Using a Multiobjective Genetic Algorithm for Curve Approximation

Tim Sabsch, Christian Braune, Alexander Dockhorn, and Rudolf Kruse
Institute of Intelligent Cooperating Systems
Department of Computer Science, Otto-von-Guericke-University Magdeburg
Universitaetsplatz 2, 39106 Magdeburg, Germany
Email: {tim.sabsch, christian.braune, alexander.dockhorn, rudolf.kruse}@ovgu.de

***Abstract*—Fitting a parametric curve to unordered point cloud data is a frequently encountered problem in areas, where raster data has to be vectorized, or advanced geometric descriptors of point clouds are to be found. Existing approaches often struggle with certain geometric properties, such as varying density, self-intersections, sharp corners, or are only designed to handle low-dimensional and discrete data. With the purpose to overcome these difficulties, the applicability of evolutionary algorithms to the topic of curve approximation is studied in this work. Based on the popular algorithm NSGA-II, an implementation has been developed that uses the distance to the point cloud and the number of control points of a curve as objective functions. The evaluation reveals that the proposed objective functions control the evolutionary process well, and the final curves fit most of the evaluated data sets correctly. The results of the study indicate the usefulness of genetic algorithms for the topic of curve fitting and form a basis for future research in this area.**

I. Introduction

Unordered point sets are frequently encountered in the field of computer-aided geometric design (CAGD) and image processing. Scanning devices, such as cameras or 3D-scanners, take images of objects and scenes and store them in a rectangular grid of pixels or voxels, known as raster data. This discrete structure represents the given input in a human-perceptible way, but often cannot directly be applied to advanced algorithms in computer-aided design. Instead, many of these applications require vector data such as lines, curves, or surfaces. In reverse engineering, geometric models of existing objects are created in order to review designs, reproduce objects, or project them into new scenarios. Here, converting raster images into vector data becomes more and more important. Of particular interest in CAGD are B-spline curves, as their piecewise polynomial structure makes them flexible and quick to compute. Additionally, their mathematical definition can be easily extended to surfaces and hyperplanes.

In data mining, large databases with high-dimensional data are often represented as point clouds, where each record in a database refers to one element in the cloud. A common task is to segment the data set into several clusters, based only on the intrinsic properties, i.e. without having any additional information like class assignments. Model-based clustering techniques such as k-means or fuzzy c-means group the data based on the distance to the geometric center of a cluster - the centroid [1]. These clustering algorithms tend to form (hyper-) spherical clusters and therefore struggle with non-convex data. To apply model-based clustering algorithms to non-convex data, one may use a different kind of centroid than a simple point [2]. For clusters whose shape is aligned towards a main direction, curves can be used as more appropriate representatives of the centroid structure.

Genetic algorithms have been successfully used in the area of curve fitting for the determination of optimal number and location of knots (e.g. [3]); however, these approaches require the order of the data to be given, for example in an interpolation task. Other strategies that do not assume such an ordering are restricted to discrete and low-dimensional data, or cannot fit point clouds of certain geometric properties. To find a solution for these constraints, in this work the applicability of evolutionary algorithms for curve approximation is studied. An exemplary implementation based on the popular multiobjective evolutionary algorithm NSGA-II is presented and compared to an approach based on the local tangential flow of a data segment. In particular, the following requirements have been specified: first, the approach should be able to process continuous data and data sets of high dimensions. Second, the algorithm should not be restricted to geometric characteristics such as local continuity or a uniform noise variance. Instead, it should be able to handle self-intersections, non-continuous segments (i.e. sharp corners), and point clouds with varying density.

