Cuckoo Search Algorithm with Lévy Flights for Surface Reconstruction from Point Clouds with Applications to Reverse Engineering

ABSTRACT

Surface reconstruction is a classical task in industrial engineering and manufacturing, particularly in reverse engineering, where the goal is to obtain a digital model from a physical object. For that purpose, the real object is typically scanned and the resulting point cloud is then fitted through mathematical surfaces via numerical optimization. The choice of the approximating functions is crucial for the accuracy of the process. Unfortunately, real-world objects often require complex nonlinear approximating functions, which are not well suited for standard numerical optimization methods. In this paper, we overcome this limitation by using a cuckoo search algorithm with Lévy flights, a swarm intelligence technique envisioned for global optimization. The method is applied to three illustrative examples of point clouds fitted by using a combination of exponential, polynomial and logarithmic functions. The experimental results show that the method performs well in recovering the shape of the point clouds accurately. We conclude that the method is promising towards its application to manufactured workpieces in real industrial settings.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Shape modeling**; **Parametric curve and surface models**; **Modeling and simulation**.

KEYWORDS

Artificial intelligence, manufacturing systems, nonlinear optimization, reverse engineering, surface reconstruction, point clouds, cuckoo search algorithm, Lévy flights

1 INTRODUCTION

1.1 Reverse engineering

The classical pipeline in current manufacturing industry is to proceed from digital designs to the manufacturing of physical products. The process begins with the initial design of the product, which is typically performed by computer, taking advantage of sophisticated computer programs for CAD/CAM (computer-aided design/manufacturing). The computer design is visualized using powerful graphical libraries and the mathematical description of the object is used for the analysis of both aesthetical and functional requirements that, in turn, will enforce new computer designs. The repetition of this process leads to a closed loop manufacturing cycle, involving different steps such as design (CAD models), measurement, quality assessment and monitoring, problem solving, tolerance analysis and simulation, and others. Once all design and engineering requirements are fully met, the computational model is validated and the digital files are finally used for final manufacturing.

There is, however, a second potential approach called *reverse engineering*, which proceeds in the opposite way [29]. Instead of digital models, the input in reverse engineering is the actual physical object, and the goal is to obtain a digital representation of the object in terms of mathematical curves and surfaces. This digital

model is expected to replicate the original object accurately, where the prescribed accuracy will depend on the industrial domain.

There are various reasons to apply reverse engineering in industry. One of the main reasons is to gain knowledge about the way a product works, or how a given workpiece will behave under different structural forces and other conditions. Reverse engineering facilitates this kind of structural analysis in a safe and controllable environment and in a more economical way via computer simulation. In other cases, reverse engineering is used to repurpose obsolete objects for which a digital model is no longer available. This is a typical situation for complex products created decades ago, which can be comprised of many complex workpieces that are out of production and, hence, become irreplaceable. In other cases, the digital model is used to analyze the quality of shape using shape interrogation techniques [28] and other methods. Another reason comes from intellectual property infringement assessment, where the digital model is used to know how a complex manufactured good has been obtained, thus helping to determine whether it infringes any intellectual or industrial property right. Finally, reverse engineering is widely used now to modify the original shape of the manufactured workpiece for mass customization, a very popular trend in current product design strategies.

1.2 Surface reconstruction

Surface reconstruction is a classical process in industrial engineering and manufacturing, particularly in reverse engineering. The usual pipeline in reverse engineering begins with the manufactured good, which is digitized in some way (typically through 3D scanning, but other methods can also be used). The usual output of this step is a collection of point data called *point cloud*. This point cloud is then fitted through mathematical surfaces using some approximating functions [33]. Generally, this process is performed by minimizing the distance between the digital model and the digitized data points. This leads to a minimization problem, usually addressed through least-squares-based numerical procedures (see Sect. 2 for further details).

The choice of the approximating functions plays a crucial role for the accuracy of the process. This is an important observation, since many real-world objects require complex nonlinear approximating functions, which are not well suited for standard numerical optimization methods. In this regard, novel approaches have been introduced during the last decades to tackle this issue. Among them, those based on artificial intelligence are receiving increasing attention during the last few years. This is also the approach followed by the authors in this work.

1.3 Aims and structure of this paper

In this paper, we address the surface reconstruction problem from point clouds. In our approach, we assume that the mathematical surfaces can be accurately approximated by fitting functions that are a combination of exponential, polynomial, and logarithmic functions. These basis functions are continuous and nonlinear, and so is the resulting combined fitting function. As a result, we are confronted with a difficult continuous and nonlinear optimization problem unsolvable with the traditional numerical techniques. This limitation is solved here through the cuckoo search algorithm with Lévy flights, a swarm intelligence technique envisioned for global optimization. The method will be applied to three illustrative examples of point clouds with successful results. The novelty of this work lies on the fact that, to the best of our knowledge, no previous artificial intelligence-based method has addressed this problem in the context of reverse engineering.

The structure of this paper is as follows: Sect. 2 summarizes the previous work in the field. Sect. 3 describes the optimization problem addressed in this work. Our method to solve it is described in detail in Sect. 4. The performance of the method is illustrated through three illustrative examples, which are discussed in Sect. 5. The paper closes in Sect. 6 with the main conclusions and some ideas for future work in the field.

