Journal of Computational Physics 229 (2010) 3295–3315

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Using level sets for creating virtual random packs of non-spherical convex shapes

D. Shane Stafford [a], Thomas L. Jackson [b,*]

[a] Theoretical and Applied Mechanics, University of Illinois at Urbana-Champaign, United States
[b] Center for Simulation of Advanced Rockets at the University of Illinois at Urbana-Champaign, United States

ARTICLE INFO

Article history:
Received 16 January 2009
Received in revised form 22 November 2009
Accepted 5 January 2010
Available online 20 January 2010

Keywords:
Level sets
Random packs
Energetic materials
Sphere packing
Solid rockets
Propellants
Heterogeneous material
Porous material

ABSTRACT

Random packs of spheres have been used to model heterogeneous and porous material morphologies during simulations of physical processes such as burning of coal char, convective burning in porous explosives, and regression of solid rocket propellant. Sphere packs have also been used to predict thermo-mechanical properties, permeability, packing density, and dissolution characteristics of various materials. In this work, we have extended the Lubachevsky–Stillinger (LS) sphere packing algorithm to create polydisperse packs of non-spherical shapes for modeling heterogeneity in complex energetic materials such as HMX and pressed gun propellants. In the method, we represent the various particle shapes using level sets. The LS framework requires estimates of inter-particle collision times, and we predict these times by numerically solving a minimization problem. We have obtained results for dense random packs of various convex shapes such as cylinders, spherocylinders, and polyhedra, and we show results with these various particles packed together in a single pack to high packing fraction.

1. Introduction

Heterogeneous solid energetic materials are widely used in the aerospace and defense industries in rockets, explosives, and diverse pyrotechnic devices. The microstructure of these composites are often composed of particles of a crystalline oxidizer, such as ammonium perchlorate (AP) embedded in a polymer such as hydroxyl-terminated polybutadiene (HTPB) or polybutadiene acrylonitrile (PBAN). The polymers serve both as a binding agent and a fuel. Metal flake or powder, usually aluminum, may be added to increase the energetic content of the composite. Other composites may substitute crystals of an energetic material such as HMX for the oxidizer. Particle sizes for the composites range from a few μm for oxidizer particles in the "dirty binder" to 10s of μm for the metal flakes to 100s of μm for the largest oxidizer crystals. In explosives, the solids are loaded to mass fractions beyond 0.90, corresponding to volume fractions of 0.6–0.8. To achieve these high solids loadings, the materials are often compressed before the binder cures during manufacture. This results in particle breakage [1]. The final particle shapes are more polyhedral than spherical, but high aspect ratios seem to be rare [2]. Under conditions supporting deflagration, these composites burn with a reaction zone a few 100 μm thick. In certain scenarios, the deflagration can transition to a detonation [3]. Shown in Fig. 1 is a slice through an HMX-based explosive, illustrating the microstructure of a typical explosive. The HMX crystals range from a few μm to several hundred μm in size and are embedded in a rubbery binder.

* Corresponding author.
E-mail addresses: stafford.shane@gmail.com (D.S. Stafford), tlj@csar.uiuc.edu (T.L. Jackson).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.01.003

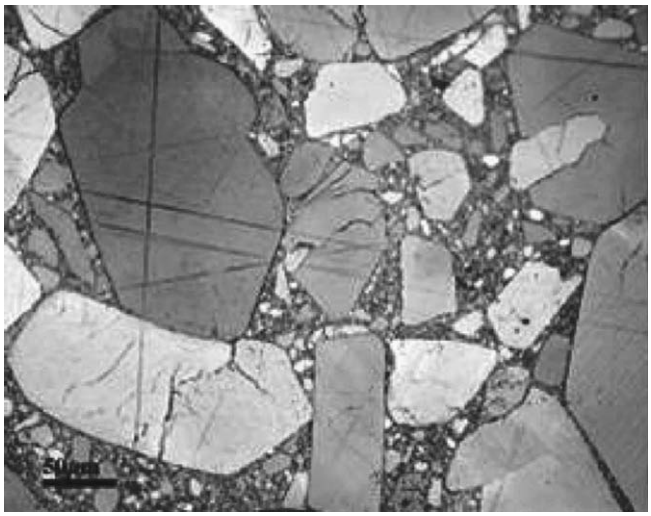

Fig. 1. A polarized light micrograph of a plastic bonded explosive (PBX) based on the energetic material HMX. The size scale is 50 μm. (Photo courtesy of C. Skidmore at LANL [43].)