B-Spline Curve Fitting

Given a set of data points, the aim of B-spline curve fitting is to find a B-spline curve $C(t) = \sum_{i=0}^{n} N_{i,k}(t)P_i$ that approximates the shape of the data best, and therefore minimizes the distance to the point cloud. Here, $\mathbf{P} = (P_0, P_1, \ldots, P_n)$ is the set of control points; $\mathbf{N_k(t)} = (N_{0,k}(t), N_{1,k}(t), \ldots, N_{n,k}(t))$ is the set of B-spline basis functions defined over a degree $k-1$ and a non-decreasing sequence of real numbers $\mathbf{T} = (t_0, t_1, \ldots, t_{n+k})$, the knot vector. Two useful properties of B-spline curves may be mentioned in the following: first, a curve is said to be clamped uniform, if the first and last knot in the knot vector each has multiplicity k, and the remaining knots are evenly spaced. Such a curve starts at the first control point and ends at the last control point. Second, due to the definition of the basis functions, a control point P_i only influences the curve in the interval $[t_i, t_{i+k})$ (local control property).

In this work, the subtopic of curve approximation (also known as curve extraction / reconstruction) is addressed, which assumes the data to be noisy or at least non-interpolable.

The remaining article is structured as follows: In Section II, an overview on existing work for the problem of curve approximation is given. Motivated by the drawbacks of existing work, Section III introduces the implemented genetic algorithm and the used objective functions. Section IV presents the results of the evaluation and discusses the potential for future work. Section V finalizes the article by summarizing the key findings.

II. Related Work

The problem of curve approximation is widely acknowledged in the literature. While a thorough review of published studies would go beyond the scope of this work, a short overview is given in the following.

If the order of the data points are already known (e.g. because it represents a function or time series), the curve can be obtained by a regression such as least squares [4]. A good data parametrization is critical to the performance of the fitting process [5]. Recently, computational intelligence techniques such as artificial immune systems [6], [7] or genetic algorithms [3], [8]–[10] became popular for this problem.

Many applications however can not provide an ordering on the point cloud. To approximate such a data set by a curve, one can either determine an order and apply one of the optimization approaches listed above, or use a fitting algorithm that does not require an ordering.

Some methods construct a one-dimensional (thinned) representation - a skeleton - of the data first. Levin [11] uses the moving least-squares method to thin a point cloud. The extracted skeleton can then be used to interpolate a smooth curve. Lee [12] studies the influence of the neighborhood size to the thinning process in moving least squares, suggests optimal neighborhood sizes, and applies the algorithm to pipe surface reconstruction. These algorithms however originate from early applications in CAGD, where discrete, low-dimensional data has to be processed, and are hard to adapt to continuous or high-dimensional data.

Other strategies divide the point cloud into several subsets first. Yan [13] for example introduces the fuzzy curve-tracing algorithm, an approach based on fuzzy-c-means that partitions the data first and constructs a relational graph based on the cluster neighborhoods. Lin et al. [14] propose the sequence joining method, a clustering algorithm to detect rectangular clusters. The sequence is enveloped by an interval B-spline curve; its centric curve is taken as the reconstructed curve. Both approaches struggle however with self-intersecting curves.

Some algorithms construct a graph representation of the data. Sun et al. [15] apply a Delaunay triangulation and weight the resulting graph with a Gaussian potential function. Given a start and end node, an ordering of the data is then determined by the shortest path computed by the A* algorithm. Bo et al. [16] delete edges in the triangulated data, whose length is larger than a threshold to receive several connected components, which are separately thinned and fitted. The drawback of these algorithms is the inefficiency of Delaunay triangulation in data sets with more than three dimensions.

Another family of approximation algorithms analyze the local tangential flow of the data. Liu et al. [17] place a short smooth curve onto the point cloud and let it grow at both ends by iteratively analyzing the principal components in each segment. This approach is able to deal with self-intersections, sharp corners, and high dimensions, but struggles with point clouds of varying thickness. Motivated by vectorizing raster image data to make it applicable to image processing techniques, Furferi et al. [18] use the PCA to construct an ordered set of points. The polynomial chain, built by this set, orders the point cloud. Ruiz et al. [19] tackle the problem of (near-parallel) self-intersections. In their work, the currently processed region can be extended to an ellipse that has its major axis in the direction of the tangent vector detected one iteration step before. Strategies based on the local tangential flow are fast and applicable to point clouds of high dimensions, but often struggle when confronted with self-intersections and sharp corners, although recent publications deal with those cases. In addition, the commonly used PCA is sensitive to the density of the point cloud. If only few data points are available in an iteration, the first principal component may not represent the true local direction.