2 PREVIOUS WORK

The topic of surface reconstruction from point clouds has been the subject of intensive research for decades. Early computational methods were introduced the 60s and 70s, mostly based on numerical procedures [7, 30, 31]. Subsequent advances during the 80s and 90s applied more sophisticated techniques, although they failed to provided general solutions [3, 6]. From a mathematical standpoint, this issue can be formulated as a least-squares optimization problem [23, 25, 28]. However, classical mathematical optimization techniques had little success in solving it beyond rather simple cases, so the scientific community focused on alternative approaches, such as error bounds [26], dominant points [27] or curvature-based squared distance minimization [34]. These methods provide acceptable results but they need to meet strong conditions such as high differentiability and noiseless data that are not so common in industrial settings.

More recently, methods based on artificial intelligence and soft computing are receiving increasing attention. Some approaches are based on neural networks [14], self-organizing maps [15], or the hybridization of neural networks with partial differential equations [2]. These neural approaches have been extended to functional networks in [16, 24] and hybridized with genetic algorithms [8]. Other approaches are based on support vector machines [22] and estimation of distribution algorithms [40]. Other techniques include genetic algorithms [10, 38, 39], particle swarm optimization [9, 11], artificial immune systems [13, 19], bat algorithm [20], cuckoo search algorithm [21] and hybrid techniques [12, 17, 18, 32]. It is worthwhile to mention that none of the previous approaches addressed the problem discussed in this paper.

3 THE OPTIMIZATION PROBLEM

As discussed above, in this paper we address a surface reconstruction from point clouds. In this case, the input of the optimization problem consists of a collection of R three-dimensional data points $\{(x_r, y_r, z_r)\}_{r=1,\dots,R}$. For simplicity, we restrict to the case when the point cloud corresponds to a height map. The goal is to obtain an accurate mathematical representation of a explicit surface $f(x, y)$ approximating the point cloud accurately. This condition can be formulated as the minimization problem:

$$min\left\{\sum_{r=1}^{R}\left[(x_r-\hat{x_r})^2+(y_r-\hat{y_r})^2+(z_r-f(\hat{x_r},\hat{y_r}))^2\right]\right\} \quad (1)$$

with (x_r, y_r, z_r) and $(\hat{x_r}, \hat{y_r}, \hat{z_r})$ denoting the original and reconstructed data points, respectively, where $\hat{x_r}$ and $\hat{y_r}$ can be obtained by projection of the point cloud on a base surface $B(x, y)$ determined by principal component analysis or other techniques. In this work, it is assumed that $f(x, y)$ is a combination of basis functions from the exponential, polynomial, and logarithmic families of functions. A convenient way to produce explicit surfaces under these conditions is to consider bivariate distributions whose conditionals belong to such families of basis functions [4]. Two feasible models in this regard are the Normal-Gamma and the Gamma-Gamma distribution functions. In the first case, the approximating function is given by:

$$f(x,y) = e^{F+Ay-Cy^2+(G+By-Dy^2)x+(H+Jy-Ky^2)log(x)} \quad (2)$$

which is a model depending on 9 parameters. However, these parameters are not fully free, but have to hold some constraints derived from the non-negativity and integrability properties, leading to the following constraints (see [1] for details):

$$\begin{gathered} C > K\left(1 - \log\left(\frac{-K}{D}\right)\right) \quad ; \quad D > 0 \\ G < \frac{-B^2}{4D} \quad ; \quad H > \frac{-J^2}{4K-1} \quad ; \quad K < 0 \end{gathered} \quad (3)$$

In the second case, the approximating function is given by:

$$f(x,y) = e^{C_0+xC_2+yC_6+xyC_8} x^{C_1+yC_7} y^{C_3+xC_5+C_4 \log(x)} \quad (4)$$

which depends on 9 parameters. Once again, the non-negati-vity and integrability of the marginal conditionals impose several constraints, leading to two feasible models [5]. The first model is given by the constraints:

$$C_1 > -1, C_2 < 0, C_3 \geq -1, C_4 = C_5 = C_7 = C_8 = 0, C_6 < 0 \quad (5)$$

and the second model is given by:

$$C_1 > -1, C_2 < 0, C_3 \geq -1, C_4 = C_5 = C_7 = 0, C_6 < 0, C_8 < 0 \quad (6)$$

Unfortunately, the resulting minimization problem from Eqs. (1)-(2) with the constraints in Eq. (3) or from Eqs. (1)-(4) with the constraints in either (5) or (6), is very difficult to solve, as it becomes a constrained, multivariate, nonlinear, multimodal continuous optimization problem. As a consequence, classical gradient-based mathematical techniques are not well suited to address it. In this paper, we apply a powerful metaheuristic algorithm called the cuckoo search algorithm with Lévy flights to solve this problem. It is explained in detail in next section.

