Granular Matter
DOI 10.1007/s10035-012-0357-9

ORIGINAL PAPER

Optimal description of two-dimensional complex-shaped objects using spheropolygons

Peter B. Dobrohotoff · Syed Imranullah Azeezullah · Federico Maggi · Fernando Alonso-Marroquin

Received: 16 January 2012

Abstract Recent advances in the morphological description of particles in granular material systems allow two-dimensional complex-shaped particles to be realistically simulated using spheropolygons, i.e., the Minkowski sum of a disk and a polygon. For identical numbers of vertices, spheropolygons achieve a better description of shapes than polygons, but require that the optimal spheroradius be determined. Here we propose a method for generating spheropolygons that optimizes the description of particle morphologies, i.e., minimizes the error images and the numbers of vertices. Because the error images of individual particles are a proxy for the accuracy of granular matter flow calculations, while the numbers of vertices are a proxy for the computational time, the method is optimally applicable to discrete element methods. We demonstrate the proposed method using pebbles, gravel, and crushed shells.

Keywords Discrete element modelling · Polygon · Spheropolygon

1 Introduction

Granular materials are recurring media in geophysics [7] and industrial processes [14]. In these processes, the flow of granular materials involves highly nonlinear dynamics that are governed by particle interactions at the microscopic scale. These interactions depend on various factors including the shapes of the particles, and therefore their correct description is crucial to the modelling of the flow with computer simulations.

The idea of using two-dimensional spheropolygons and three-dimensional spheropolyhedrons to model complex shapes dates from the work of Pournin and Liebling [16–19], who presented a method in three dimensions to calculate particle interactions based on a single contact between each pair of particles; the method was subsequently developed to efficiently handle multiple contacts [1,2,8–11] and has become a computationally efficient technique in numerical simulations of granular materials.

A spheropolygon is the Minkowski sum of a base polygon with N vertices, representing the general shape (e.g., convexity, jointness, and branchness), and a disk with radius r (called the spheroradius), representing the roundness. Formally, the Minkowski sum of two sets of points P and Q is given by

$$P + Q = \{\mathbf{u} + \mathbf{v} \mid \mathbf{u} \in P,\ \mathbf{v} \in Q\}. \tag{1}$$

This is a commutative operation geometrically equivalent to the sweeping of one set around the profile of the other without changing the relative orientation. For example, the image of the cow in Fig. 1a can be approximated by the spheropolygon in Fig. 1b. The coordinates of the vertices and the spheroradius are normalized by the length of the spheropolygon.

A variety of shapes can be generated for any number of dimensions using spheropolygons, spheropolyhedra, and spheropolytopes. The method is particularly useful for dynamic simulation of flow of granular materials under load, or gravity-driven granular flows of complex-shaped particles such as rice or capsules (a disk swept on straight lines) [5],

P. B. Dobrohotoff · S. I. Azeezullah · F. Maggi · F. Alonso-Marroquin (✉)
School of Civil Engineering, The University of Sydney, Sydney, NSW 2006, Australia
e-mail: fernando.alonso@sydney.edu.au

P. B. Dobrohotoff
e-mail: p.dobrohotoff@physics.usyd.edu.au

Published online: 09 June 2012

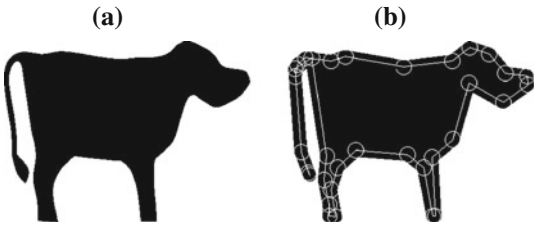

Fig. 1 **a** Test input image and **b** corresponding spheropolygon with 34 vertices (indicated by *circles*) and a (normalized) spheroradius of 0.02

peanuts (a disk swept on "V" shapes), pebbles (a disk swept on polygons) [2], Voronoi spheropolygons [19], and Voronoi spheropolyhedra [11].

Spheropolygons allow the representation of complex shapes, from rounded to angular and from convex to non-convex, that can be effectively simulated without introducing the roughness of boundaries associated with clumped particle representations. In fact, the Minkowski sum does not need to be explicitly calculated during the simulation to determine the characteristics of the particle interactions, which makes the use of spheropolygons in granular material simulations an efficient technique for calculating inter-particle forces.

Two key parameters that determine the suitability of a particular spheropolygon are its number of vertices N and the spheroradius r of each. The challenge is to find optimal combinations of N and r to describe complex shapes that allow the contact forces between them to be calculated efficiently, minimizing the CPU calculation time of a simulation. This paper addresses that challenge and presents a new algorithm to model complex shapes based on spheropolygons.

