Efficiently Updating Feasible Regions for Fitting Discrete Polynomial Curve

Fumiki Sekiya[1(✉)] and Akihiro Sugimoto[2]

[1] Department of Informatics, SOKENDAI (The Graduate University for Advanced Studies), Tokyo, Japan
[2] National Institute of Informatics, Tokyo, Japan
{sekiya,sugimoto}@nii.ac.jp

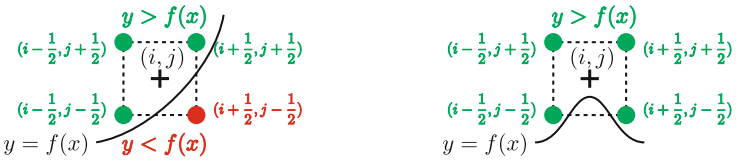
Abstract. We deal with the problem of fitting a discrete polynomial curve to 2D data in the presence of outliers. Finding a maximal inlier set from given data that describes a discrete polynomial curve is equivalent with finding the feasible region corresponding to the set in the parameter space. When iteratively adding a data point to the current inlier set, how to update its feasible region is a crucial issue. This work focuses on how to track vertices of feasible regions in accordance with newly coming inliers. When a new data point is added to the current inlier set, a new vertex is obtained as the intersection point of an edge (or a face) of the feasible region for the current inlier set and a facet (or two facets) of the feasible region for the data point being added. Evaluating all possible combinations of an edge (or a face) and a facet (or two facets) is, however, computationally expensive. We propose an efficient computation in this incremental evaluation that eliminates combinations producing no vertices of the updated feasible region. This computation facilitates collecting the vertices of the updated feasible region. Experimental results demonstrate our proposed computation efficiently reduces practical running time.

1 Introduction

Contour detection is unavoidable for many image processing and/or computer vision tasks such as object recognition, image segmentation and shape approximation. A contour is usually represented as a curve and, thus, contour detection is reduced to fitting a curve to noisy data. Since curves are discretized in the digital image, discrete curve fitting has been studied for decades for different classes of curves and different discretization models [1–6,9,11–13]. An important advantage of using a discrete curve over a continuous one, when used for fitting, is that it requires no empirical threshold in error to define an inlier that affects the output. We note that an underlying threshold that a discrete model uses to collect its points is usually designed only to achieve some properties such as connectivity (see [7,10] for example), and thus such a threshold is clearly justified.

This paper deals with the problem of fitting a discrete polynomial curve to 2D data in the presence of outliers, which is formulated as follows [8]: For a given

W.G. Kropatsch et al. (Eds.): DGCI 2017, LNCS 10502, pp. 254–266, 2017.
DOI: 10.1007/978-3-319-66272-5_21



(a) $(i,j) \in D(\boldsymbol{a})$. (b) $(i,j) \notin D(\boldsymbol{a})$.

Fig. 1. Integer point in (not in) $D(\boldsymbol{a})$. The black curves depict the underlying continuous polynomial curve $y = f(x) = \sum_{l=0}^{d} a_l x^l$.

data set $P = \{(i_p, j_p) \in \mathbb{Z}^2 \mid p = 1, \ldots, n\}$ $(n < \infty)$ and a degree d, the discrete polynomial curve fitting is to find the discrete polynomial curve $D(\boldsymbol{a})$ that has the maximum number of *inliers*, i.e., data points in $D(\boldsymbol{a})$. Here $D(\boldsymbol{a})$ is defined [10] by, with coefficients $\boldsymbol{a} = (a_0, \ldots, a_d) \in \mathbb{R}^{d+1}$,

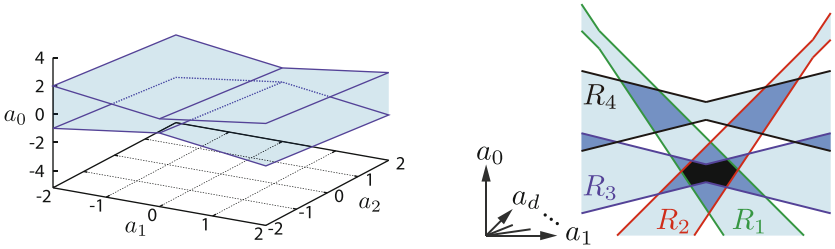
$$D(\boldsymbol{a}) = \left\{ (i,j) \in \mathbb{Z}^2 \;\middle|\; \min_{s \in \{1,\ldots,4\}} \left[(j + y_s) - \sum_{l=0}^{d} a_l (i + x_s)^l \right] \leq 0 \leq \max_{s \in \{1,\ldots,4\}} \left[(j + y_s) - \sum_{l=0}^{d} a_l (i + x_s)^l \right] \right\}, \tag{1}$$

where $(x_1, y_1) = (-\frac{1}{2}, -\frac{1}{2})$, $(x_2, y_2) = (\frac{1}{2}, -\frac{1}{2})$, $(x_3, y_3) = (\frac{1}{2}, \frac{1}{2})$, $(x_4, y_4) = (-\frac{1}{2}, \frac{1}{2})$. Equation (1) collects $(i,j) \in \mathbb{Z}^2$ iff the four points $(i + x_s, j + y_s)$ for $s \in \{1, \ldots, 4\}$ lie on the both sides ($y > \sum_{l=0}^{d} a_l x^l$ and $y < \sum_{l=0}^{d} a_l x^l$) of the underlying continuous polynomial curve $y = \sum_{l=0}^{d} a_l x^l$, or at least one of them is on $y = \sum_{l=0}^{d} a_l x^l$ (Fig. 1). We employ this discretization model because the connectivity is guaranteed [7].