4 THE PROPOSED METHOD

4.1 The cuckoo search algorithm

As discussed above, the minimization problem described in Sect. 3 becomes unsolvable for traditional mathematical techniques. As indicated in Sect. 2, artificial intelligence-based methods have proved to be good alternatives in many cases. In this paper, we consider a powerful metaheuristic technique called *cuckoo search algorithm* (CSA), introduced in 2009 by Xin-She Yang and Suash Deb to solve continuous optimization problems [36, 37]. The cuckoo search algorithm is inspired by a peculiar behavior of some cuckoo species called *obligate interspecific brood-parasitism*, in which such cuckoo species lay their eggs in the nests of host birds of other species. This is a natural yet sophisticated and striking strategy to escape from the parental investment in raising their offspring, while also minimizing the risk of egg loss to other species. Of course, this behavior plays an inspirational role exclusively, in the form of some idealized rules as the basis for a computational optimization model. In other words, the method is not a model of the natural behavior of the cuckoos, but a nature-inspired metaphor for a computational procedure, so the method must be understood in this way.

Table 1: Original Cuckoo Search Algorithm with Lévy flights proposed by Yang and Deb in [36, 37].

Algorithm: Cuckoo Search via Lévy Flights

begin
  Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, \ldots, x_d)^T$
  Generate initial population of N host nests $\mathbf{x}_i$
  while $(t < MaxGeneration)$ or (stop criterion)
    Get a cuckoo (say, i) randomly by Lévy flights
    Evaluate its fitness F_i
    Choose a nest among N (say, j) randomly
    if $(F_i > F_j)$
      Replace j by the new solution
    end
    A fraction (p_a) of worse nests are abandoned and
      new ones are built via Lévy flights
    Keep the best solutions (or nests with quality solutions)
    Rank the solutions and find the current best
  end while
  Postprocess results and visualization
end

Putting the nature-based metaphor aside, the computational approach can be summarized as follows. The method starts with an initial population of n host nests and runs in an iterative fashion. In the original proposal, the initial values of the kth component of the jth nest are determined by the expression $x_j^k(0) = \mu.(up_j^k - low_j^k) + low_j^k$, where up_j^k and low_j^k are two variables to represent the upper and lower bounds of that kth component, respectively, and μ represents a uniform random variable on the open interval $(0, 1)$. This choice ensures that the initial values of the variables are within the search space domain. Of course, these boundary conditions should also be controlled during execution at each iteration step.

For each iteration t, a cuckoo egg, say i, is selected randomly and new solutions $\mathbf{x}_i^{t+1}$ are generated. In their original paper, Yang and Deb discussed that the random search performed for the exploration stage of the algorithm can be executed more efficiently by using Lévy flights rather than with a simple random walk. The Lévy flights are a type of random walk in which the steps are defined in terms of the step-lengths. Such step-lenghts should follow a certain probability distribution in which the directions of the steps must be isotropic and random. In this case, the general equation for the

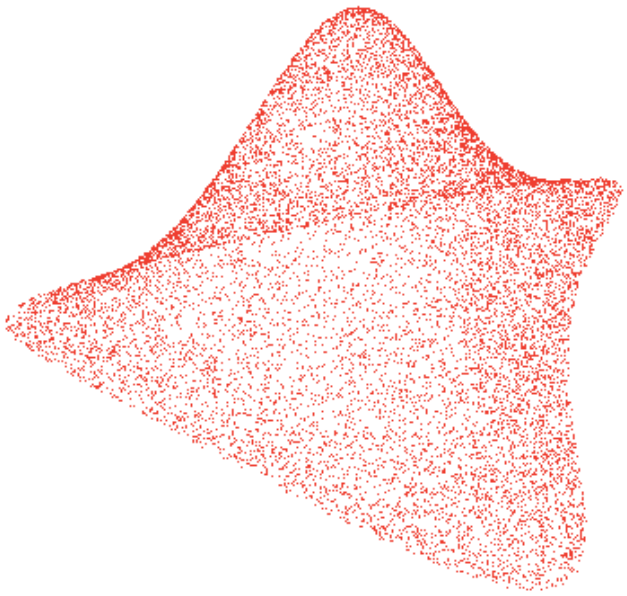

Figure 1: Original point cloud of *Example I.*

Lévy flight is given by:

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \alpha \oplus levy(\lambda) \tag{7}$$

where the superscript t is used to indicate the current generation, the symbol $\oplus$ is used to indicate the entry-wise multiplication, and $\alpha > 0$ indicates the step size. This step size determines how far a particle can move by random walk for a fixed number of iterations, and is therefore expected to be related to the scale of the particular problem under analysis.

The Lévy flights in Eq. (7) are, by construction, Markov processes with a transition probability modulated by the Lévy distribution as:

$$levy(\lambda) \sim g^{-\lambda}, \qquad (1 < \lambda \leq 3) \tag{8}$$

which has an infinite variance with an infinite mean. This means that the steps form a random walk process with a power-law step-length distribution with a heavy tail. From the computational standpoint, the generation of random numbers with Lévy flights consists of two main steps: firstly, a random direction according to a uniform distribution is chosen; then, the sequence of steps following the chosen Lévy distribution is generated (see [35] for details).