III. Fitting Curves with Evolutionary Algorithms

The main idea of this work is to use the flexibility of evolutionary algorithms for the task of curve fitting. While most other approaches fail at certain properties of the data, a genetic algorithm may still find a good solution in those cases. To investigate this statement, an exemplary implementation based on NSGA-II [20] is presented and evaluated in the following.

A. Constraints

To make the challenge of curve fitting with genetic algorithms feasible in a first study, several restrictions of the curves' structure and the optimization problem have been defined beforehand.

First, the curves developed in the algorithm are fixed to be cubic, i.e. of degree 3. This guarantees a parametric continuity of C^2, which leads to a visible smoothness at each location on the curve [21]. This is in most cases desired, except for very sharp corners in the cloud, where a non-continuous curve may fit the data better. In addition, a cubic degree does not weaken the local control property of B-spline curves much; changing a control point will only influence the curve in an interval of four knots.

Second, the optimization of a curve to a given point cloud consists of two parts: finding an ideal set of control points and finding an ideal knot vector. In this work, the second step - optimizing the knot vector given a set of control points - is omitted; instead, the knot vector is defined as being clamped uniform, and automatically computed.

Lastly, the output of the proposed algorithm is not a single candidate, but the first non-dominated Pareto front, i.e. a set of solutions. Selecting one individual out of this set as the final result remains a task of future research. In the evaluation, this has been done manually.

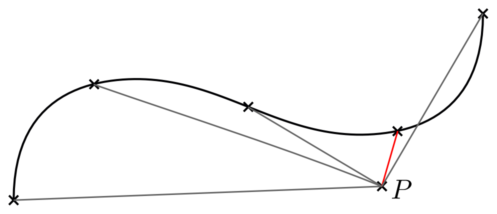


Fig. 1. Distance computation. Several points are sampled uniformly on the curve. The distance of a data point P to the curve is approximated by the distance to the closest sample point (red line).

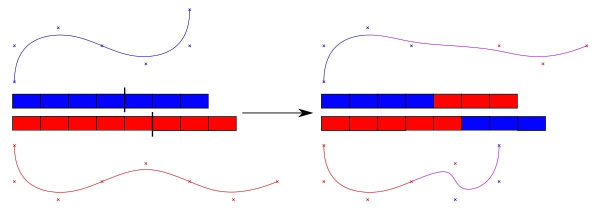


Fig. 2. One-point crossover. The parents are cut after position 4 and 5. Blue and red curve segments are influenced by control points of one parent. Purple curve segments are influenced by control points of both parents.

B. Objectives

The goal of curve fitting is to find a B-spline curve that represents the point cloud as good as possible, i.e. minimizes the distance from curve to data. Determining the distance from a point to the curve however is not trivial and usually done by a Newton iteration [22]. In this work, a sufficient large number of points is sampled uniformly on the curve; the minimum distance from a point to the curve is set as the distance to the closest sample (Figure 1). For a single individual in one generation, this yields a complexity of $\mathcal{O}(m \cdot n)$ for m samples and n data points in the set.

An algorithm only minimizing the distance may tend to interpolate the data instead of approximating it. If the data contains noise, interpolation is not desirable, as it causes overfitting. For curve approximation, an algorithm might neglect the distance minimization in favor of preserving simplicity. Here, the objective can therefore be extended by the desire of keeping the curve as simple as possible. Several properties of a curve can be used to detect this characteristic, including the number of control points, the curve length, near-parallel segments, or segments outside the point cloud. In this work, the number of control points is analyzed. In a later test, the curve length has been incorporated as a third fitness criterion. The computation of both criteria is less complex than the distance approximation and therefore do not increase the overall computational costs.