This problem can be discussed in the parameter (coefficient) space. For the data point (i_p, j_p) $(p \in \{1, \ldots, n\})$, the *feasible region* $R_p \subseteq \mathbb{R}^{d+1}$ is defined by

$$R_p = \left\{ \boldsymbol{a} \in \mathbb{R}^{d+1} \;\middle|\; \min_{s \in \{1,\ldots,4\}} h_{(p,s)}(\boldsymbol{a}) \leq 0 \leq \max_{s \in \{1,\ldots,4\}} h_{(p,s)}(\boldsymbol{a}) \right\},$$

where $h_{(p,s)}(\boldsymbol{a}) = (j_p + y_s) - \sum_{l=0}^{d} (i_p + x_s)^l a_l$ (Fig. 2(a)). We remark that $(i_p, j_p) \in D(\boldsymbol{a})$ iff $\boldsymbol{a} \in R_p$. R_p is an unbounded concave polytope, which is the union of two unbounded convex polytopes defined by $h_{(p,1)(\boldsymbol{a})} \leq 0 \leq h_{(p,3)}(\boldsymbol{a})$ and $h_{(p,2)(\boldsymbol{a})} \leq 0 \leq h_{(p,4)}(\boldsymbol{a})$ (cf. Fig. 3(a)). For $\mathit{\Pi} \subseteq \{1, \ldots, n\}$, the *feasible region* $R_{\mathit{\Pi}}$ is defined as the intersection of the feasible regions each of which is for a data point (i_p, j_p) where $p \in \mathit{\Pi}$: $R_{\mathit{\Pi}} = \bigcap_{p \in \mathit{\Pi}} R_p$ (Fig. 2(b)). Since $R_{\mathit{\Pi}} = \emptyset$ if there exists no $\boldsymbol{a} \in \mathbb{R}^{d+1}$ that satisfies $\{(i_p, j_p) \mid p \in \mathit{\Pi}\} \subseteq D(\boldsymbol{a})$, we may assume $R_{\mathit{\Pi}} \neq \emptyset$ below. Accordingly, our fitting problem is formulated in the parameter



(a) R_p. (b) Parameter space.

Fig. 2. Feasible region. (a) shows R_p for $d = 2$ and $(i_p, j_p) = (0, 0)$. (b) shows intersections among the feasible regions for four data points indexed from 1 to 4. A darker region has a larger number of inliers.

space as follows: Given P and d, find $\Pi \subseteq \{1, \ldots, n\}$ with the maximum $|\Pi|$ and $\boldsymbol{a} \in R_\Pi$ for that Π.

This problem requires evaluating R_Π for all $\Pi \subset \{1, \ldots, n\}$, which is reduced to classify each data point into an inlier or an outlier (we have 2^n instances). To this end, a heuristic based incremental approach [8] was proposed where it iteratively evaluates whether a data point can be added to the current inlier set until the inlier set does not have its superset. In this approach, the feasible region for the current inlier set is tracked by its vertices: when a new data point is added to the current inlier set, a vertex of the new feasible region can be obtained from the intersection points of an edge (or a face) of the current feasible region and a facet (or two facets) of the feasible region for the data point being added. Evaluating such possible combinations all is, however, computationally expensive. The contribution of this paper is to facilitate this incremental evaluation by introducing an efficient computation of the vertices of the new feasible region. Our introduced computation eliminates combinations producing no vertex of the new feasible region, based on the property that an edge or face of a bounded feasible region is inside the convex hull of its vertices. Though the computational complexity is not reduced, our introduced computation efficiently reduces running time in practice, as shown in experiments.

2 Brief Review of Incremental Approach

The incremental approach [8] starts with computing the feasible region for an initialized inlier set. It then evaluates each data point one by one to update the feasible region. If the updated feasible region is not empty, the data point is added to the inlier set; it is regarded as an outlier otherwise. How to represent and update the feasible region is a key issue there, which is briefly explained below.

2.1 Representing a Feasible Region Using Its Vertices

A vertex of a feasible region is defined as an intersection point of its facets. For $p = 1, \ldots, n$, and $s = 1, \ldots, 4$, a *facet* $F(p,s)$ *of* R_p is defined by

$$F(p,s) = \left\{ \boldsymbol{a} \in \mathbb{R}^{d+1} \;\middle|\; \begin{array}{l} h_{(p,s)}(\boldsymbol{a}) = 0 \text{ and} \\ s \in \underset{s' \in \{1,\ldots,4\}}{\arg\min}\; h_{(p,s')}(\boldsymbol{a}) \cup \underset{s' \in \{1,\ldots,4\}}{\arg\max}\; h_{(p,s')}(\boldsymbol{a}) \end{array} \right\}.$$

$F(p,s)$ is a part of the hyperplane $h_{(p,s)}(\boldsymbol{a}) = 0$ supporting R_p (Fig. 3(a)). Similarly, a *facet* $F_{\Pi}(p,s)$ *of* R_{Π} is defined by $F_{\Pi}(p,s) = F(p,s) \cap R_{\Pi}$ for $\Pi \subseteq \{1, \ldots, n\}$ and $(p,s) \in \Pi \times \{1, \ldots, 4\}$ (Fig. 3(b)). Note that $F_{\Pi}(p,s)$ may be empty for some (p,s).

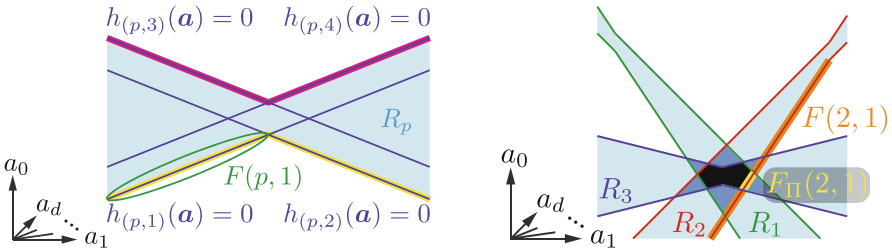
A vertex of R_{Π} is given as the intersection of $d+1$ facets. The set V_{Π} of the vertices of R_{Π} is defined by

$$V_{\Pi} = \left\{ \boldsymbol{a} \in \mathbb{R}^{d+1} \;\middle|\; \begin{array}{l} \boldsymbol{a} \in \bigcap_{\lambda=1}^{d+1} F_{\Pi}(p_\lambda, s_\lambda) \text{ for some} \\ (p_1,s_1), \ldots, (p_{d+1}, s_{d+1}) \subseteq \Pi \times \{1,\ldots,4\} \\ \text{such that } h_{(p_\lambda,s_\lambda)}(\boldsymbol{a}) = 0 \text{ for } \lambda = 1,\ldots,d+1 \\ \text{are linearly independent} \end{array} \right\}. \tag{2}$$

See Fig. 4 for an illustration of V_{Π}. Each combination of $d+1$ facets determining an element in V_{Π} is indicated by an element in Ψ_{Π}, which is defined by

$$\Psi_{\Pi} = \left\{ \begin{array}{r} \{(p_1,s_1), \ldots, (p_{d+1}, s_{d+1})\} \\ \subseteq \Pi \times \{1,\ldots,4\} \end{array} \;\middle|\; \begin{array}{l} h_{(p_\lambda,s_\lambda)}(\boldsymbol{a}) = 0 \text{ for } \lambda = 1,\ldots,d+1 \\ \text{are linearly independent and} \\ \text{their solution is in } \bigcap_{\lambda=1}^{d+1} F_{\Pi}(p_\lambda, s_\lambda) \end{array} \right\}. \tag{3}$$

In this way, the feasible region of the inlier set Π can be represented by its vertices V_{Π} with the help of Ψ_{Π}. Note that different elements in Ψ_{Π} may determine the same element of V_{Π}.