Porous energetic materials are also common in the aerospace and defense industries, where the porosity is present either by choice, as in the case of rocket igniters, or by nature, as in the case of aging fractured explosives. These materials often also have heterogeneous microstructure, having been manufactured using the composites mentioned above. The grains in igniters and gun propellants are often characterized by a particular shape, such as perforated rods, spherocylinders, or prismatic stars, into which they have been pressed or cast. These grains are poured loosely or packed tightly into rigid containers such as gun shells. The size of propellant grains varies widely, from several 100s of μm to 10s of mm. Solids loading ranges from 50% to 70% by volume (the remainder is void space). The grains are often coated with deterrents that reduce the initial burn rate and enhance propellant performance [4–7]. Fractured energetic materials have significantly different properties than gun and igniter propellants, but they share some common porous material characteristics. The fractures develop over time as the explosive ages, increasing the porosity to the point that the overall device becomes unstable and unsafe. Because fractured explosive began life as solid composites, they typically exhibit much lower porosity than igniters and gun propellants, with volume fractions ranging from 0.7 to 1.0 [8]. In either case, these porous materials have macroscopic burn characteristics quite different than the solid composites [9].

Designers of devices using these materials are concerned with various properties of the materials as they relate to safety and engineering issues. These include thermo-mechanical properties such as Young's modulus, mechanical stability and thermal conductivity, as well as burn rate and metal agglomeration characteristics, among others. These bulk properties are strongly dependent on the morphology of the materials, and it is thus necessary to have a proper model for the morphology. Packs of disks and spheres have been extensively used for this purpose with much success. In particular the dependence of the burn rate on the morphology of composite AP/HTPB has been studied extensively using packs of spheres [10,11]. At least for burn rate studies, spheres appear to be a reasonable approximation to the shape of the particles, and a study of spheroids showed little dependence of burn rate on sphericity [12]. Likewise, much success has also been reported in aluminum agglomeration model development with packs of polydisperse spheres [13].

Nevertheless, spheres can be poor approximations to heterogeneous and porous materials when various other properties are of interest. Bulk material properties such as thermal conductivity [14] and elastic properties [15] strongly depend on the statistical details of the microstructure, and spheres do not properly replicate these statistics. Efforts to design a statistically optimal periodic unit cell as a model of the microstructure of a composite energetic material also depend on better approximations of the shape of the particulates [16].

Porous flow is another area where spheres may perform poorly as models of morphology. Understanding the fluid flow in porous energetic materials is crucial when designing explosive devices to ensure their proper operation and long term safety [6]. Fluid flow phenomena strongly affect burn rates in porous materials and may play a strong role in the detonation to deflagration transition (DDT) in energetic materials. DDT is an important topic for the safety and reliability of the nuclear stockpile [8]. As with the thermo-mechanical properties, fluid flow through porous energetic materials is also strongly dependent on the shape of the particle. For example, the well-known Ergun correlation for pressure drop through a packed bed is accurate for monodisperse spheres under many conditions and even works well for monodisperse packs of some non-spherical particles when the particle shape is adequately accounted for [17]. However, when used where the particles have large polydispersity or are moderately non-spherical, the Ergun relation fares poorly [18]. Other empirical correlations for the drag coefficient and permeability of a porous medium also work well for spheres, but fail for more complex structures. In one particularly relevant experiment, empirical relations for predicting the permeabilities of packed beds of spheres were found to over-predict the measured permeability of the non-spherical explosive CP by factors of 5–50 [19].