All criteria are independent from each other, which is why they can be treated as fitness functions in a multiobjective optimization scenario. The algorithm of choice in this work is non-dominated sorting genetic algorithm II (NSGA-II) [20].

C. Implementation Details

As described in the constraints, subject of optimization is the set of control points, where the optimal number of points is not known beforehand. The obvious encoding for this setting is a vector, in which each record stores the location of a control point. Since the number of control points is not fixed, the length of the chromosome is variable, albeit it has a lower boundary. Due to the cubic degree of the curves, at least four control points are needed to construct a curve. The search space induced by this encoding is closed, with one exception: crossing two individuals or mutating a chromosome may construct a chromosome with less than four control points, which is not allowed. To avoid this edge case, it is treated as a special case in the implementation.

TABLE I
PARAMETER SETTINGS IN THE EVALUATION

Parameter	Value
Number of generations	1000
Population size	100
Crossover probability	0.10
Probability of mutating control point number	0.01
Probability of mutating control point location	0.25

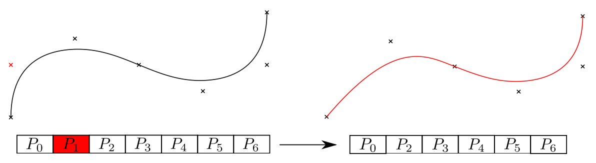
To recombine two individuals, the well-known one-point crossover is used (Figure 2). As the chromosomes have a variable length, the cutting position is determined independently for each parent.

Three mutation operators are implemented in order to modify a chromosome: First, a control point may be removed from the vector. Alternatively, a new control point may be added randomly to the individual. When the number of control points changes, the knot vector of the curve has to be recomputed. This affects the shape of the whole curve. Figure 3a shows an example of this mutation type. Second, the location of a control point may change. For each dimension, a random factor based on a zero-mean Gaussian distribution is generated and added to the current location. The standard deviation should be based on the size of the data. As the processed data is normalized into the interval of $[0, 1]$, the standard deviation is fixed to $\sigma = 0.05$. Changing a control point only affects the curve on a local interval due to the local control (Figure 3b). Lastly, two control points may be swapped. For two control points P_i and P_j, the curve is affected in the area of $[t_i, t_{i+k}) \cup [t_j, t_{j+k})$. An example is given in Figure 3c. An evaluation of the mutation operations revealed that swapping control points is not beneficial to the performance and is not considered for the following section.

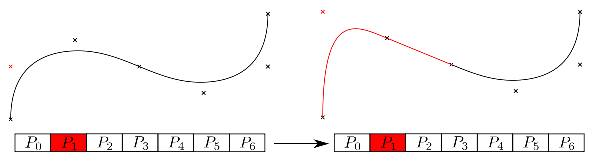
IV. Evaluation

The presented algorithm (in the following denoted by **EA-Fit**) has been evaluated on seven different data sets to analyze its behavior with respect to certain geometric properties. The algorithms' parameter settings have been determined empirically and are shown in Table I. Additionally, it has been compared to the PCA-based strategy published by Furferi et al. [18] (in the following denoted by **PCA-Fit**).

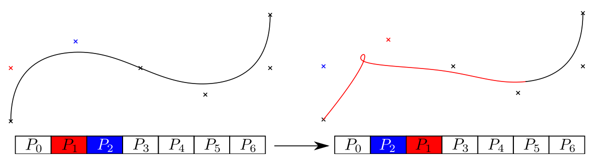
The quality of the approximations is determined with respect to the objective of curve fitting: the minimization of the distance from the curve to the point cloud. To compute the distance, the same method is used as in the evolutionary algorithm. Because


P_0 P_1 P_2 P_3 P_4 P_5 P_6
P_0 P_2 P_3 P_4 P_5 P_6

(a) Mutation of control point number. The second control point is removed.


P_0 P_1 P_2 P_3 P_4 P_5 P_6
P_0 P_1 P_2 P_3 P_4 P_5 P_6

(b) Mutation of control point location. The second control point is changed.