2 Methods

2.1 Optimization technique

A spheropolygon for a particle can be generated by "eyeballing" the particle image and placing the initial vertices by hand with an appropriate spheroradius. But such a process is laborious, and does not ensure that the resulting spheropolygon optimizes either spheroradius or number of vertices.

Spheropolygons can alternatively be constructed as shown in Algorithm 1. This is an iterative algorithm that minimizes the difference between the input image and the image of the spheropolygon by counting the number of error pixels.

To implement this algorithm, we first define three mathematical quantities:

- **Input image**, that is effectively a Boolean matrix with h rows and w columns. Each element of the matrix is set to either 1 (for a black pixel) or 0 (for a white pixel).

```
generate test spheropolygon;
while neither residual error nor maximum number of iterations
has been reached do
    for each vertex do
        sample V =number of error pixels in x and y dimension;
        move the vertex to the local minimum in V in each
        dimension;
    end
    generate test spheropolygon;
end
```

Algorithm 1: Construction of a spheropolygon from an image.

- **Spheropolygon**, that is a $(2N + 1)$-vector

$$SP = (x_1, y_1, x_2, y_2, \ldots, x_N, y_N, r) \tag{2}$$

  in the Euclidean space $\mathbb{R}^{2N+1}$. The components (x_i, y_i) are the positions of N vertices, and r is the spheroradius of the spheropolygon;
- **Image of spheropolygon**, that is defined by the function $I_{SP} = f(SP)$, whose argument is the spheropolygon and whose value is a Boolean matrix of size $h \times w$ representing its image. To construct the image, we first set a square mesh of $h \times w$ nodes containing the spheropolygon. Then the Boolean matrix is constructed by assigning black (one) to the nodes inside of the spheropolygon, and white (zero) to the nodes outside it.

Using these definitions, we articulate the details of Algorithm 1 as follows:

1. Generate a test spheropolygon.
2. Generate the image of the resulting spheropolygon $I_{SP} = f(SP)$.
3. Obtain the absolute difference between the original image I_0 and the image of the test spheropolygon I_{SP}, and count the number of error pixels V:

$$V = |I_{SP} - I_0|\,; \tag{3}$$

4. For each vertex i of coordinates (x_i, y_i), calculate the number of error pixels resulting from incrementally moving the vertex in the plus and minus x and y directions from $-r_s$ to r_s. This is done by sampling $V_i(\Delta x, \Delta y)$, where

$$\Delta x, \Delta y = 0, \pm\frac{1}{n_s}r_s, \ldots, \pm\frac{n_s - 1}{n_s}r_s, \pm r_s \tag{4}$$

   with radius r_s and $2n_s + 1$ samples in each dimension. Δx and Δy define the offset of the sampling points from the vertex, and $V_i(\Delta x, \Delta y)$ represents the error pixels between the input image I_0 and the image of

the spheropolygon $SP' = (x_1, y_1, \ldots, x_i + \Delta x, y_i + \Delta y, \ldots, x_N, y_N, r)$

5. For each dimension, if a "local" minimum in $V_i(\Delta x, \Delta y)$ is found, i.e., the minimum is not at the edge of the sampling range, and $\Delta V > \Delta V_{cutoff}$ for some ΔV_{cutoff}, move the vertex by the position of the minimum in that dimension. For each of these criteria that is not satisfied (i.e., the minimum is at the edge), multiply the offset Δx or Δy by a weighting $0 < g < 1$ before moving the vertex. The purpose of the weighting factor is to reduce the impact of instabilities which may arise for poorly resolved local minima;[1]
6. Return to step 2, iterating until V is within a tolerance specified as a small percentage of the total number of pixels in the original image, or the maximum number of iterations has been reached.

2.2 Computational efficiency

The efficiency of a discrete element simulation that includes contact forces only is measured by its Cundall number $c = N_T \times N_P / T_{CPU}$, where N_T is the number of time steps, N_P is the number of particles, and T_{CPU} is the CPU time of the simulation [2]. The Cundall number represents the product of number of particles and number of time steps, per second; larger c means faster simulations. Earlier simulations of granular flow using spheropolygons included a lookup table of nearest neighbours to improve efficiency [1]. Benchmarks of simulations with various numbers of spheropolygon vertices N showed that the Cundall number scales as $c \sim 1/N$ [10]. Thus the CPU time of the simulation scales as $T_{CPU} \sim N \times N_T \times N_P$. The number of time steps is given by $N_T = T/\Delta t$, where T is the simulation time and Δt is the time step. For constant T and N_P the CPU time scales linearly as the number of vertices in the spheropolygons:

$$T_{CPU} \sim \frac{N}{\Delta t}. \tag{5}$$

Interaction between the spheropolygons is based on distances between their corresponding polygons, that must be always positive. We will see that this condition imposes a relationship between the size of time step and the spheroradius of the particle. To guarantee numerical stability, the time step is chosen to be $\Delta t = 2\pi a\sqrt{m/k_n}$, where m is the mass of the lightest particle, k_n the normal stiffness, and a is a constant that depends on the numerical integration scheme [13]. To avoid unrealistic contact interactions, we impose the constraint that the distance between the polygons should be less than the minimal spheroradius r. This can be guaranteed if one chooses a normal stiffness large enough to guarantee $k_n \times r = F_{max}$, where F_{max} is the maximal force a particle can experience. These two equations lead to

$$\Delta t = 2\pi a\sqrt{\frac{m \times r}{F_{max}}}. \tag{6}$$

Introducing Eq. (6) into Eq. (5) and assuming m, a, and F_{max} are constant, we obtain the relationship between CPU time, number of vertices, and spheroradius,

$$T_{CPU} \sim \frac{N}{\sqrt{r}}. \tag{7}$$

This equation shows that an approximation to the morphology of a real particle must be sought in such a way that it minimizes the number of vertices N and maximizes the spheroradius r.

3 Results

3.1 Approximation to a cow

We tested Algorithm 1 using a cow schematic as a test particle, (Fig. 1a). This shape was challenging as it included rounded and angular morphologies, and an extended feature (the tail) that is difficult to describe mathematically. The test image was used to analyze the sensitivity of the method to the initial number, positions, and spheroradii of the vertices.

The results of tests using the cow image with three different initial conditions, 35 vertices, and a normalized spheroradius of 0.02 are shown in Fig. 2. Starting with the vertices arranged in a circle (a) produced the greatest initial error and slowest convergence. Equally spacing the vertices on the boundary gave a much smaller initial error and faster convergence (b), and having these vertices most closely spaced where the curvature was highest (c) gave the best result. We used method (c) for the remaining tests in this paper.

Spheropolygons were also generated for a range of spheroradii ($0.01 \leq r \leq 0.1$) and number of vertices ($1 \leq N \leq 60$). A sample of corresponding output images are shown in Fig. 3. The contours of the error V between the images are shown in Fig. 4a. The smallest errors were achieved with a large number of vertices at small spheroradius, but beyond a certain threshold (dependent on spheroradius), adding more vertices did not substantially decrease the final error. Consider, for example, the 2 % contour in Fig. 4a. For $r \approx 0.02$, the error was approximately constant for $N > 30$. We note that although the process for generating spheropolygons is nonlinear, the method proved to be robust and produced acceptable model shapes given the constraints on r and N.

The error contours of Fig. 4a along with the CPU time given by Eq. (5) and Fig. 4b were used to obtain the (N, r)

[1] A weighting of $g = 0.5$ was used for all simulations in this paper.

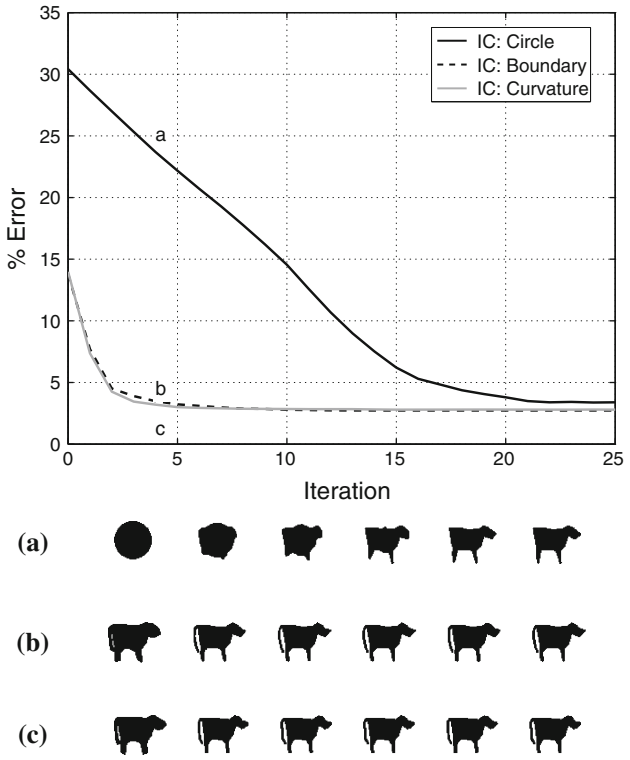

Fig. 2 Snapshots of convergence for various initial conditions for spheropolygons generated from the cow image using 35 vertices with spheroradius 0.02. Spheropolygons at iterations 0, 5, 10, 15, 20, and 25 are shown horizontally for initial conditions placing vertices **a** in a circle, **b** at the boundary, and **c** at maximum curvature