(a) $F(p,s)$. (b) $F_{\Pi}(p,s)$ ($\Pi = \{1,2,3\}$).

Fig. 3. Facets of a feasible region. (a): $h_{(p,s)}(\boldsymbol{a}) = 0$, $s = 1, ...4$ are depicted in blue lines. $\min_{s \in \{1,\ldots,4\}} h_{(p,s)}(\boldsymbol{a}) = 0$ is depicted in yellow, while $\max_{s \in \{1,\ldots,4\}} h_{(p,s)}(\boldsymbol{a}) = 0$ is depicted in pink. (Color figure online)

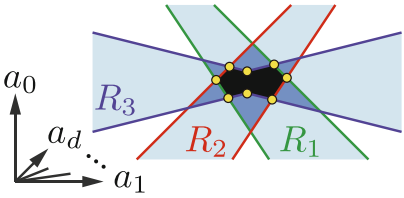


Fig. 4. Vertices V_{Π} (yellow) of R_{Π} ($\Pi = \{1, 2, 3\}$). (Color figure online)

2.2 Tracking the Vertices of the Feasible Region

Theorem 1 indicates that Ψ_{Π} plays an important role in tracking the vertices of the updated feasible region when a new inlier (represented by p^*) comes in. Note that R_{Π} is almost always bounded (see [8]).

Theorem 1 (Sekiya and Sugimoto[8]). *For $\Pi \subsetneq \{1, \ldots, n\}$ such that R_{Π} is bounded and $p^* \in \{1, \ldots, n\} \backslash \Pi$, $\Psi_{\Pi \cup \{p^*\}} \subseteq \Psi_{\Pi} \cup \Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}$, where*

$$\Phi^1_{\Pi,p^*} = \left\{ \begin{array}{r} \{(p_1, s_1), \ldots, (p_d, s_d)\} \\ \cup \{(p^*, s^*)\} \end{array} \middle| \begin{array}{l} \{(p_1, s_1), \ldots, (p_d, s_d)\} \text{ is a subset of} \\ \text{an element in } \Psi_{\Pi} \text{ and } s^* = 1, \ldots, 4 \end{array} \right\},$$

$$\Phi^2_{\Pi,p^*} = \left\{ \begin{array}{r} \{(p_1, s_1), \ldots, (p_{d-1}, s_{d-1})\} \\ \cup \{(p^*, s_1^*), (p^*, s_2^*)\} \end{array} \middle| \begin{array}{l} \{(p_1, s_1), \ldots, (p_{d-1}, s_{d-1})\} \text{ is} \\ \text{a subset of an element in } \Psi_{\Pi} \\ \text{and } (s_1^*, s_2^*) \in \{(1, 2), (3, 4)\} \end{array} \right\}.$$

$\{(p_1, s_1), \ldots, (p_{\omega}, s_{\omega})\}$ where $\omega = d$ and $d - 1$ respectively corresponds to an *edge* and a (2-dimensional) *face* of R_{Π}: $\bigcap_{\lambda=1}^{\omega} F_{\Pi}(p_{\lambda}, s_{\lambda})$ (the intersection of ω facets). An element in Φ^1_{Π,p^*} (resp. Φ^2_{Π,p^*}) is thus considered to be the combination of an edge (resp. a face) of R_{Π} and a facet (resp. two facets) of R_{p^*}. Theorem 1 therefore indicates that a vertex of $R_{\Pi \cup \{p^*\}}$ is a vertex of R_{Π} or otherwise obtained as the intersection point of an edge (resp. a 2-dimensional face) of R_{Π} and a facet (resp. two facets) of R_{p^*}.

3 Efficient Update of Vertices of Feasible Region

Sekiya and Sugimoto [8] evaluates whether each element in $\Psi_{\Pi} \cup \Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}$ satisfies the condition in Eq. (3) to extract elements in $\Psi_{\Pi \cup \{p^*\}}$. When R_{Π} is bounded, any edge or face of R_{Π} is inside the convex hull of the vertices of R_{Π} on that edge or face (Lemma 1). Based on this, we introduce a computation to eliminate elements in $\Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}$ that cannot be in $\Psi_{\Pi \cup \{p^*\}}$. This enables us to compute $\Psi_{\Pi \cup \{p^*\}}$ efficiently.

We define a set of edges (or faces) of R_{Π}. Namely, for $\omega = d$ (edge), $d - 1$ (face), we define

$$\Psi^{(\omega)}_{\Pi} = \left\{ \begin{array}{r} \{(p_1, s_1), \ldots, (p_{\omega}, s_{\omega})\} \\ \subseteq \Pi \times \{1, \ldots, 4\} \end{array} \middle| \ \{(p_1, s_1), \ldots, (p_{\omega}, s_{\omega})\} \subseteq \xi \text{ such that } \xi \in \Psi_{\Pi} \right\}.$$