P_0 P_1 P_2 P_3 P_4 P_5 P_6
P_0 P_2 P_1 P_3 P_4 P_5 P_6

(c) Mutation of control point order. The second and third control point are swapped.

Fig. 3. Mutation operators. The red segments show the curve intervals affected by the mutation.

TABLE II
SUM OF SQUARED ERROR RESULTS.

Data Set	EA-Fit	PCA-Fit
Open Point Cloud	**0.4314**	0.4806
Closed Point Cloud	**0.4744**	1.3197
Varying Density	**0.2500**	0.3617
Sharp Corners	**0.0577**	6.0421
Self-Intersections	**0.6195**	53.5088
Three-Dimensional Data	**0.6819**	1.1549
Background Noise	**0.8018**	2.5212

the result of the implemented genetic algorithm is not a single solution, but a Pareto front, one has to choose one solution candidate among the front members as the final curve. Figure 4 shows an example of such a resulting final Pareto front. Due to the diversity preservation of NSGA-II, most solution candidates either under- or overfit the data. Nevertheless, those individual can prove useful during the evolutionary optimization process, since they cover properties that fit specific objectives best. Combining other elements with those individuals can be the key for obtaining even better solutions.

As only two of the eight front members have the desired shape, it becomes obvious that the selection of the final curve is critical to the overall algorithm's performance. In this work, no automatic decision making system has been implemented; instead, the final solution is selected manually. The results of the evaluation are presented in Table II and Figure 5.

A simple open point cloud, such as shown in Figure 5a, can be properly approximated by both algorithms. This also applies, if the noise level of the cloud is not uniform, but varies (Figure 5b). If the point cloud has a closed form, like in a circle, the correct placement of the end points is most important. In the tests, EA-Fit does that slightly better (Figure 5c).

A reliable fitting algorithm should be able to handle geometric characteristics such as self-intersections or sharp corners. The latter is evaluated in Figure 5d. EA-Fit finds the correct angles at the turn points by setting multiple control points at the corners. The PCA-based approach on the other hand fails to detect the sharp corners and stops, where the tangents cannot follow the cloud any more. The behavior of the algorithms at intersecting segments is considered in Figure 5e. EA-Fit correctly identifies the directions of the data set at the intersection, and sets the end points of the resulting curve at proper positions.

Another data set tests the algorithms with regard to their applicability in high-dimensional spaces, here for the three-dimensional case. Figure 5f shows the resulting curves. The simple spring is correctly detected in both curves.

In contrast to the previous data sets, Figure 5g contains data points that do not belong to the shape that is to be approximated. As one can see, EA-Fit finds the sinus shape, but generates a longer curve to approximate some of the outlier data. In a later study, this data set has been tested again by an extended version of EA-Fit that considers the curve length as a third fitness criterion. The result is shown in Figure 5h. The genetic algorithm almost found the desired shape.

Discussion and Outlook

The evaluation shows that EA-Fit presents an attractive solution to the problem of curve approximation in unorganized point clouds. Characteristics such as a varying density, sharp corners or self-intersections do not limit the algorithm. As the algorithm cannot detect outliers in a data set, they worsen the quality of the final result.

It should be pointed out again that the current implementation does not contain a decision maker; the final selection among the last Pareto front has to be done manually. A disadvantage of the proposed strategy is its need to compute the distance of the curve to all points in the cloud in each generation, and for each individual . This is computationally expensive, even with the used sampling strategy and moderately sized data sets ($<$1000 points). For larger data sets, a more efficient distance computation is necessary.

Yet, the results of this study show the potential of evolutionary computation for the problem of curve approximation and justify future research, that may consider many aspects. For one, the implementation of an automatic decision-making system can finalize the process of finding a single curve for a given data set. A popular choice for such a decision maker is choosing the individual that has the largest marginal hypervolume [23]. This may exclude individuals which rank high in a minority of objectives and favor well-balanced solutions. Another method would be to enforce a weight on each objective to calculate a final fitness value and report the best individual or to use additional objectives as validation criterions.