combination leading to the optimal cow for simulations. Since we need to take into account both error image and simulation time, we define the "O-variable" as

$$\mathrm{O} = t_{CPU} \times \mathrm{error}\,, \tag{8}$$

where $t_{CPU} = T_{CPU}/(N_P \times T)$.

The relationship between the O-variable, number of vertices, and spheroradius is shown in Fig. 4c. We identified a minimum at $(N, r) = (1, 0.1)$, and a local minimum at $(N, r) = (30, 0.01)$. The global minimum (not shown in the chart) is $(N, r) = (1, 0.56)$, which corresponds to a disk with the same area of the original image. This is the "spherical cow" representation. This minimum lies in the region with smallest simulation times but with a large error image. The local minimum corresponds to the best "non-spherical cow" representation. Using this representation leads to slower simulations, but most features of the shape, including the tail, are captured. The O-chart is useful to quantify the balance between accuracy of shape representation and simulation time, and to decide on the best shape representation for simulation.

3.2 Spheropolygons of real particles

We photographed pebbles, pieces of basalt gravel, and shell chips, and generated spheropolygons for each. These samples were chosen as they share similar size distribution properties while having different shape characteristics.

The size distribution for the samples is shown in Fig. 5. The basalt sample had the greatest range of sizes and the most even distribution. The pebbles had the least variable size. The size distributions were used to calculate the coefficient of uniformity C_u and coefficient of curvature C_c of the samples (Table 1). All samples classify as poorly-graded gravel (PG).

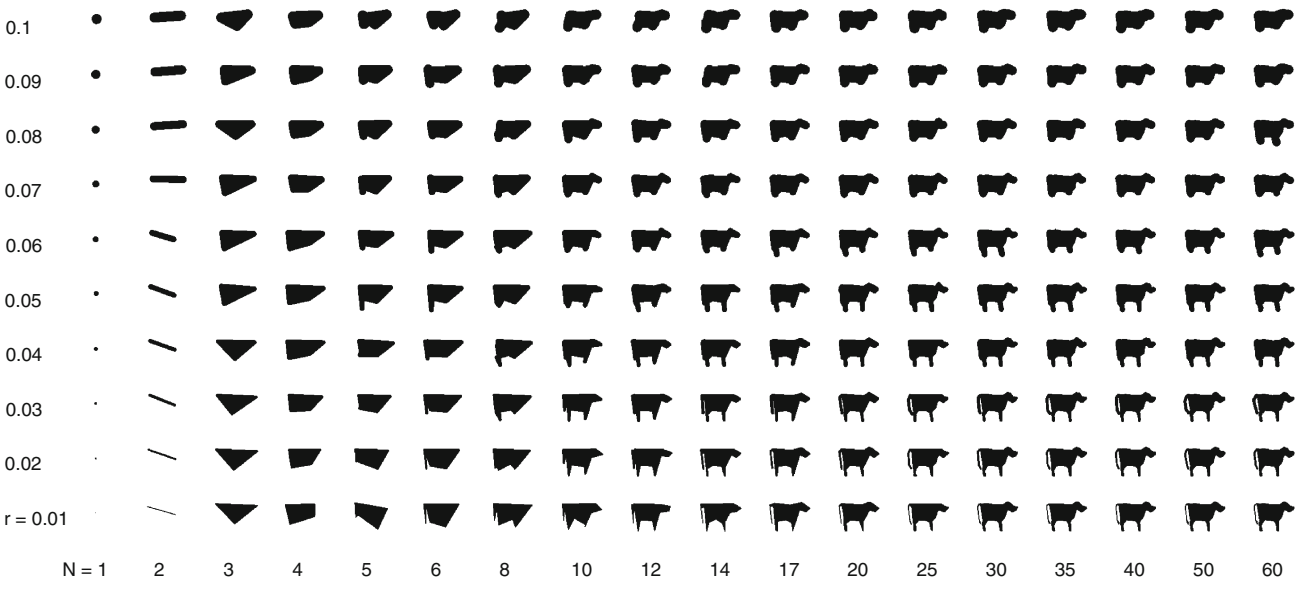

Fig. 3 A sample of spheropolygons generated for the cow image with normalized spheroradius r and number of spheropolygon vertices N