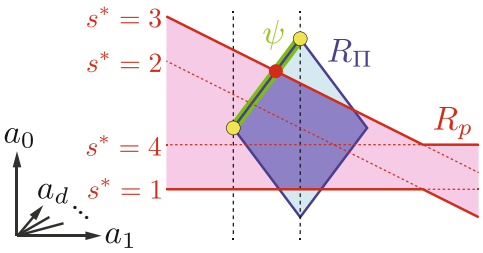


Fig. 5. Illustration of $A_{\Pi,p^*}(\psi)$. For $\psi \in \Psi_\Pi^{(\omega)}$, $\bigcap_{(p,s)\in\psi} F_\Pi(p,s)$ is depicted in green and $V_\Pi(\psi)$ in yellow. For $s^* = 1,\ldots,4$, $h_{(p^*,s^*)}(\boldsymbol{a}) = 0$ is depicted in a solid and dotted red line where the solid part depicts $F(p^*,s^*)$. In this example, s^* satisfies (i) in Eq. (4) if $h_{(p^*,s^*)}(\boldsymbol{a}) = 0$ runs between the two yellow vertices, and (ii) if $h_{(p^*,s^*)}(\boldsymbol{a}) = 0$ is depicted in a solid line on either of the dotted black lines which are parallel to the a_0 axis and passes through yellow vertices. $A_{\Pi,p^*}(\psi) = \{3\}$, accordingly. (Color figure online)

We note that an edge (or face) is determined as the intersection of d (or $d-1$) facets of R_Π. For an edge (or face) $\psi \in \Psi_\Pi^{(\omega)}$ $(\omega = d, d-1)$, we denote by $V_\Pi(\psi)$ the vertices on ψ:

$$V_\Pi(\psi) = \{\boldsymbol{a} \in V_\Pi \mid \boldsymbol{a} \text{ is determined by } \xi \in \Psi_\Pi \text{ such that } \psi \subseteq \xi\}.$$

Let $\mathrm{Conv}(V_\Pi(\psi))$ denote the convex hull of $V_\Pi(\psi)$. Then we have Lemma 1 whose proof is provided in Appendix A.

Lemma 1. *For $\Pi \subseteq \{1,\ldots,n\}$ for which R_Π is bounded and $\psi \in \Psi_\Pi^{(\omega)}$ $(\omega = d, d-1)$, $\bigcap_{(p,s)\in\psi} F_\Pi(p,s) \subseteq \mathrm{Conv}(V_\Pi(\psi))$.*

Suppose we are adding an new inlier p^* to the current inlier set Π. For each $\psi \in \Psi_\Pi^{(d)}$, Φ^1_{Π,p^*} has four elements $\psi \cup \{(p^*,s^*)\}$, $s^* = 1,\ldots,4$. Lemma 1 allows to identify $\tilde{s}^* \in \{1,\ldots,4\}$ such that $\bigcap_{(p,s)\in\psi} F_\Pi(p,s) \cap F(p^*,\tilde{s}^*) = \emptyset$ (the facet of R_{p^*} corresponding to $\tilde{s}^*$ does not intersect with the edge ψ). We can thus define the subset of Φ^1_{Π,p^*} by eliminating $\psi \cup \{(p^*,\tilde{s}^*)\}$ and use the subset instead of Φ^1_{Π,p^*} itself in computing $\Psi_{\Pi\cup\{p^*\}}$. The same argument can be applied to Φ^2_{Π,p^*}.

To exclude $\tilde{s}^*$ above, we define $A_{\Pi,p^*}(\psi)$ for $\psi \in \Psi_\Pi^{(\omega)}$ $(\omega = d, d-1)$, by

$$A_{\Pi,p^*}(\psi) = \left\{ s^* \in \{1,\ldots,4\} \;\middle|\; \begin{array}{l} \text{(i)} \ \min\limits_{\boldsymbol{a}\in V_\Pi(\psi)} h_{(p^*,s^*)}(\boldsymbol{a}) \le 0 \le \max\limits_{\boldsymbol{a}\in V_\Pi(\psi)} h_{(p^*,s^*)}(\boldsymbol{a}) \ \text{and} \\ \text{(ii)} \ s^* \in \operatorname{arg\,min}\limits_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a}) \cup \operatorname{arg\,max}\limits_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a}) \\ \quad \text{for some } \boldsymbol{a} \in V_\Pi(\psi) \end{array} \right\} \quad (4)$$

With $A_{\Pi,p^*}(\psi)$, we can identify the facets of R_{p^*} that potentially intersect with the edge/face ψ (see Fig. 5). (i) in Eq. (4) means that the hyperplane

$h_{(p^*,s^*)}(\boldsymbol{a}) = 0$ runs between two vertices of ψ, or passes through one of its vertices. Since the edge or face is inside the convex hull of its vertices (Lemma 1), it follows that the hyperplane intersects with ψ. This however does not indicate that the facet determined by s^* intersects with ψ (the facet is only a part of the hyperplane). We therefore have to evaluate whether the facet indeed intersects with ψ, for which (ii) plays the role. (ii) means that, for at least one vertex $\boldsymbol{a} \in V_{\Pi}(\psi)$, s^* achieves the maximum or minimum of $h_{(p^*,s')}(\boldsymbol{a}), s' \in \{1,\ldots,4\}$; unless (ii) is satisfied, the intersection of the hyperplane with ψ is out of the facet.

Using only $s^* \in \{1,\ldots,4\}$ in $A_{\Pi,p^*}(\psi)$ for each ψ, we can define the subsets of Φ^1_{Π,p^*} and Φ^2_{Π,p^*} that can be used in computing $\Psi_{\Pi\cup\{p^*\}}$.

$$X^1_{\Pi,p^*} = \left\{ \psi \cup \{(p^*,s^*)\} \;\middle|\; \psi \in \Psi^{(d)}_{\Pi} \text{ and } s^* \in A_{\Pi,p^*}(\psi) \right\},$$

$$X^2_{\Pi,p^*} = \left\{ \psi \cup \{(p^*,s^*_1),(p^*,s^*_2)\} \;\middle|\; \begin{array}{l} \psi \in \Psi^{(d-1)}_{\Pi} \text{ and } s^*_1, s^*_2 \in A_{\Pi,p^*}(\psi) \\ \text{where } (s^*_1,s^*_2) = (1,2) \text{ or } (3,4) \end{array} \right\}.$$

Now we formally prove that no element in $\Phi^1_{\Pi,p^*}\backslash X^1_{\Pi,p^*}$ or $\Phi^2_{\Pi,p^*}\backslash X^2_{\Pi,p^*}$ is in $\Psi_{\Pi\cup\{p^*\}}$.