As the next step, the cuckoo search algorithm evaluates the fitness of the new solution and compares its value with the value of the current one. In case that the new solution yields a better fitness value, this new solution replaces the current one so that further improvement for the method is attained. Also, after evaluation of the fitness, a fraction of the worse nests (according to the fitness) are skipped and replaced by new solutions so as to increase the exploration of the search space looking for more promising solutions. The rate of replacement is determined stochastically through a parameter of the model called the probability value p_a, which should be properly tuned for better performance. Then, all current

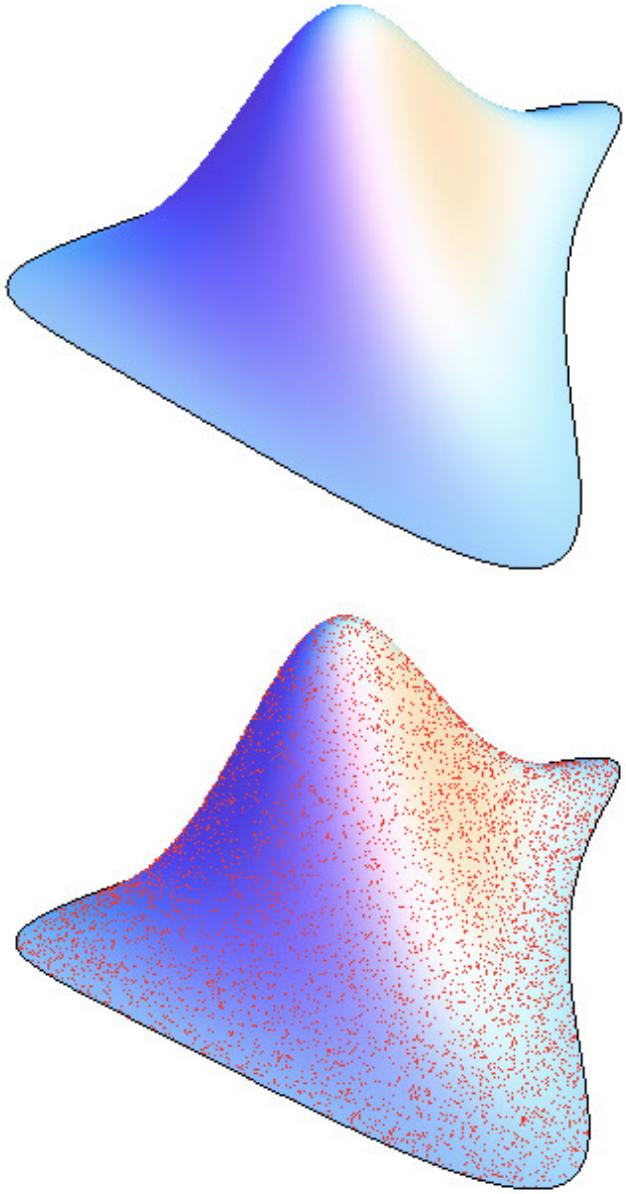

Figure 2: *Example I*: **(top) Reconstructed surface; (bottom) reconstructed surface and data points.**

solutions are ranked at each iteration step according to their fitness and the best solution reached so far is stored as the vector $\mathbf{x}_{best}$. This procedure is repeated iteratively for a maximum number of iterations, given as a parameter of the algorithm. The best solution at last iteration is chosen as the final solution of the optimization problem.

The best reconstructed surface is displayed in Fig. 2(top), and Fig. 2(bottom) shows the superposition of the fitting surface and the

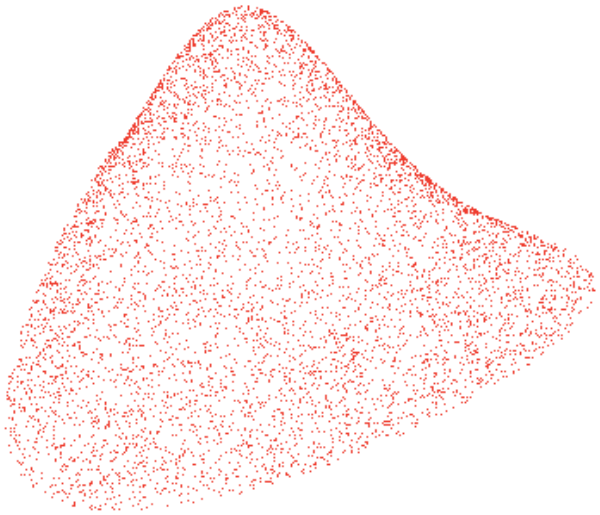

Figure 3: Original point cloud of *Example II*.

original point cloud for better visualization. From those figures, we can confirm visually the good numerical accuracy of the method, and that the fitting surface replicates the shape of the point cloud with satisfactory visual fidelity.

4.2 Population representation

To apply the cuckoo search algorithm to the optimization problem described in Sect. 3, it is required to have an adequate representation for the individuals of the population. In this work, we consider a population of N_p individuals $\mathbf{x}_i$ given by a vector of $2R + L$ components corresponding to the parameterization of the data points, $(\hat{x_r}, \hat{y_r})$, with $r = 1, \ldots, R$ and the problem constraints. Without loss of generality, each parameter can be assumed to take values on the unit interval $[0, 1]$. In other words, $\mathbf{x}_i = (\hat{x_1}^i, \ldots, \hat{x_R}^i, \hat{y_1}^i, \ldots, \hat{y_R}^i, C_1, \ldots, C_L)$, where C_j represents the j-th constraint of the optimization problem.