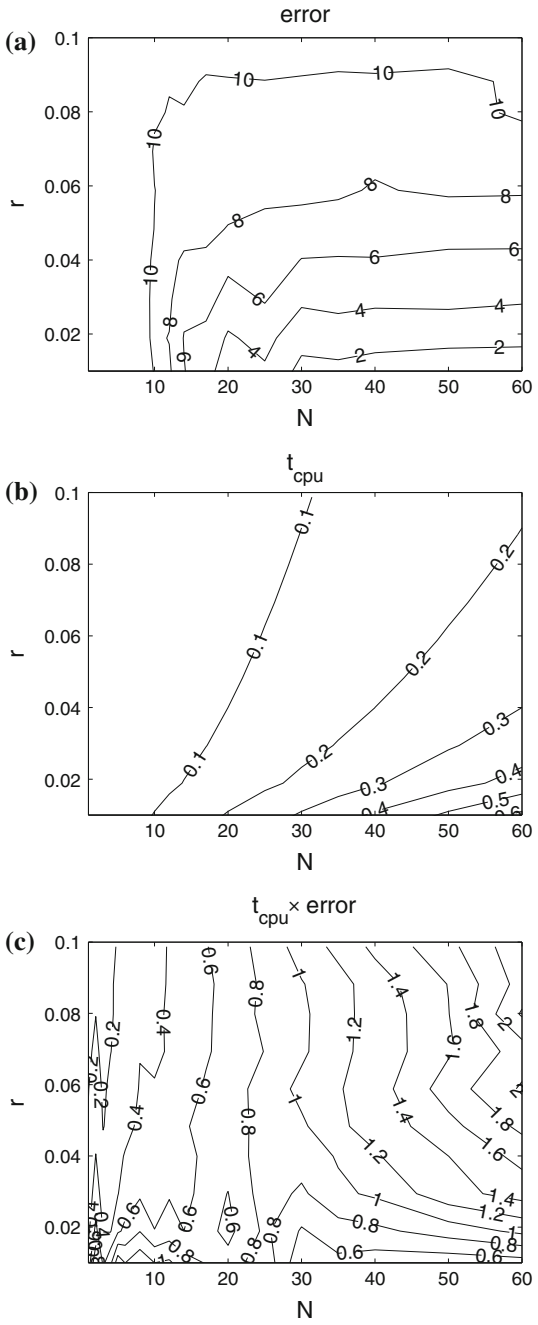

Fig. 4 Contour plot of percentage of error pixels **a**, normalized CPU-time **b**, and O-variable **c**, versus spheroradius r and number of vertices N. The error is % (number of error pixels)/(pixels of the original image); the normalized CPU time is a non-dimensional quantity calculated as $t_{CPU} = T_{CPU}/(N_P \times T)$; and the dimensionless O-variable is given by $\mathrm{O} = t_{CPU} \times \text{error}$

To obtain the shape distribution, we characterized the shape of each particle by its "roundness", which we defined as [4]

$$R \equiv \frac{4\pi A}{P^2}, \tag{9}$$

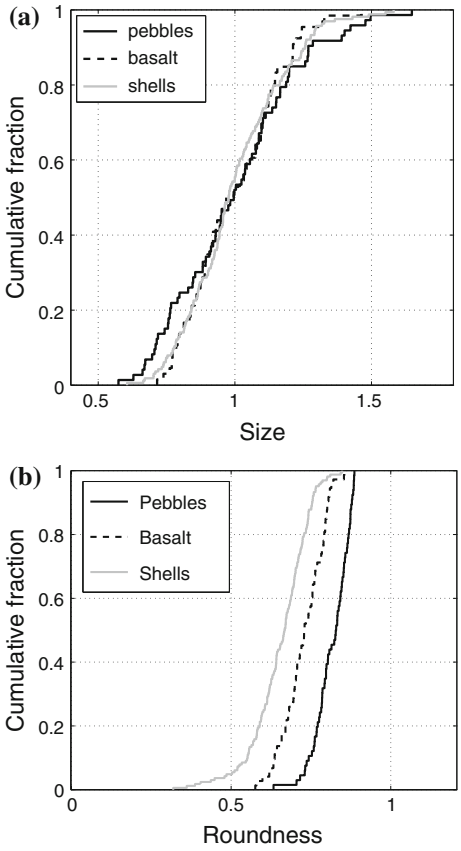

Fig. 5 **a** Size distributions for 66 pebbles, 73 pieces of basalt gravel, and 164 shells. The median of each sample was normalized to 1. **b** Shape distributions for 66 pebbles, 73 pieces of basalt gravel, and 164 shell chips