Theorem 2. *For $\Pi \subsetneq \{1,\ldots,n\}$ such that R_Π is bounded and $p^* \in \{1,\ldots,n\}\backslash\Pi$, $\Psi_{\Pi\cup\{p^*\}} \subseteq \Psi_\Pi \cup X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}$.*

Proof. Consider $\psi \in \Psi^{(\omega)}_{\Pi}$ $(\omega = d, d-1)$. We show $\bigcap_{(p,s)\in\psi} F_\Pi(p,s) \cap F(p^*,s^*) = \emptyset$ for any $s^* \in \{1,\ldots,4\}\backslash A_{\Pi,p^*}(\psi)$. Note that this means $\psi' \notin \Psi_{\Pi\cup\{p^*\}}$ for any $\psi' \in \Phi^{d+1-\omega}_{\Pi,p^*}\backslash X^{d+1-\omega}_{\Pi,p^*}$.

We first assume that s^* does not satisfy (i) in Eq. (4): $h_{(p^*,s^*)}(\boldsymbol{a}) < 0$ for $\forall \boldsymbol{a} \in V_\Pi(\psi)$ or $h_{(p^*,s^*)}(\boldsymbol{a}) > 0$ for $\forall \boldsymbol{a} \in V_\Pi(\psi)$. From Lemma 1 and the linearity of $h_{(p^*,s^*)}$, we have $h_{(p^*,s^*)}(\boldsymbol{a}) < 0$ for $\forall \boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F_\Pi(p,s)$ or $h_{(p^*,s^*)}(\boldsymbol{a}) > 0$ for $\forall \boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F_\Pi(p,s)$. It follows that $\bigcap_{(p,s)\in\psi} F_\Pi(p,s) \cap F(p^*,s^*) = \emptyset$.

We next assume that s^* does not satisfy (ii): $s^* \notin \arg\min_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a}) \cup \arg\max_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a})$ for $\forall \boldsymbol{a} \in V_\Pi(\psi)$. Lemma 1 and the linearity of $h_{(p^*,s^*)}$ imply $s^* \notin \arg\min_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a}) \cup \arg\max_{s'\in\{1,\ldots,4\}} h_{(p^*,s')}(\boldsymbol{a})$ for $\forall \boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F_\Pi(p,s)$. It follows that $\bigcap_{(p,s)\in\psi} F_\Pi(p,s) \cap F(p^*,s^*) = \emptyset$. □

4 Algorithm

Algorithm 1 [8][1] is the incremental approach to solve the discrete polynomial curve fitting for a given data point set P and a given degree d. It classifies each data index into two classes: Π (inlier) and $\Pi^{\complement}$ (outlier). Π is first initialized to be a set of $d+1$ data indices for which V_Π and Ψ_Π are computed at low cost

[1] Because the initial inlier selection is not the scope of this paper, Algorithm 1 is presented without any initial inlier set.

Algorithm 1. Incremental algorithm (Sekiya+ [8]).

Require: P, d.
Ensure: $\mathit{\Pi} \subseteq \{1,\ldots,n\}$ and $V_{\mathit{\Pi}}$.
1: Initialize $\mathit{\Pi}$:= any $d+1$ data indices in $\{1,\ldots,n\}$ for which $R_{\mathit{\Pi}}$ is bounded.
2: Initialize $\mathit{\Pi}^{\complement} := \emptyset$.
3: Compute $V_{\mathit{\Pi}}$ and $\Psi_{\mathit{\Pi}}$ using Eqs. (2) and (3).
4: **while** $\{1,\ldots,n\} \setminus (\mathit{\Pi} \cup \mathit{\Pi}^{\complement}) \neq \emptyset$ **do**
5: p^* := any data index in $\{1,\ldots,n\} \setminus (\mathit{\Pi} \cup \mathit{\Pi}^{\complement})$.
6: Compute $V_{\mathit{\Pi}\cup\{p^*\}}$ and $\Psi_{\mathit{\Pi}\cup\{p^*\}}$
7: **if** $V_{\mathit{\Pi}\cup\{p^*\}} \neq \emptyset$ **then**
8: $\mathit{\Pi} := \mathit{\Pi} \cup \{p^*\}$ and update $V_{\mathit{\Pi}}$ and $\Psi_{\mathit{\Pi}}$.
9: **else**
10: $\mathit{\Pi}^{\complement} := \mathit{\Pi}^{\complement} \cup \{p^*\}$.
11: **end if**
12: **end while**
13: **return** $\mathit{\Pi}$ and $V_{\mathit{\Pi}}$.

(see [8] for the sufficient condition that $R_{\mathit{\Pi}}$ is bounded). In the following loop (Steps 4–12), we add new data indices to either $\mathit{\Pi}$ or $\mathit{\Pi}^{\complement}$ one by one. When $\mathit{\Pi}$ is updated, $V_{\mathit{\Pi}}$ and $\Psi_{\mathit{\Pi}}$ are also updated. Since $\Phi_{\mathit{\Pi}\cup\{p^*\}} \neq \emptyset$ if $R_{\mathit{\Pi}\cup\{p^*\}} \neq \emptyset$ (see [8]), an inlier set obtained by Algorithm 1 is guaranteed to have no superset.

The purpose of Algorithm 2 is to efficiently compute $V_{\mathit{\Pi}\cup\{p^*\}}$ and $\Psi_{\mathit{\Pi}\cup\{p^*\}}$ in Step 6 of Algorithm 1: efficient computation of the vertices of the feasible region updated by an additional data point. Some of the vertices are inherited from the current feasible region. So, the first loop (Steps 2–7) evaluates each vertex of the current feasible region to check if it serves as a vertex of the updated feasible region, where it suffices only to verify that the vertex is inside the feasible region for the additional data point (Step 5), since $F_{\mathit{\Pi}}(p,s) \cap R_{p^*} = F_{\mathit{\Pi}\cup\{p^*\}}(p,s)$ for any $(p,s) \in \mathit{\Pi} \times \{1,\ldots,4\}$. The vertices appearing only in the updated feasible region are obtained from $X^1_{\mathit{\Pi},p^*} \cup X^2_{\mathit{\Pi},p^*}$ in the second loop (Steps 9–16). For each element in $X^1_{\mathit{\Pi},p^*} \cup X^2_{\mathit{\Pi},p^*}$, we first check if the hyperplanes corresponding to the element intersect at a unique point (Step 10), and if so, we then check if that intersection point serves as a vertex of the updated feasible region (Step 12). Note that the condition in Step 12 is equivalent with $\boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F_{\mathit{\Pi}\cup\{p^*\}}(p,s)$; $\boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F(p,s)$ implies $\boldsymbol{a} \in R_{p^*}$ since any $\psi \in X^1_{\mathit{\Pi},p^*} \cup X^2_{\mathit{\Pi},p^*}$ contains (p,s) such that $p = p^*$. The most efficiently working part is Step 8, which reduces the number of iterations in the second loop.