4.3 Parameter Tuning

An important issue of the metaheuristic techniques is that their performance is typically affected by the choice of suitable values for their parameters. In this regard, a critical advantage of this method is its simplicity: the cuckoo search algorithm requires only two parameters, many fewer than any other metaheuristic approach, so the parameter tuning becomes a very easy task. In particular, the cuckoo search requires only two parameters: the population size N_p, and the probability p_a. In this paper, we consider a population of $N_p = 100$ host nests, representing the number of candidate solutions for the method. Regarding the parameter p_a, our choice is completely empirical: we carried out some simulations for different values of this parameter, and found that the values around $p_a = 0.2$ reduce the number of iterations required for convergence, so this is the value taken in this paper. However, we noticed that the method is not strongly sensitive to variations of the p_a for the problem in this paper.

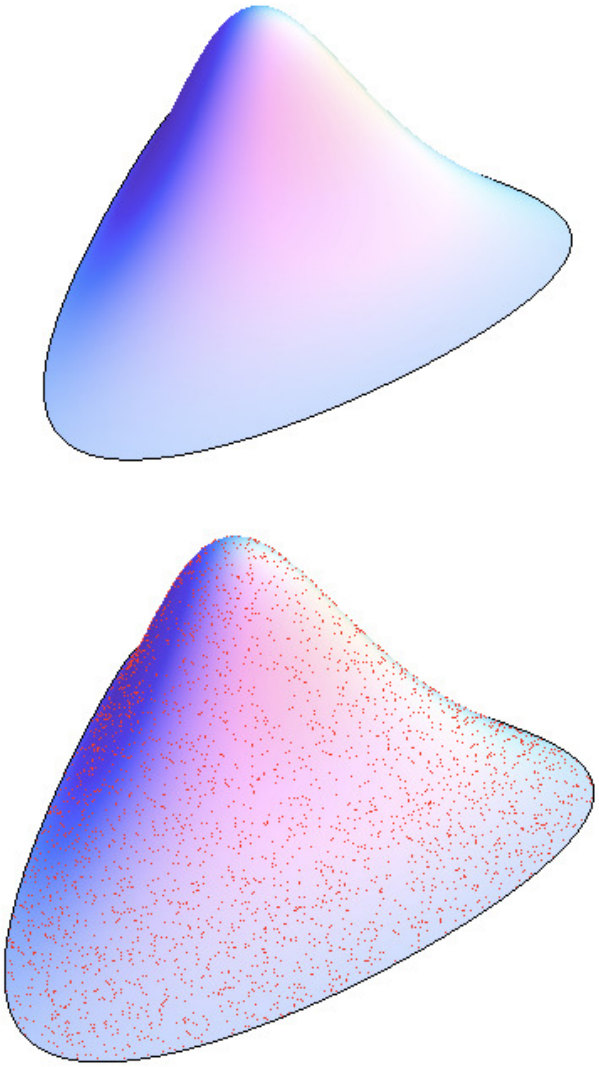

Figure 4: *Example II*: (top) Reconstructed surface; (bottom) reconstructed surface and data points.

4.4 Implementation issues

The computations in this paper have been carried out on a PC desktop with a processor Intel Core i9 running at 3.7 GHz and with 64 GB of RAM. The source code has been implemented by the authors in the programming language of the scientific program *Mathematica* version 12. About the computational times, our method is quite fast. Each execution of the method takes only a few seconds of CPU time, including rendering.

5 EXPERIMENTAL RESULTS

The method described in the previous section has been applied to several instances of point clouds. For limitations of space, this section restricts the discussion to three illustrative examples, fitted according to the model in Eq. (2) for *Example I*, and the model in Eq. (4) for *Example II* and *Example III*. They are discussed in the following paragraphs.

5.1 Example I

This example consists of a cloud of $R = 34,863$ three-dimensional data points displayed in Fig. 1. The data points do follow a uniform parameterization and are affected by white noise of low intensity (SNR=12). The point cloud is fitted according to Eq. (2) with the constraints in Eq. (3). Therefore, the resulting optimization problem consists of minimizing the functional:

$$\Xi = \sum_{r=1}^{R} \left[(x_r - \hat{x_r})^2 + (y_r - \hat{y_r})^2 + (z_r - f(\hat{x_r}, \hat{y_r}))^2 \right]$$

for

$$f(\hat{x_r}, \hat{y_r}) = e^{F + A\hat{y_r} - C\hat{y_r}^2 + (G + B\hat{y_r} - D\hat{y_r}^2)\hat{x_r} + (H + J\hat{y_r} - K\hat{y_r}^2) log(\hat{x_r})}$$

Applying our method to the minimization of the functional Ξ with the constraints in Eq. (3) leads to the values: $A = 3.0168; B = 1.9431; C = 1.0028; D = 2.9857; K = -2.0193; F = 3.9938; G = -0.9819; H = 2.0062; I = 1.9896$. For these values, the mean squared error (MSE), denoted in this paper as Δ and defined as $\Delta = \frac{\Xi}{R}$, becomes: $\Delta = 0.028816$, which shows that the method is very accurate in recovering the underlying mathematical structure of the data.