Nevertheless, the focus in this work was laid on the general applicability of evolutionary algorithm to the problem of curve approximation, not its own optimization. Later work may

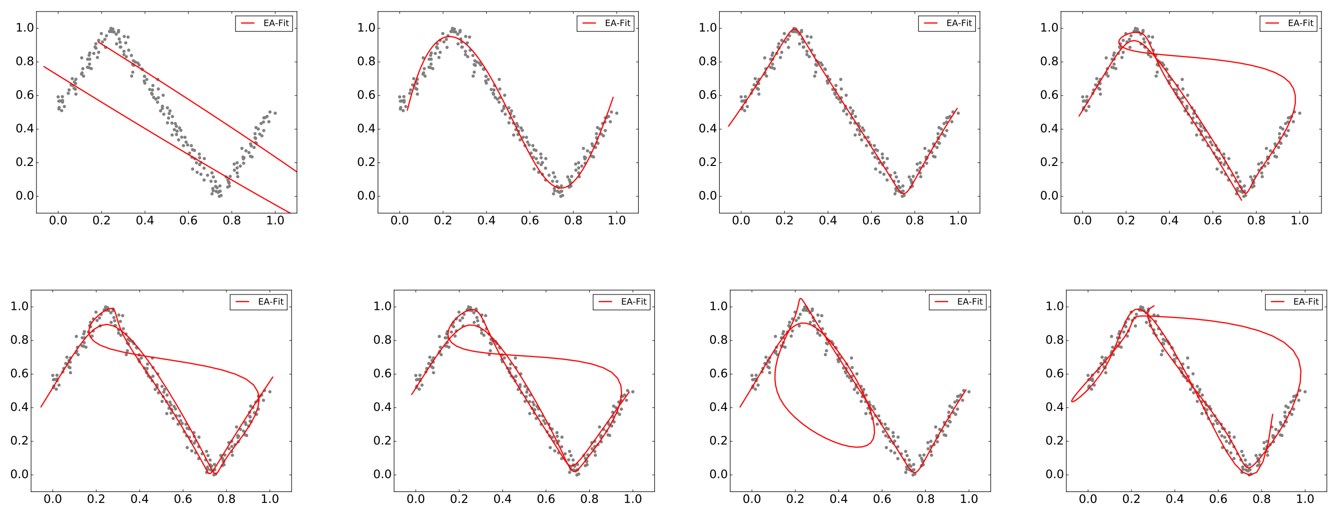
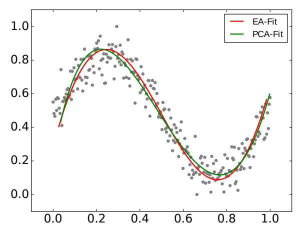
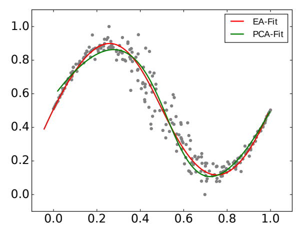


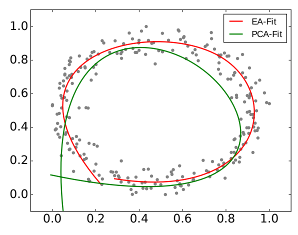
Fig. 4. Exemplary final Pareto front of EA-Fit. Each figure shows one front member.