The computational complexity for this method is the same with [8]: $\mathcal{O}(n^{d+2})$ for a variable number n of data and a fixed degree d, because $\mathcal{O}(|X^{\alpha}_{\mathit{\Pi},p^*}|) = \mathcal{O}(|\Phi^{\alpha}_{\mathit{\Pi},p^*}|) = \mathcal{O}(|\Psi^{(\omega)}_{\mathit{\Pi}}|)$ for $\alpha = 1, 2$ where $\omega = d+1-\alpha$. The practical efficiency of the method is evaluated in the next section.

Algorithm 2. Efficient update of $V_{\mathit{\Pi}}$ and $\Psi_{\mathit{\Pi}}$ for an additional inlier.

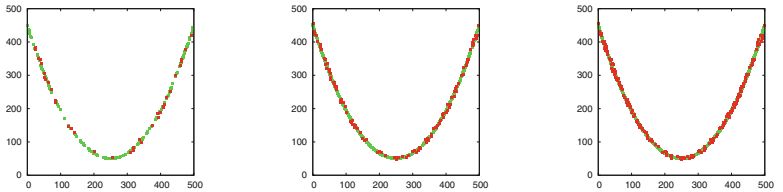
Require: P, d, $\mathit{\Pi} \subsetneq \{1, \ldots, n\}$, $p^* \in \{1, \ldots, n\} \setminus \mathit{\Pi}$, $V_{\mathit{\Pi}}$ and $\Psi_{\mathit{\Pi}}$.
Ensure: $V_{\mathit{\Pi} \cup \{p^*\}}$ and $\Psi_{\mathit{\Pi} \cup \{p^*\}}$.
1: Initialize $V := \emptyset$ and $\Psi := \emptyset$.
2: **for all** $\psi \in \Psi_{\mathit{\Pi}}$ **do**
3: $\boldsymbol{a} :=$ vertex determined by ψ.
4: **if** $\boldsymbol{a} \in R_{p^*}$ **then**
5: $V := V \cup \{\boldsymbol{a}\}$ and $\Psi := \Psi \cup \{\psi\}$.
6: **end if**
7: **end for**
8: Compute $A_{\mathit{\Pi},p^*}(\psi)$ for all $\psi \in \Psi_{\mathit{\Pi}}^{(\omega)}$ $(\omega = d, d-1)$ to have $X^1_{\mathit{\Pi},p^*}$ and $X^2_{\mathit{\Pi},p^*}$.
9: **for all** $\psi \in X^1_{\mathit{\Pi},p^*} \cup X^2_{\mathit{\Pi},p^*}$ **do**
10: **if** $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p,s) \in \psi\}$ has a unique solution $\boldsymbol{a}$ **then**
11: $\boldsymbol{a} :=$ the unique solution to $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p,s) \in \psi\}$
12: **if** $\boldsymbol{a} \in \bigcap_{(p,s)\in\psi} F(p,s) \cap R_{\mathit{\Pi}}$ **then**
13: $V := V \cup \{\boldsymbol{a}\}$ and $\Psi := \Psi \cup \{\psi\}$.
14: **end if**
15: **end if**
16: **end for**
17: **return** $V = V_{\mathit{\Pi} \cup \{p^*\}}$ and $\Psi = \Psi_{\mathit{\Pi} \cup \{p^*\}}$.

5 Experiments

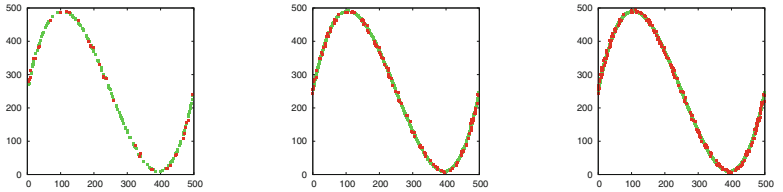
For $d = 2$, we generated input data sets P for $n = 200, 400, 600, 800, 1000$ as follows: setting $(a_0, a_1, a_2) = (450, -3.2, 0.0064)$, we randomly generated n integer points within $[0, 499] \times [0, 499]$ so that 80% of the points, called *ground-truth inliers*, are in $D(a_0, a_1, a_2)$ while the other 20% points, called *ground-truth outliers*, are not in $D(a_0, a_1, a_2)$, where each ground-truth outlier was generated so that its Euclidean distance from its closest point in $D(a_0, a_1, a_2)$ is in $[1, 4]$. In the same way, we generated data sets for $d = 3$ where we used $(a_0, a_1, a_2, a_3) = (250, 5, -0.03, 4.0 \times 10^{-5})$ to generate their ground-truth inliers and outliers. P is shown in Fig. 6 for $n = 200, 600, 1000$. In the experiments, we used a PC with an Intel Xeon 3.7 GHz processor with 64 GB memory.

We applied our proposed method (Algorithm 1 together with Algorithm 2) 100 times to each P to see the efficiency of our introduced computation. At each trial, we randomly initialized $\mathit{\Pi}$ (Step 1) and selected p^* (Step 5), where the $d + 1$ data indices in the initial $\mathit{\Pi}$ are chosen only from the ground-truth inliers. For comparison, we also applied Algorithm 1 alone (Sekiya+ [8]) 100 times to each P using the same initialization and data point selection. We then evaluated the recall (the ratio of ground-truth inliers in the output against the whole ground-truth inliers) and the computational time (i.e., processing time).

Tables 1 and 2 show the average of recalls over 100 trials for each P and the average of computational times over 100 trials for the two methods. We remark that the outputs by our proposed method are exactly the same as those by Algorithm 1 alone.



(a) $d = 2, n = 200$. (b) $d = 2, n = 600$. (c) $d = 2, n = 1000$.



(d) $d = 3, n = 200$. (e) $d = 3, n = 600$. (f) $d = 3, n = 1000$.