5.2 Example II

The second example consists of a cloud of $R = 23,714$ three-dimensional data points shown in Fig. 3. The data points are affected by the same conditions as the previous example, and are is fitted according to Eq. (4) with the constraints in Eq. (5). Therefore, the resulting optimization problem consists of minimizing the functional:

$$\Xi = \sum_{r=1}^{R} \left[(x_r - \hat{x_r})^2 + (y_r - \hat{y_r})^2 + (z_r - f(\hat{x_r}, \hat{y_r}))^2 \right]$$

for

$$f(\hat{x_r}, \hat{y_r}) = e^{C_0 + \hat{x_r} C_2 + \hat{y_r} C_6 + \hat{x_r}\,\hat{y_r} C_8}\, \hat{x_r}^{C_1 + \hat{y_r} C_7}\, \hat{y_r}^{C_3 + \hat{x_r} C_3 + C_4 \log(\hat{x_r})}$$

Application of our method yields the values: $C_0 = -0.1347$; $C_1 = 0.9973; C_2 = -2.0059; C_3 = 2.9967; C_6 = -1.0132$, for which the mean squared error takes the value: $\Delta = 0.00974$, which is considered a very good approximation.

Fig. 4 shows the optimal reconstructed surface (top) and its superposition with the point cloud (bottom). Note again the excellent visual quality of the surface reconstruction from the point cloud in Fig. 3, which confirms our good numerical results.

5.3 Example III

For the third example, we consider the cloud point depicted in Fig. 5. In this case, the point cloud consists of $R = 27,261$ data points, which is fitted according to Eq. (4) with the constraints in Eq. (6).

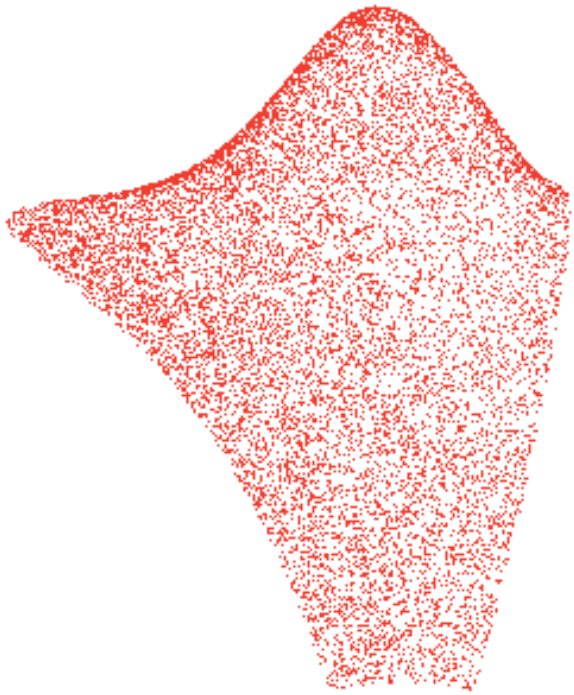

Figure 5: Original point cloud of *Example III*.

Application of our method yields the values: $C_0 = 1.0029; C_1 = 2.9726; C_2 = -1.9850; C_3 = 1.0603; C_6 = -1.0474; C_8 = -0.9806$, for which the mean squared error takes the value: $\Delta = 0.05156$, a satisfactory indicator of a good approximation. The resulting best approximating surface is shown in Fig. 6 (top) and superimposed by the original point cloud in Fig. 6 (bottom).

6 CONCLUSIONS AND FUTURE WORK

This paper addresses the surface reconstruction problem from point clouds of 3D data points obtained by reverse engineering processes from a manufactured workpiece. In this work we assume that the point cloud can be approximated by surfaces represented mathematically as a combination of exponential, polynomial, and logarithmic functions and subjected to several constraints. The resulting nonlinear constrained continuous minimization problem is solved by using a metaheuristic method called cuckoo search algorithm with Lévy flights. The application of this method to a benchmark of three examples of point clouds shows that the method is able to provide good visual and numerical results. We conclude that the method is promising towards its application to manufactured workpieces in real industrial settings.

In spite of these encouraging results, the method can still be further extended in several ways. On the one hand, one the limitations of this work is that the models used to fit the point clouds have been manually selected by the authors, based on their previous experience when working with several point clouds. However, an automatic procedure to select the most suitable model would be