The latter two classes of problems demand a more accurate representation of the particle structure than can be provided by spheres. Thus, the major focus of this research was to devise a method for generating packs of arbitrary particle shapes with high packing fractions and large particle size ratios.

This article is organized as follows. First, in Section 2, we briefly describe *Rocpack*, our existing code for packing large numbers of polydisperse spheres. We provide further motivation for extending *Rocpack* to handle non-spherical shapes, and survey similar methods that have been presented in the literature. In Section 3, the new method for generating packs of arbitrary shapes is presented. The method builds upon the LS algorithm, introducing a flexible shape representation without changing the original LS framework. A few representative packs of various kinds of shapes are demonstrated, including some packs that resemble some real propellants, in Section 4. Comparisons are made to the equivalent packs of spheres and some higher-order statistics are shown. Finally, we summarize our conclusions and suggest further research in Section 5.

2. Background for the method

Packings of disks (in 2D) and spheres are the workhorses of the materials science community and have been used for many decades to study physical processes such as liquid flow, crystallizations, hardening, melting, granular flow, gasification, and propellant combustion, among many others [20–25]. The most obvious advantage of these shapes lies in the simplicity of their representation, which facilitates theoretical, experimental, and computational efforts in materials science.

As noted above, energetic materials tend to have low porosity, and so a key requirement of any sphere packing algorithm used in this context is the capability of generating packs with high packing fractions. In addition, one more constraint applies for our particular focus: the algorithm must handle polydispersity in an efficient manner. For propellant burn rate studies, the resolution of the numerical simulation of the combustion is a limiting factor, and for these computations packs of only up to a few 100 particles are sufficient. However, the packs will often contain large amounts of polydispersity, and the ratio of the size of the largest particle to the size of the smallest particle can (and often does) exceed 50:1. For concurrent packing methods, this can greatly increase the time required to generate a pack.

In light of these critical constraints, we adapted the Lubachevsky–Stillinger (LS) packing algorithm for spheres and have used it in our code *Rocpack* to successfully model energetic materials for almost a decade [11]. The LS algorithm begins by placing N spheres with zero initial radii (points) at random locations inside the domain of interest. The spheres are given random velocities $\boldsymbol{v}_i$ sampled from a Maxwell–Boltzmann distribution at temperature Θ_0 and are allowed to grow at a specified growth rate g_i. The particles undergo classical (super-)elastic collision dynamics as they grow to fill the space in the domain. The algorithm stops when either a specified density is reached or when a specified jamming criterion is met.

Modern implementations of rigid sphere packing take advantage of an event-driven molecular dynamics (EDMD) approach. In EDMD, particles are advanced between "events", where an event is loosely defined as anything that changes a particle's state. The event could be a binary collision, a collision between a particle and a domain boundary, or a transfer of a particle across an internal or external boundary. Instead of advancing the particles by a fixed time step as in time-driven MD (TDMD), the particles are always advanced to the next event time.

As an improvement to the LS algorithm, we use a hierarchical cell structure to drastically reduce the computation time when the particle set is very polydisperse. Cell schemes in general are well-known in the MD community, and a dynamic hierarchical cell variant has been implemented in two dimensions by Wackenhut et al. [26]. However, we are not aware of any other implementation of a fully three-dimensional multi-level cell method. The hierarchical cell scheme allows *Rocpack* to quickly create packs of hundreds of thousands of spheres with size ratios of up to 100:1 on a laptop in a matter of hours. Details of the algorithm are presented in Refs. [27,28] along with validation of the sphere packs and statistical analyses of the results.

The focus of the current article is a recent extension to *Rocpack*: a new packing method for arbitrary convex shapes that can achieve high packing fractions without introducing artificial ordering. The method can, in principle, be applied to pack any set of shapes at any size ratio, provided one can first calculate the signed minimum distance field for each shape. Because the new method is still based on the LS algorithm, it leverages two important properties of the LS method: (1) capability for very high packing fractions, and (2) excellent statistical properties [27].