Fig. 6. Examples of input data sets. Ground-truth inliers are depicted in green while ground-truth outliers in red. (Color figure online)

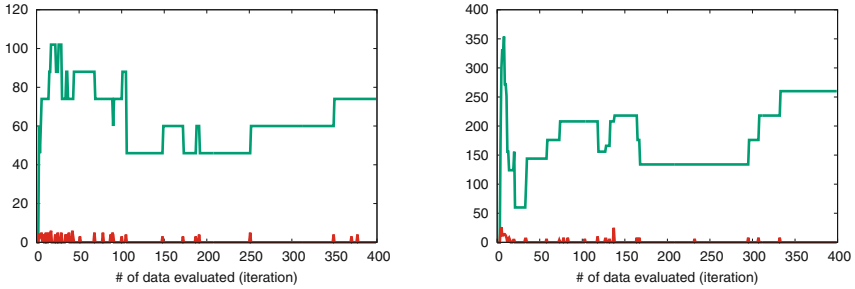
We see in Table 2 that our proposed method achieves the same results much faster than Algorithm 1 alone. We see that Algorithm 2 indeed efficiently updates feasible regions in computational time thanks to $|X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}| < |\Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}|$. To visualize this efficiency, we compared $|X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}|$ and $|\Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}|$ in each iteration in a trial in the case of $n = 400$ (Fig. 7). We observe that (1) $|X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}|$ is significantly smaller than $|\Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}|$ and that (2) $|X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}|$ is almost constant independent of the size of the inlier set. We remark that $X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*} = \emptyset$ indicates $A_{\Pi,p^*}(\psi) = \emptyset$ for all $\psi \in \Psi^{(\omega)}_{\Pi}$ $(\omega = d, d-1)$.

Table 1. Recall of ground-truth inliers (average over 100 trials).

n	200	400	600	800	1000
$d = 2$	0.870	0.860	0.810	0.859	0.855
$d = 3$	0.817	0.805	0.773	0.806	0.822

Table 2. Computational time (ms) (average over 100 trials).

	n	200	400	600	800	1000
$d = 2$	Sekiya+[8]	36.92	69.16	90.92	138.32	177.28
	Proposed	1.16	1.88	2.32	3.28	3.60
$d = 3$	Sekiya+[8]	154.16	267.32	366.16	488.36	627.16
	Proposed	8.68	9.12	10.36	9.76	11.12



(a) $d = 2$. (b) $d = 3$.

Fig. 7. $|X^1_{\Pi,p^*} \cup X^2_{\Pi,p^*}|$ (red) and $|\Phi^1_{\Pi,p^*} \cup \Phi^2_{\Pi,p^*}|$ (green) in each iteration. The horizontal axis is the number of data points already evaluated (i.e., $|\Pi \cup \Pi^{\complement}|$). The results are from a trial of $n = 400$. (Color figure online)

6 Conclusion

We dealt with the problem of fitting a discrete polynomial curve to 2D data in the presence of outliers. We discussed how to efficiently compute the vertices of the feasible region for an incrementally updated inlier set in the parameter space. Based on the property that an edge or face of a bounded feasible region is inside the convex hull of its vertices, we introduced a computation to facilitate updating the vertices of the feasible region when a new data point is added to the current inlier set. The efficiency of our proposed computation was demonstrated by our experimental results.

Acknowledgements. This work is in part supported by Grant-in-Aid for Scientific Research (Grant No. 16H02851) of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

A Appendix: Proof of Lemma 1

Proof. For any $\boldsymbol{b} \in \bigcap_{(p,s)\in\psi} F_\Pi(p,s)$, we prove that $\boldsymbol{b} \in \mathrm{Conv}(V_\Pi(\psi))$, i.e., $\boldsymbol{b}$ is represented as a convex combination of some vertices in $V_\Pi(\psi)$. In fact, it suffices to show that there exist $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \bigcap_{(p,s)\in\psi} F_\Pi(p,s)$ and $(p_1,s_1),(p_2,s_2) \in \Pi \times \{1,\dots,4\}$ that satisfy the following: $\boldsymbol{c}_1 \in F_\Pi(p_1,s_1)$ and $\boldsymbol{c}_2 \in F_\Pi(p_2,s_2)$, the linear systems $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p,s) \in \psi \cup \{(p_1,s_1)\}\}$ and $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p,s) \in \psi \cup \{(p_2,s_2)\}\}$ are respectively independent, and $\boldsymbol{b}$ is represented as a convex combination of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. Why this proves $\boldsymbol{b} \in \mathrm{Conv}(V_\Pi(\psi))$ is explained as follows. For $\psi \in \Psi^{(d)}_\Pi$, it is obvious since we have $\boldsymbol{c}_1, \boldsymbol{c}_2 \in V_\Pi(\psi)$ immediately. For $\psi \in \Psi^{(d-1)}_\Pi$, next, each of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ can be seen as $\boldsymbol{b}$ in the case of $\psi \in \Psi^{(d)}_\Pi$; $\psi \cup \{(p_1,s_1)\}, \psi \cup \{(p_2,s_2)\} \in \Psi^{(d)}_\Pi$ is proven by Lemma 1 in [8]. From the result already obtained for $\psi \in \Psi^{(d)}_\Pi$, therefore, $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are respectively represented as a convex combination of two vertices in $V_\Pi(\psi \cup \{(p_1,s_1)\})$ and

$V_{\Pi}(\psi \cup \{(p_2, s_2)\})$. $\boldsymbol{b}$ is represented as a convex combination of the four vertices in $V_{\Pi}(\psi)$, accordingly.