Table 1 Parameters describing shape and size distributions

Sample	Roundness	Size (mm)	C_u	C_c
Shells	0.67	7.27	1.30	1.02
Basalt	0.73	13.83	1.49	0.98
Pebbles	0.83	6.09	1.35	0.94

where A is the particle area and P the perimeter. Using this definition, $R = 1$ for a circle and $R = \pi/4$ for a square. For a rectangle with $p = w/h$ (the ratio of minor to major length), $R_{p\to 0} = 0$ and $R_{p\to 1} = \pi/4$; thus a square is the roundest rectangle. The shape distributions are show in Fig. 5. The pebbles were the most rounded particles and the least diverse. Crushed shell chips were the most angular, but also had the greatest range of roundness, the roundest chips being similar to pebbles. Basalt had an intermediate shape distribution.

Table 1 includes the median of the roundness distribution. Based on these values and the images of the grains, we classified pebbles as rounded grains, and basalt and shells as angular. The main difference between the shell and basalt is that the basalt particles are more convex.

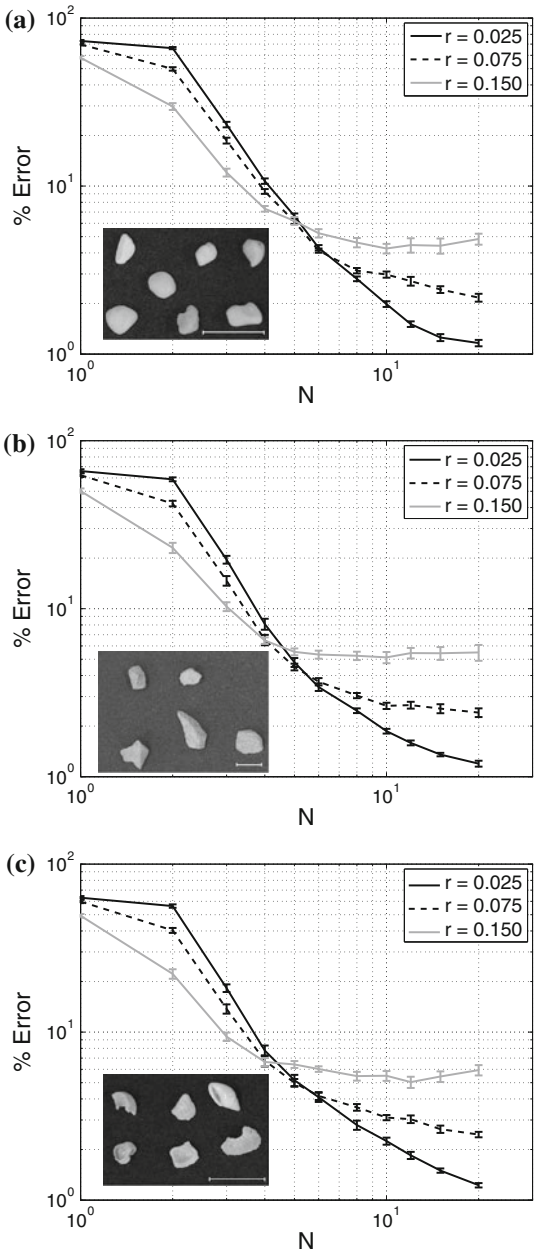

Fig. 6 Residual image errors for **a** pebbles, **b** basalt gravel, and **c** crushed shells. Each sample consisted of 25 particles. The standard error is indicated by *error bars*. *Insets* show photographs of samples. The *scale bar* in each image is 1 cm

Now we turn to the question of how the shape distribution affects the error of the approximation of the shape using spheropolygons. Each photograph (Fig. 6) was segmented into individual particles using the Hoshen–Kopelman algorithm [12] in MATLAB, and the method proposed in Sect. 2 to produce spheropolygons was applied to a random sub-sample of 25 segmented particles of each particle type for numbers of vertices and spheroradii in the ranges $1 \leq N \leq 20$ and $0.025 \leq r \leq 0.15$, respectively. The spheroradii were normalized relative to the median size of the complete samples.

The resulting errors for each particle in each sample were calculated using Eq. (3) and are shown in Fig. 6. For a small number of vertices, a large spheroradius produced the lowest error, while for a large number of vertices, a small spheroradius led to the lowest error. This pattern was found to be persistent for all types of particles analyzed here, and we presume it also holds for other morphologies. For each sample there was an intermediate number of vertices for which the error did not vary much with spheroradius.

