1   Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building Rome in a day. In *Commun. ACM 54(10)*, pages 105–112. ACM, 2011.

2   D. Aiger, H. Kaplan, E. Kokiopoulou, M. Sharis, and B. Zeisl. General techniques for approximate incidences and their application to the camera posing problem. *CoRR*, 2019. `arXiv:1903.07047`.

3   Dror Aiger, Haim Kaplan, and Micha Sharir. Output Sensitive Algorithms for Approximate Incidences and Their Applications. In *Computational Geometry, to appear. Also in European Symposium on Algorithms*, volume 5, pages 1–13, 2017.

4   Guilherme D Da Fonseca and David M Mount. Approximate range searching: The absolute model. *Computational Geometry*, 43(4):434–444, 2010.

5   Martin A Fischler and Robert C Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.

6   Christian Häne, Lionel Heng, Gim Hee Lee, Friedrich Fraundorfer, Paul Furgale, Torsten Sattler, and Marc Pollefeys. 3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. *Image and Vision Computing*, 68:14–27, 2017.

7   Bert M Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, 1994.

8   Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *ISMAR*, pages 83–86. IEEE, 2009.

9   Sven Middelberg, Torsten Sattler, Ole Untzelmann, and Leif Kobbelt. Scalable 6-DOF Localization on Mobile Devices. In *European Conference on Computer Vision*, pages 268–283. Springer, 2014.

10   Marc Pollefeys, Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, and Reinhard Koch. Visual Modeling with a Hand-Held Camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

11   Johannes L Schonberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016.

12   Chris Sweeney, John Flynn, Benjamin Nuernberger, Matthew Turk, and Tobias Höllerer. Efficient Computation of Absolute Pose for Gravity-Aware Augmented Reality. In *ISMAR*, pages 19–24. IEEE, 2015.

13   Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera Pose Voting for Large-Scale Image-Based Localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2704–2712, 2015.