In the discussion that follows, we will compare various methods for representing the arbitrary shapes in a pack, including other methods for packing arbitrary shapes that are discussed in the literature. Note that *any* of the methods discussed below could be adapted for use within the LS framework. To reiterate, the major desirable properties of the overall method when used to model energetic materials are the following: (1) it should be able to generate packs with high packing fractions $\rho > 0.6$, (2) it should efficiently handle large differences in particle sizes (polydispersity), (3) it should reasonably reproduce the randomness present in real materials, and (4) it should be able to produce packs of $N > 100$ in a reasonable amount of time.

For the representations of the shapes themselves, we also have specific requirements, some which parallel the requirements listed above for the overall method. Our interest is primarily in packs that resemble real rocket propellant and explosive microstructures. These microstructures often consist of many different shapes and many different sizes bound together in a single composite. Some of the shapes may resemble polyhedra; others may be approximated by cylinders, etc.; and all of these must be packed together in the same pack. Thus, we would like to be able to specify the particle size distribution, the particle shape distribution, and the porosity so that we have the opportunity to accurately model these details of real microstructures.

Two other requirements for the shape representation stem from the use of the LS algorithm. First, the shape representation must be consistent or nearly consistent under any arbitrary affine transformation: that is, when we compute the distance to the shape from any point, the computation must yield results that are independent of orientation and translation to within a small tolerance. If there is too much error in the distance estimates, then the LS algorithm may fail. Thus, the shape representation need not be an exact, closed-form analytical relation. Numerical methods are also acceptable if they can maintain a small enough error in the distance estimate. This tolerance is difficult to determine *a priori* but easy to determine in the implementation of the method.

The second requirement that is due to the use of the LS algorithm is that the distance computation between any two particles must be a fairly efficient process. The LS algorithm is a concurrent algorithm, and as such, is not exceedingly fast. We will show in Section 3 that the LS framework will require numerically finding the root of the distance function between two shapes as they move in time. This root solve is already rather expensive, and adding the additional burden of a numerical distance estimate will increase CPU time. This is, however, the route we choose.

Thus, our requirements for a shape representation can be summarized. (1) The representation must be sufficiently generic to handle various shapes such as spheres, cylinders, and polyhedra. (2) Multiple shapes must be possible within a single pack. (3) We must be able to specify the shapes. (4) The representation must be consistent under affine transformations. (5) The representation must be reasonably efficient. Of course we may choose to trade efficiency to satisfy the other requirements.

There have been very few attempts to create packs of non-spherical shapes reported in the literature. Of the methods that have been published, there are three main classes of methods for representing the 3D shapes: methods for shapes that have convenient analytical representations, voxel based methods, and Voronoi methods. The first class of methods is very useful in cases where the desired particle shapes have clean analytical representations and there exists a tractable method for computing the distance between the shapes (either analytically or numerically). Examples of the use of such methods include packing ellipsoids to high density [29], packing super-ellipses (rounded rectangles) in 2D [30], and packing ellipses and spheroids at various orientations as a model of rocket propellant [12]. These examples are all based on the LS algorithm and thus achieve high packing fraction and can be expected to yield good statistics. A commercial software application, MacroPac [31], uses a "blackberry" model – agglomerations of spheres – to model virtually any shapes in any combination in a single pack. MacroPac is a sequential method and is thus most useful for low packing fractions where the ordering and arrangement of the particles are not crucial in the end result.

Analytical shapes can have major advantages. The first and most obvious advantage is computational efficiency if a closed form solution exists for determining the distance between the two analytical shapes. This is of course true for sphere–sphere interactions, but is unfortunately not true for even moderately more complex shapes. For example, determining the distance between two cylinders at arbitrary orientation in 3D is extremely difficult because of the need to determine the distance between the four circles at the ends of any two cylinders. In such situations, the analytical shape might still be used, along with a numerical minimization or root solver for determining the distance between the shapes. However, as we will discuss further, this can greatly increase computation time.