4 Discussion

We have shown that by choosing as our initial conditions vertices at the image boundary concentrated in regions of maximum curvature, we can generate spheropolygons for a variety of real particles. Error pixels, Eq. (3), approach 1 % for $N \gtrsim 20$ and $r \lesssim 0.025$ across both angular and rounded particles. While the use of error pixels is only an indicator of the accuracy of a model representation (it may neglect details like tails, which include a relatively small number of pixels), it is useful to set a criterion for convergence towards an approximated solution.

Further testing of the parameter space for our scheme is planned to optimize accuracy and speed of execution. For example, when sampling the error potential for a vertex, it might not be necessary to generate complete spheropolygons for the entire image but to work on a subset of the image. This would produce significant speed-ups in the spheropolygon generation, and it may not degrade the final image appreciably.

We also note that since the optimal position of most vertices will be approximately one spheroradius inside the original object, it may be more efficient to start them there. Furthermore, it would be beneficial to add the constraint that each concave or convex region of the image should contain at least one vertex, preferably near the center of that region.

4.1 Number of vertices

Recall that the ultimate goal in this work was not just to produce model images that are close to real particles, but to produce simulations of granular flows. The efficiency of such simulations was inversely proportional to the number of vertices used to model the shapes, Eq. (7). While 60 vertices produced the most realistic cows (Fig. 3), 30 vertices still produced a shape that may behave similarly in simulations, and is probably a better choice. Similarly, we found that ≈ 6 vertices produced useful model shapes of basalt gravel, pebbles, and crushed shells. In general, the O-chart can be used to decide what the optimal shape is for a discrete element simulation that takes into account both shape accuracy and simulation time.

4.2 Spheroradius

It is desirable to produce model particles with as large a spheroradius as possible within the accuracy constraints. This is because in the simulations the spheroradius is closely connected with the stiffness of the contact interaction between particles: a small spheroradius requires high contact stiffness that leads to small time steps for stability, and therefore a long simulation time. A good choice for spheroradius is determined by the number of vertices chosen. For example, Fig. 6 indicates that for a small number of vertices the best choice was a large spheroradius, whereas for a high number of vertices the best choice was a small spheroradius. Note that at an intermediate number of vertices, the error did not vary much with spheroradius, so such a number might be a good choice for generating particles for real simulations.

The choice of spheroradius effectively sets the resolution of our modelling. Spatial scales smaller than the spheroradius are not modelled. For example, the tail on the cow in Fig. 3 disappears for a normalized spheroradius $\gtrsim 0.07$. Future work will investigate how modelling each vertex with its own spheroradius might improve the optimization. Other shape parameters might be worth investigating: for example spatial-frequency spectrum, or perimeter.

4.3 Limitations

Spheropolygons are suitable for descriptions of rounded shapes, but they lead to computationally expensive simulations if the particles are very angular. In these cases it is probably better to represent the shapes using polygons in two dimensions [20] or polyhedrons in three dimensions [6]. Packings of non-convex polyhedrons have been successfully modelled using Monte Carlo simulations on triangular meshes [6] or contact dynamics methods [3]. In molecular dynamics, deriving contact forces from potentials is difficult due to pathological cases that arise where the normal direction is not well defined [15]. This limitation is removed when using spheropolygons, because the normal direction is given in terms of the closest points between the features of the underlying polygons of the particles [1,10]; and similarly for spheropolyhedrons. It is recommended to use polygons if one is interested in packings of angular particles, and spheropolygons for simulations of dynamics in granular materials with relatively rounded particles.

5 Conclusion

We have demonstrated a method for constructing models of complex shapes using spheropolygons based on optimization of the image. The method is robust, producing shape representation for real particles with a variety of sizes and shapes, from rounded to angular and from convex to non-convex, with a percentage of error pixels below 2 %.

Using the so-called O-chart that relates error and computational time to number of vertices and spheroradius, we have further shown how the model shapes can be optimized to produce efficient numerical simulations of granular flows by a judicious choice of the number of vertices and spheroradii.

The scheme could be extended by relaxing the constraint that all vertices have the same spheroradius. This will increase the number of parameters to describe the particle shape from $2N + 1$ to $3N$, where N is the number of vertices. This extension will require modification of our algorithm by sampling the potential error function from the two-dimensional (x, y) plane to a three-dimensional (x, y, r) space. It should be remembered, however, that the purpose of generating spheropolygons is to use them in simulations, and that the stiffness of the contact interactions is determined by the smallest spheroradius. Introducing a variable spheroradius may therefore be implemented under the prescription of a specific criterion set on the maximum acceptable stiffness. This working hypothesis may be tested in future works.