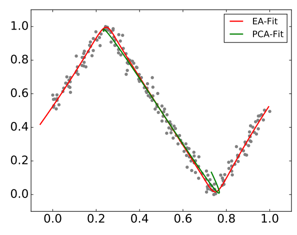
(a) Open Point Cloud



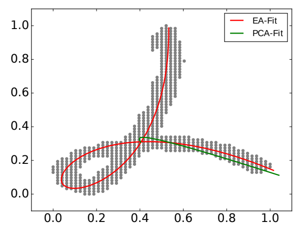
(b) Varying Density



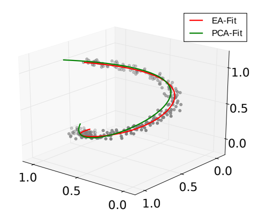
(c) Closed Point Cloud



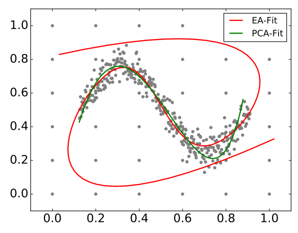
(d) Sharp Corners



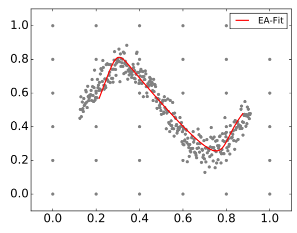
(e) Self-Intersection



(f) Three-Dimensional Data



(g) Background Noise



(h) Background Noise

Fig. 5. Evaluation results. (a) - (g): resulting curves of EA-Fit and PCA-Fit for different data sets. (h): the curve length has been added as a third objective.

consider an evaluation of different strategies in the algorithm design, such as the choice of recombination or mutation operators. Another possible improvement of the algorithm concerns the reduction of execution time caused by the distance computation.

One may also discuss other fitness criteria. Objectives like a short curve length, avoidance of near-parallel curve segments or detection of curve segments outside the point cloud might be beneficial to the performance and should be discussed more intensly in later studies. Adding additional criteria can prove useful in detecting and modelling a higher variety of shapes, e.g. detecting closed circles.

In a second evaluation series, the curve length has been added as a third criterion, which has improved the curve shape in an environment with background noise. While in the current implementation all criteria have been weighted equally, it might be worth to study the influence of each criterion and find better weights thereby. Of course, the more fitness criteria are added, the less significant the Pareto front becomes. For a higher number of objectives, one should therefore consider using a Many-Objective Genetic Algorithm instead, such as NSGA-III [24].

The proposed algorithm has been restricted to several simplifications, which can be addressed as well. On the one hand, the optimization process is constrained to the control point vector; the knot vector is computed automatically. While this is sufficient in most cases, not all point clouds are C^2-continuous at all locations and require a non-uniform curve. One the other hand, the incorporation of the knot vector into the optimization might be subject of future research, too.

Follow-up work may also extend the domain of the fitting algorithm. The current approach only considers one point cloud per data set, which is fitted by one curve. A more advanced strategy may detect, if a data set contains more than one separated point cloud, and fits a curve to each of the clouds. This idea may as well be used to find a clustering on the data. Also, the design of the algorithm allows to fit more complex structures to point clouds. The concept of B-spline curves can be easily expanded to parametric surfaces or hyperplanes of any dimensionality.

V. Conclusion

In this paper, the applicability of evolutionary algorithms to the problem of curve fitting of an unordered, noisy point cloud has been studied. A survey of existing research revealed that many publications deal with the problem of approximation, but often struggle with point clouds of high dimensions or specific geometric characteristics. To overcome these difficulties, a strategy was presented, which is based on the multiobjective genetic algorithm NSGA-II. Two properties of fitting curves were defined as objective functions for the evolutionary algorithm: the distance to the point cloud as well as the number of control points to preserve simplicity. An evaluation on seven data sets substantiates the useful- and robustness of the algorithm. Critical characteristics, such as non-uniform noise, sharp corners and self-intersections as well as higher-dimensional data was fitted properly. The results show potential for improvements and extensions to other domains.

References

[1] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, and M. Steinbrecher, *Computational Intelligence*, 2nd ed., ser. Texts in Computer Science. London: Springer London, 2016.

[2] C. Braune and R. Kruse, "Fuzzy Density-Based Clustering with Generalized Centroids," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, no. 3. IEEE, 2016.

[3] C. H. Garcia-Capulin, F. J. Cuevas, G. Trejo-Caballero, and H. Rostro-Gonzalez, "A hierarchical genetic algorithm approach for curve fitting with b-splines," *Genetic Programming and Evolvable Machines*, vol. 16, no. 2, pp. 151–166, 2015.