Another advantage for analytical shapes is that it is possible to represent them exactly. The ability to calculate exact distance to a surface eases the root finding process for determining the distance between two shapes by reducing the overall amount of error in the process. This is especially important for the LS algorithm, which has tolerance issues when the collision prediction is not exact. If the overall error in the process is too large, them the particles might overlap slightly, causing the LS heap to get stuck. Having an exact calculation for the location of a particle surface reduces the likelihood of occurrence of this phenomena.

Even if an analytical representation is not available for a particular shape of interest, an analytical method might still be possible using constructive solid geometry (CSG). With CSG, complex solid shapes are constructed from boolean operations on simple analytical shapes such as spheres, cylinders, and polyhedra. CSG is available in many commercial solid modeling and visualization applications.

If CSG is considered as a viable solution for creating more complex (but still analytical) shapes, then the major drawback of using analytical shapes is an overall lack of flexibility to accommodate new shapes. For each pair of shapes, one must be able to predict collisions, which in turn requires calculating the distance between the two shapes. When the two shapes are spheres or polygons, this is not difficult. However, adding a new shape requires adding new code to compute the distance between the new shape and all existing shapes. Thus, the development cost of adding a new shape becomes prohibitive as the number of shapes increases.

The second drawback of using analytical shapes was mentioned above briefly. There is no guarantee that an analytical solution can be found for the distance between two shapes that have simple, closed form descriptions of their surface locations. For example, computing the distance between two circles in 3D is a notoriously difficult problem [32]. Thus, for some shapes, it may be more simple to compute the distance numerically – even though the shapes may be very simple. However, if a numerical method is used for the distance calculation, the first disadvantage discussed above still remains, and is in fact made more important because of the limited scope of the numerical method. Also, the computation time can be increased drastically because each collision prediction requires multiple distance computations, which each in turn compute many function evaluations (more on this later).

Still, a CSG method that restricts itself to a particular simple shape or set of shapes, such as MacroPac's blackberry model, can be very useful for many physical problems. A method similar to MacroPac based on CSG using only voxels (a voxel is the

analog of a pixel in 3D) was presented in [33] and commercialized as DigiPac. Although DigiPac is a sequential method, it has evolved to include DEM-like forces and has been able to achieve fairly high packing fractions with small numbers of particles ($N \leqslant 175$). DigiPac has been successfully validated with experiments (using only the packing fractions ρ) with various particle shapes [34]. In the propellant and energetic materials context, the disadvantages of the method used by DigiPac are that it cannot achieve high enough packing fractions and the shapes are not resolved below the voxel resolution (they are not smooth). The former shortcoming is crucial for our problems; the latter is only important when fine detail is needed on both small and large particles.

Another widely-utilized approach to creating packs of non-spherical shapes is based on the Voronoi tessellation. The Voronoi method begins with a tessellation about random points and then shrinks the resultant polyhedra to achieve a particular packing fraction. This approach has been used in the combustion and energetic materials community [35,18] for the compelling reason that it can be used to create packs with *any* packing fraction $0 \leqslant \rho \leqslant 1$. The method also yields very realistic-looking particle shapes. Among the methods discussed here, the Voronoi-based packing codes are the only ones able to reach the high packing factions that suit our needs. Unfortunately, the method is not without shortcomings.

First, it is difficult to alter the shape of the resulting polyhedra. In particular, it is impossible to alter the aspect ratios of the particles without affecting the packing fraction. For packs with low packing fractions, this might not be a problem, since there is likely to be ample space to change the aspect ratios while keeping the particles from overlapping. At high packing fractions, however, there is unlikely to be enough space to significantly alter the aspect ratio. A second, but similar problem exists when there is large polydispersity. Unless the particles can be significantly shrunk, all of the largest particles will be roughly spherical due to the properties of the tessellation itself. Since the shape of the largest particles in a microstructure is of utmost importance, this is a major drawback.