We therefore prove for the existence of $\boldsymbol{c}_1, \boldsymbol{c}_2, (p_1, s_1), (p_2, s_2)$ as described above. If there exists $(p', s') \in (\Pi \times \{1, \ldots, 4\}) \backslash \psi$ such that $\boldsymbol{b} \in F_{\Pi}(p', s')$ and $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p, s) \in \psi \cup \{(p', s')\}\}$ is independent, then the required condition is immediately satisfied for $\boldsymbol{c}_1 = \boldsymbol{c}_2 = \boldsymbol{b}$ and $(p_1, s_1) = (p_2, s_2) = (p', s')$. We therefore give the proof for the other case. Let us first consider the $(d + 1 - \omega)$-dimensional flat $\{\boldsymbol{a} \in \mathbb{R}^{d+1} \mid h_{(p,s)}(\boldsymbol{a}) = 0 \text{ for } (p, s) \in \psi\}$, which includes $\bigcap_{(p,s)\in\psi} F_{\Pi}(p, s)$. We then consider an arbitrary half-line on the flat running from $\boldsymbol{b}$. Note that such a half-line necessarily exists since $d + 1 - \omega \geq 1$. A point in the half-line is represented by $\boldsymbol{c}(r) = \boldsymbol{b} + r\boldsymbol{v}$ for some vector $\boldsymbol{v} \in \mathbb{R}^{d+1} \backslash \{\boldsymbol{0}\}$ and a parameter $r \in \mathbb{R}_{\geq 0}$. For such a half-line, it has been already shown in the proof of Lemma 1 in [8] that, for some $r_1 > 0$ $(r_1 < \infty)$, $\boldsymbol{c}_1 = \boldsymbol{c}(r_1)$ satisfies $\boldsymbol{c}_1 \in F_{\Pi}(p_1, s_1)$ for $\exists (p_1, s_1) \in (\Pi \times \{1, \ldots, 4\}) \backslash \psi$ such that $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p, s) \in \psi \cup \{(p_1, s_1)\}\}$ is independent. $\boldsymbol{c}_2$, on the other hand, is found on the half-line running in the opposite direction from $\boldsymbol{b}$, whose point is represented by $\boldsymbol{c}'(r) = \boldsymbol{b} + r(-\boldsymbol{v})$. Since it is also a half-line on the same flat, for some $r_2 > 0$ $(r_2 < \infty)$, $\boldsymbol{c}_2 = \boldsymbol{c}'(r_2)$ satisfies $\boldsymbol{c}_2 \in F_{\Pi}(p_2, s_2)$ for $\exists (p_2, s_2) \in (\Pi \times \{1, \ldots, 4\}) \backslash \psi$ such that $\{h_{(p,s)}(\boldsymbol{a}) = 0 \mid (p, s) \in \psi \cup \{(p_2, s_2)\}\}$ is independent. Now, $\boldsymbol{b}$ is on the line segment connecting $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, which means that $\boldsymbol{b}$ is represented as a convex combination of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. □

References

1. Buzer, L.: An incremental linear time algorithm for digital line and plane recognition using a linear incremental feasibility problem. In: Braquelaire, A., Lachaud, J.-O., Vialard, A. (eds.) DGCI 2002. LNCS, vol. 2301, pp. 372–381. Springer, Heidelberg (2002). doi:10.1007/3-540-45986-3_33
2. Kenmochi, Y., Buzer, L., Talbot, H.: Efficiently computing optimal consensus of digital line fitting. In: International Conference on Pattern Recognition (ICPR 2010), pp. 1064–1067. IEEE (2010)
3. Largeteau-Skapin, G., Zrour, R., Andres, E.: $O(n^3 \log n)$ time complexity for the optimal consensus set computation for 4-connected digital circles. In: Gonzalez-Diaz, R., Jimenez, M.-J., Medrano, B. (eds.) DGCI 2013. LNCS, vol. 7749, pp. 241–252. Springer, Heidelberg (2013). doi:10.1007/978-3-642-37067-0_21
4. Largeteau-Skapin, G., Zrour, R., Andres, E., Sugimoto, A., Kenmochi, Y.: Optimal consensus set and preimage of 4-connected circles in a noisy environment. In: Proceedings of the International Conference on Pattern Recognition (ICPR 2012), pp. 3774–3777. IEEE (2012)
5. Provot, L., Gerard, Y.: Recognition of digital hyperplanes and level layers with forbidden points. In: Aggarwal, J.K., Barneva, R.P., Brimkov, V.E., Koroutchev, K.N., Korutcheva, E.R. (eds.) IWCIA 2011. LNCS, vol. 6636, pp. 144–156. Springer, Heidelberg (2011). doi:10.1007/978-3-642-21073-0_15
6. Sekiya, F., Sugimoto, A.: Fitting discrete polynomial curve and surface to noisy data. Ann. Math. Artif. Intell. **75**(1–2), 135–162 (2015)
7. Sekiya, F., Sugimoto, A.: On connectivity of discretized 2D explicit curve. In: Ochiai, H., Anjyo, K. (eds.) Mathematical Progress in Expressive Image Synthesis II. MI, vol. 18, pp. 33–44. Springer, Tokyo (2015). doi:10.1007/978-4-431-55483-7_4

8. Sekiya, F., Sugimoto, A.: Discrete polynomial curve fitting guaranteeing inclusion-wise maximality of inlier set. In: Chen, C.-S., Lu, J., Ma, K.-K. (eds.) ACCV 2016. LNCS, vol. 10117, pp. 477–492. Springer, Cham (2017). doi:10.1007/978-3-319-54427-4_35
9. Sere, A., Sie, O., Andres, E.: Extended standard Hough transform for analytical line recognition. In: Proceedings of International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT 2012), pp. 412–422. IEEE (2012)
10. Toutant, J.-L., Andres, E., Largeteau-Skapin, G., Zrour, R.: Implicit digital surfaces in arbitrary dimensions. In: Barcucci, E., Frosini, A., Rinaldi, S. (eds.) DGCI 2014. LNCS, vol. 8668, pp. 332–343. Springer, Cham (2014). doi:10.1007/978-3-319-09955-2_28
11. Zrour, R., Kenmochi, Y., Talbot, H., Buzer, L., Hamam, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital line and plane fitting. Int. J. Imaging Syst. Technol. **21**(1), 45–57 (2011)
12. Zrour, R., Largeteau-Skapin, G., Andres, E.: Optimal consensus set for annulus fitting. In: Debled-Rennesson, I., Domenjoud, E., Kerautret, B., Even, P. (eds.) DGCI 2011. LNCS, vol. 6607, pp. 358–368. Springer, Heidelberg (2011). doi:10.1007/978-3-642-19867-0_30
13. Zrour, R., Largeteau-Skapin, G., Andres, E.: Optimal consensus set for nD fixed width annulus fitting. In: Barneva, R.P., Bhattacharya, B.B., Brimkov, V.E. (eds.) IWCIA 2015. LNCS, vol. 9448, pp. 101–114. Springer, Cham (2015). doi:10.1007/978-3-319-26145-4_8