[4] P. Diercx, *Curve and Surface Fitting with Splines*, 1st ed. New York: Oxford University Press, 1995.

[5] W. Ma and J. Kruth, "Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces," *Computer-Aided Design*, vol. 27, no. 9, pp. 663–675, 1995.

[9] F. Yoshimoto, T. Harada, and Y. Yoshimoto, "Data fitting with a spline using a real-coded genetic algorithm," *Computer-Aided Design*, vol. 35, no. 8, pp. 751–760, 2003.

[6] E. Ülker and A. Arslan, "Automatic knot adjustment using an artificial immune system for B-spline curve approximation," *Information Sciences*, vol. 179, no. 10, pp. 1483–1494, 2009.

[7] A. Iglesias, A. Gálvez, and A. Avila, *Discrete Bézier Curve Fitting with Artificial Immune Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 59–75.

[8] F. Yoshimoto, M. Moriyama, and T. Harada, "Automatic Knot Placement by a Genetic Algorithm for Data Fitting with a Spline," in *Proceedings Shape Modeling International '99. International Conference on Shape Modeling and Applications*. IEEE, 1999, pp. 162–169.

[10] L. Pingping, Z. Xiuyang, and Y. Bo, "Automatic Knot Adjustment by an Improved Genetic Algorithm," in *2010 2nd International Conference on Future Computer and Communication*, vol. 3. IEEE, 2010, pp. 763–768.

[11] D. Levin, "The Approximation Power of Moving Least-Squares," *Mathematics of Computation*, vol. 67, no. 224, pp. 1517–1532, 1998.

[12] I.-K. Lee, "Curve reconstruction from unorganized points," *Computer Aided Geometric Design*, vol. 17, no. 2, pp. 161–177, 2000.

[13] H. Yan, "Fuzzy Curve-Tracing Algorithm," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 31, no. 5, pp. 768–780, 2001.

[14] H. Lin, W. Chen, and G. Wang, "Curve Reconstruction Based on an Interval B-Spline Curve," *The Visual Computer*, vol. 21, no. 6, pp. 418–427, 2005.

[15] Y. Sun, C. Zhou, C. Cai, and M. Ding, "Curve Reconstruction from a Set of Dense Scattered Points using A* algorithm," in *MIPPR 2009: Automatic Target Recognition and Image Analysis*, T. Zhang, B. Hirsch, Z. Cao, and H. Lu, Eds., vol. 7495, 2009.

[16] P. Bo, G. Luo, and K. Wang, "A graph-based method for fitting planar B-spline curves with intersections," *Journal of Computational Design and Engineering*, vol. 3, no. 1, pp. 14–23, 2016.

[17] Y. Liu, H. Yang, and W. Wang, "Reconstructing B-spline Curves from Point Clouds–A Tangential Flow Approach Using Least Squares Minimization," in *International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, vol. 2005. IEEE Computer Society, 2005, pp. 4–12.

[18] R. Furferi, L. Governi, M. Palai, and Y. Volpe, "From unordered point cloud to weighted b-spline: A novel pca-based method," in *Proceedings of the 2011 American Conference on Applied Mathematics and the 5th WSEAS International Conference on Computer Engineering and Applications*. World Scientific and Engineering Academy and Society, 2011, pp. 146–151.

[19] O. Ruiz, C. Vanegas, and C. Cadavid, "Ellipse-based principal component analysis for self-intersecting curve reconstruction from noisy point sets," *The Visual Computer*, vol. 27, no. 3, pp. 211–226, 2011.

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[21] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction To Splines For Use In Computer Graphics & Geometric Modeling*. Los Altos: Morgan Kaufmann, 1987.

[22] L. Piegl and W. Tiller, *The NURBS Book*, 2nd ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996.

[23] E. Zitzler, D. Brockhoff, and L. Thiele, "The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration," in *Evolutionary Multi-Criterion Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, vol. 4403, pp. 862–876.

[24] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints." *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.