Another problem with Voronoi methods is that although the resulting polyhedra are truly random, their orientations are completely non-random with respect to their neighbors' orientations. In 2D, the faces are seen to fit together like puzzle pieces or the spots of a giraffe, indicating a lack of randomness. Depending on the details of the algorithm, it is also likely that the closest distance between any two neighboring particles is a constant. However, at this point, we cannot explicitly quantify the effect of this lack of randomness would have on computed physical properties.

In light of the fact that none of the available methods meet the requirements detailed above, we propose a new method based entirely on level sets. Level set methods (isosurfaces) have some major advantages. First, it is possible to represent *any* shape for which one can calculate the signed minimum distance field. Also, because the underlying data structure is always a scalar field (an isosurface), it is possible to mix different shapes with no changes to the method. In addition, subgrid resolution enables precise surface mappings with small amounts of discrete data, limited only by the order of the interpolation method. A simplifying property is that the method to calculate the distance between the shapes can be independent of the shapes. Lastly, isosurface scaling (necessary for concurrent packing) is of course trivial if the signed minimum distance is used to represent the shape. Despite these advantages, we are not aware of any other use or proposed use of level sets or other type of isosurface in the context of computer generated random packs.

However, level sets have been used in other discrete element modeling contexts. For example, a few of the properties of level sets were exploited in a stacking algorithm by Guendelman et al. [36], where a level set was coupled with a triangulated mesh to produce fast inside/outside checks and normal vector computations. This method could possibly be adapted to work within the LS algorithm, but the authors chose not to do so for two reasons. First, the method requires both a triangulation and a level set field for each object. For smooth objects, triangulation reduces the accuracy of the surface representation, and this is unnecessary if the level set can be used alone. Second, the method uses the triangulation vertices as sample points for evaluating the level set field. This is very similar to a direct search method. Although direct search methods may be acceptable for stacking, they are prohibitively expensive within the LS packing framework due to the tolerance requirements of the latter method. Thus, we developed an algorithm that relies on level sets as the sole representation of the particle's surfaces and that also computes inter-particle distances in constant time (independent of the mesh resolution). Of course no method is without its challenges, and these will also be discussed below.

3. Algorithm

Because this algorithm will be used in the context of the LS framework for packing, we can view the shape representation as an unspecified function or a "black box". We put into the function a pair of particles and the function returns a prediction for the time of the particles' next collision (or ∞ if they will not collide in their current trajectories). Additionally, if the two particles are actually the next pair to collide, the LS algorithm will also require the location of the contact and a vector normal to the surface at the point of contact so that the proper changes in trajectory can be applied after the collision is processed.

For two particles i and j the black box functions can be approximately described by the pseudocode of Algorithm 1. The function Predict-Collision takes as arguments the two particles i and j and returns the predicted time of the next collision between the particles. The member function Update simply applies a time step update to a particular particle to calculate its position, velocity, orientation, and angular momentum after the time step Δt. Predict-Collision functions as a root solve, searching for the first time after $t = 0$ that the distance d falls below zero. Note that much of the detail is hidden in the important function call to Compute-Distance(a,b), which returns the instantaneous signed minimum distance between the two particle copies a and b.

Algorithm 1. A "black box" that predicts the time for the next collision between particles i and j.

```
PREDICT-COLLISION(i,j)
  ▷ Operate on copies of i and j.
  a ← i
  b ← j
  t ← 0
  repeat
      t ← t + Δt
      a.UPDATE(t)
      b.UPDATE(t)
      d_ij ← COMPUTE-DISTANCE(a,b)
    until d_ij > 0
  return t
```

The procedure described above is obviously simplified somewhat, but it illustrates the basic computational task: we must integrate the particle positions in time, checking for overlap after each small time step Δt. The rest of the LS framework does not know or care how this is done – a prediction of the next collision time is all that is necessary. This property makes it relatively easy to incorporate any shape representation into the LS framework as long as one can implement a suitable PREDICT-COLLISION function for the representation. In the next section, we delve further into the details of the root solver.