Acknowledgments The authors want to thank Danilo Carluccio, the high-school student who helped us to develop the intuitive framework of the algorithm.

References

1. Alonso-Marroquin, F.: Spheropolygons: a new method to simulate conservative and dissipative interactions between 2d complex-shaped rigid bodies. EPL Europhys. Lett. **83**, 14001 (2008)
2. Alonso-Marroquín, F., Wang, Y.: An efficient algorithm for granular dynamics simulations with complex-shaped objects. Granul. Matter **11**(5), 317–329 (2009)
3. Azéma, E., Radjai, F., Saussine, G.: Quasistatic rheology, force transmission and fabric properties of a packing of irregular polyhedral particles. Mech. Mater. **41**(6), 729–741 (2009)
4. Burger, W., Burge, M.: Principles of Digital Image Processing: Core Algorithms, vol. 2. Springer, New York (2009)
5. Cruz-Hidalgo, R., Kadau, D., Kanzaki, T., Herrmann, H.: Granular packings of cohesive elongated particles. Arxiv preprint arXiv:1107.3040 (2011)
6. de Graaf, J., van Roij, R., Dijkstra, M.: Dense regular packings of irregular nonconvex particles. Phys. Rev. Lett. **107**(15), 155501 (2011)
7. Duran, J.: Sands, Powders, and Grains: An Introduction to the Physics of Granular Materials. Springer, Berlin (2000)
8. Galindo-Torres, S., Muñoz, J., Alonso-Marroquín, F.: Minkowski-voronoi diagrams as a method to generate random packings of spheropolygons for the simulation of soils. Phys. Rev. E **82**(5), 056713 (2010)
9. Galindo-Torres, S., Pedroso, D., Williams, D., Li, L.: Breaking processes in three-dimensional bonded granular materials with general shapes. Comput. Phys. Commun. **183**(2), 266–277 (2012). doi:10.1016/j.cpc.2011.10.001
10. Galindo-Torres, S.A., Alonso-Marroquín, F., Wang, Y.C., Pedroso, D., Muñoz Castaño, J.D.: Molecular dynamics simulation of complex particles in three dimensions and the study of friction due

to nonconvexity. Phys. Rev. E **79**(6), 060301 (2009). doi:10.1103/PhysRevE.79.060301
11. Galindo-Torres, S.A., Pedroso, D.M.: Molecular dynamics simulations of complex-shaped particles using voronoi-based spheropolyhedra. Phys. Rev. E **81**(6), 061303 (2010). doi:10.1103/PhysRevE.81.061303
12. Hoshen, J., Kopelman, R.: Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm. Phys. Rev. B **14**(8), 3438 (1976)
13. Luding, S., Herrmann, H.J., Blumen, A.: Simulations of two-dimensional arrays of beads under external vibrations: scaling behavior. Phys. Rev. E **50**, 3100–3108 (1994). doi:10.1103/PhysRevE.50.3100. http://link.aps.org/doi/10.1103/PhysRevE.50.3100
14. Nedderman, R.: Statics and Kinematics of Granular Materials. Cambridge University Press, Cambridge (2005)
15. Poeschel, T., Schwager, T.: Computational Granular Dynamics, chap. 2.4. pp. 61–65. Springer, Berlin (2004)
16. Pournin, L.: On the behavior of spherical and non-spherical grain assemblies, its modeling and numerical simulation. Ph.D. thesis, Lausanne (2005). doi:10.5075/epfl-thesis-3378. url:http://library.epfl.ch/theses/nr=3378
17. Pournin, L., Liebling, T.: A generalization of distinct element method to tridimensional particles with complex shapes. Powders Grains **2**, 1375–1378 (2005)
18. Pournin, L., Weber, M., Tsukahara, M., Ferrez, J.A., Ramaioli, M., Liebling, T.M.: Three-dimensional distinct element simulation of spherocylinder crystallization. Granul. Matter **7**, 119–126 (2005). doi:10.1007/s10035-004-0188-4
19. Ramaioli, M., Pournin, L., Liebling, T.M.: Vertical ordering of rods under vertical vibration. Phys. Rev. E **76**, 021304 (2007). doi:10.1103/PhysRevE.76.021304. url:http://link.aps.org/doi/10.1103/PhysRevE.76.021304
20. Tillemans, H.J., Herrmann, H.J.: Simulating deformations of granular solids under shear. Physica A **217**, 261–288 (1995)