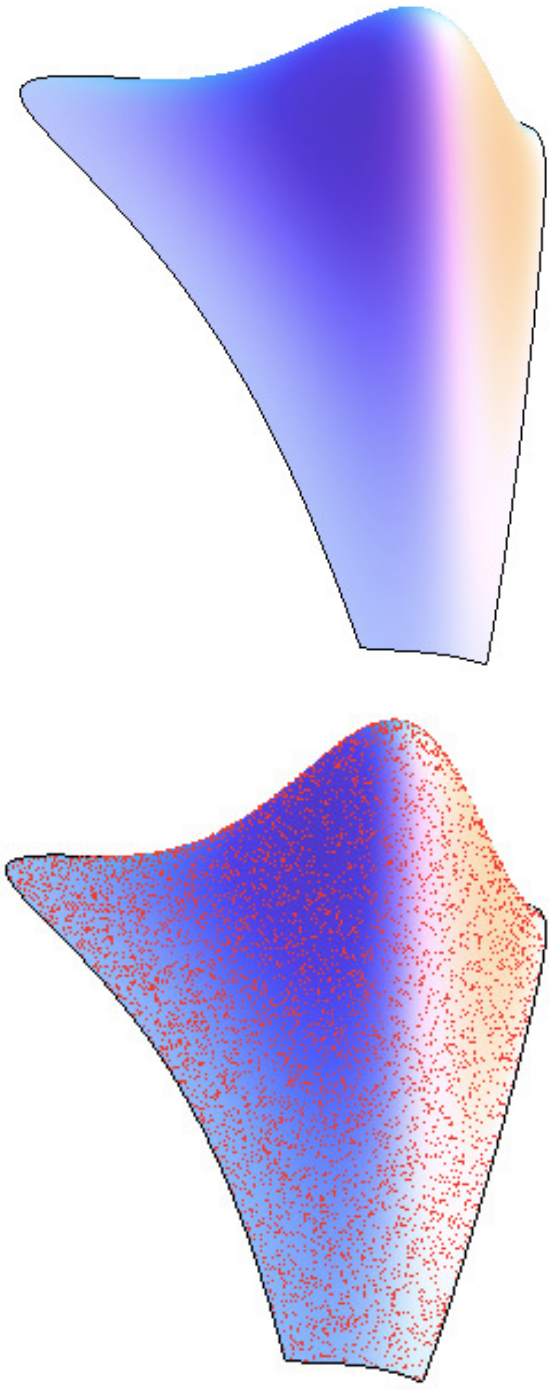

Figure 6: *Example III*: **(top) Reconstructed surface; (bottom) reconstructed surface and data points.**

a great enhancement for this method. On the other hand, we also want to extend this method to other families of basis functions. Finally, we want to apply this methodology to workpieces from different manufacturing industries subjected to other types of functional constraints. This will be part of our plans for future work in the field.

ACKNOWLEDGMENTS

Akemi Gálvez and Andrés Iglesias thank the financial support from the project PDE-GIR of the European Union's Horizon 2020 research and innovation programme, in the Marie Sklodowska-Curie Actions (MSCA) programme, with grant agreement of reference number H2020-MSCA-RISE-2017-778035, and also from the Agencia Estatal de Investigación (AEI) of the Spanish Ministry of Science and Innovation, for the grant of the Computer Science National Program with reference number PID2021-127073OB-I00 of the MCIN/AEI/10.13039/501100011033/FEDER, EU. Iztok Fister thanks the Slovenian Research Agency for the financial support under Research Core Funding No. P2-0042 - Digital twin. Iztok Fister Jr. thanks the Slovenian Research Agency for the financial support under Research Core Funding No. P2-0057.

REFERENCES

[1] Arnold, B., Castillo, E., Sarabia, J.M.: *Conditionally Specified Distributions*. Springer Verlag, Lecture Notes in Statistics, **73**, Berlin, Germany (1992).
[2] Barhak, J., Fischer, A.: Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques. *IEEE Trans. on Visualization and Computer Graphics*, **7**(1), 1–16 (2001).
[3] Barnhill, R.E.: *Geometric Processing for Design and Manufacturing*. SIAM, Philadelphia (1992).
[4] Castillo, E., Iglesias, A.: Some characterizations of families of surfaces using functional equations. *ACM Transactions on Graphics*, **16**(3), 296–318 (1997).
[5] Castillo, E., Iglesias, A., Ruiz, R.: *Functional Equations in Applied Sciences*. Elsevier Science, Mathematics in Science and Engineering, **199**, Amsterdam, The Netherlands (2004).
[6] Dierckx, P.: *Curve and Surface Fitting with Splines*. Oxford University Press, Oxford (1993).
[7] Farin, G.: *Curves and surfaces for CAGD (5th ed.)*. Morgan Kaufmann, San Francisco (2002).
[8] Gálvez, A., Iglesias, A., Cobo, A., Puig-Pey, J., Espinola, J.: Bézier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least-squares approximation. *Lectures Notes in Computer Science*, **4706**, 680–693 (2007).
[9] Gálvez, A., Iglesias A.: Efficient particle swarm optimization approach for data fitting with free knot B-splines. *Computer-Aided Design*, **43**(12), 1683–1692 (2011).
[10] Gálvez, A., Iglesias A., Puig-Pey J.: Iterative two-step genetic-algorithm method for efficient polynomial B-spline surface reconstruction. *Information Sciences*, **182**(1), 56–76 (2012).
[11] Gálvez A., Iglesias A.: Particle swarm optimization for non-uniform rational B-spline surface reconstruction from clouds of 3D data points. *Information Sciences*, **192**(1), 174–192 (2012).
[12] Gálvez A., Iglesias A.: A new iterative mutually-coupled hybrid GA-PSO approach for curve fitting in manufacturing. *Applied Soft Computing*, **13**(3), 1491–1504 (2013).
[13] Gálvez A., Iglesias A., Avila, A., Otero, C., Arias, R., Manchado, C.: Elitist clonal selection algorithm for optimal choice of free knots in B-spline data fitting. *Applied Soft Computing*, **26**, 90–106 (2015).
[14] Gu, P., Yan, X.: Neural network approach to the reconstruction of free-form surfaces for reverse engineering. *Computer-Aided Design* **27**(1), 59–64 (1995).
[15] Hoffmann, M.: Numerical control of Kohonen neural network for scattered data approximation. *Numerical Algorithms*, **39**, 175–186 (2005).
[16] Iglesias, A., Echevarría, G., Gálvez, A.: Functional networks for B-spline surface reconstruction. *Future Generation Computer Systems*, **20**(8), 1337–1353 (2004).
[17] Iglesias, A., Gálvez, A.: Hybrid functional-neural approach for surface reconstruction. *Mathematical Problems in Engineering*, Article ID 351648, 13 pages (2014).
[18] Iglesias, A., Gálvez, A.: Memetic electromagnetism algorithm for surface reconstruction with rational bivariate Bernstein basis functions. *Natural Computing*, **16**, 511–525 (2017).
[19] Iglesias, A., Gálvez, A., Avila, A.: Immunological approach for full NURBS reconstruction of outline curves from noisy data points in medical imaging. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **15** (6), 929–1942 (2018).

[20] Iglesias, A., Gálvez, A., Collantes, M.: Multilayer embedded bat algorithm for B-spline curve reconstruction. *Integrated Computer-Aided Engineering*, **24**(4), 385–399 (2017).
[21] Iglesias, A., Gálvez, A., Suárez, P., Shinya, M., Yoshida, N., Otero, C., Manchado, C., Gómez-Jauregui, V.: Cuckoo search algorithm with Lévy flights for global-support parametric surface approximation in reverse engineering. *Symmetry*, 10, Paper 58 (2018).
[22] Jing, L., Sun, L.: Fitting B-spline curves by least squares support vector machines. In: *Proc. of the 2nd. Int. Conf. on Neural Networks & Brain*. Beijing (China). IEEE Press, 905–909 (2005).
[23] Jupp, D.L.B.: Approximation to data by splines with free knots. *SIAM Journal of Numerical Analysis*, **15**, 328–343 (1978).
[24] Knopf, G.K., Kofman, J.: Adaptive reconstruction of free-form surfaces using Bernstein basis function networks. *Engineering Applications of Artificial Intelligence*, **14**(5), 577–588 (2001).
[25] Ma, W.Y., Kruth, J.P.: Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces. *Computer-Aided Design*, **27**(9), 663–675 (1995).
[26] Park, H.: An error-bounded approximate method for representing planar curves in B-splines. *Computer Aided Geometric Design* **21**, 479–497 (2004).
[27] Park, H., Lee, J.H.: B-spline curve fitting based on adaptive curve refinement using dominant points. *Computer-Aided Design* **39**, 439–451 (2007).
[28] Patrikalakis, N.M., Maekawa, T.: *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer Verlag, Heidelberg (2002).
[29] Pottmann, H., Leopoldseder, S., Hofer, M., Steiner, T., Wang, W.: Industrial geometry: recent advances and applications in CAD. *Computer-Aided Design*, **37**, 751–766 (2005).
[30] Powell, M.J.D.: Curve fitting by splines in one variable. In: Hayes, J.G. (editor): *Numerical approximation to functions and data*. Athlone Press, London (1970).
[31] Rice, J.R.: *The Approximation of Functions. Vol. 2.* Addison-Wesley, Reading, MA (1969).
[32] Sarfraz, M., Raza, S.A.: Capturing outline of fonts using genetic algorithms and splines. *Proc. of Fifth International Conference on Information Visualization IV'2001*, IEEE Computer Society Press,738–743 (2001).
[33] Varady, T., Martin, R.: Reverse Engineering. In: Farin, G., Hoschek, J., Kim, M. (eds.): *Handbook of Computer Aided Geometric Design*. Elsevier Science (2002).
[34] Wang, W.P., Pottmann, H., Liu, Y.: Fitting B-spline curves to point clouds by curvature-based squared distance minimization. *ACM Transactions on Graphics*, **25**(2), 214–238 (2006).
[35] Yang, X.-S.: *Nature-Inspired Metaheuristic Algorithms (2nd. Edition)*. Luniver Press, Frome, UK (2010).
[36] Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. In: *Proc. World Congress on Nature & Biologically Inspired Computing (NaBIC)*. IEEE, 210–214 (2009).
[37] Yang, X.S., Deb, S.: Engineering optimization by cuckoo search. *Int. J. Mathematical Modelling and Numerical Optimization*, **1**(4), 330–343 (2010).
[38] Yoshimoto, F., Moriyama, M., Harada, T.: Automatic knot adjustment by a genetic algorithm for data fitting with a spline. *Proc. of Shape Modeling International'99*, IEEE Computer Society Press, 162–169 (1999).
[39] Yoshimoto, F., Harada T., Yoshimoto, Y.: Data fitting with a spline using a real-coded algorithm. *Computer-Aided Design*, **35**, 751–760 (2003).
[40] Zhao, X., Zhang, C., Yang, B., Li, P.: Adaptive knot adjustment using a GMM-based continuous optimization algorithm in B-spline curve approximation. *Computer-Aided Design*, **43**, 598–604 (2011).