One of the most desirable characteristics of the level set representation is that we can easily (in principle) compute a "distance" function that is piecewise smooth and monotonically decreasing at the instant that the shapes begin to overlap. This has important ramifications when attempting to zero in on an impending collision. According to the skeleton definition of PREDICT-COLLISION(i,j) above, we need only a boolean check on $d_{ij} > 0$. We have presupposed that d_{ij} is available and is also well-behaved as the particles begin to overlap. In fact, we do not need to have a distance function at all; any metric that indicates when particles have overlapped can (in principle) be used to find the exact instant that the particles touch. Other methods for finding the collision point between arbitrary shapes include simply checking for voxel overlap [33] or scaling the particle sizes smoothly until overlap is detected by some other method [29]. Of these methods, the former is simply a boolean check and the latter generates a smoothly varying function that changes sign at the moment of overlap. However, a boolean check is a poor method for finding a root, with a convergence rate of $O(\Delta t)$ (note that the method in Ref. [33] does not use the LS algorithm, and thus never needs to find a root). A smooth function is much more suitable for pinpointing the precise location of an impending collision, but the method used in [29] is not generally available for arbitrary shapes. Fortunately, as we will show below, we can directly compute the signed minimum distance function $d_{ij}(t)$ for two level set shapes i and j, and furthermore, this function is well-behaved at the zero-crossing.

Assuming we can compute the instantaneous distance $d_{ij}(t)$ between two shapes represented by level sets, we must then find the next root of $d_{ij}(t)$ numerically. Because we intend for this method to be used with arbitrary shapes, there will be no analytical method for finding the root. We can, however, gain some insight into how the shape of $d_{ij}(t)$ might look by decomposing the shape's motion into that of its maximum enclosed sphere (with radius r_i) and its non-spherical motion. The distance between the two maximum enclosed spheres $d_{12,\text{sphere}}(t)$ looks something like a parabola

$$d_{12,\text{sphere}} = \sqrt{\|\boldsymbol{v}_{ij}\|^2 t^2 + 2\boldsymbol{v}_{ij}\cdot\boldsymbol{x}_{ij}t + \|\boldsymbol{x}_{ij}\|^2} - (r_{ij} + g_{ij}t) \tag{1}$$

where $\boldsymbol{v}_{ij} = \boldsymbol{v}_j - \boldsymbol{v}_i$, $r_{ij} = r_i + r_j$, and $g_{ij} = g_i + g_j$. This function is nearly parabolic near the point of collision and becomes linear as $t \to \pm\infty$. Superimposed on this distance will be the periodic oscillations of the rotational degrees of freedom of the two non-spherical shapes. Thus, in general, $d_{ij}(t)$ will be highly nonlinear and oscillating near the next collision $d_{ij}(t) \to 0^+$.

Shown in Fig. 2 is the computed $d_{ij}(t)$ for two gelcaps (spherocylinders). To make this figure, the two gelcaps were given random velocities, angular velocities, orientations, and positions and were allowed to pass through each other. The overall shape resembles Eq. (1), but there are oscillations due to the changing orientations of the particles. Note that in general, these oscillations do not have a regular period and that they can cause rather sharp peaks to be present in the distance function $d_{ij}(t)$.

To find this root, we resort to an ODE-like curve fit and attempt to find the next t for which $d_{ij}(t) < \epsilon$, where ϵ is some small tolerance below which we are not concerned with overlap. Donev et al. have implemented a method using Hermite polynomials for ellipsoids [29]. For arbitrary shapes, $d'(t)$ is not generally available, and so we choose a piecewise quadratic fit. An error estimate is made at each step and the size of the interval is adapted if the following criterion is not met:

$$\int_t^{t+\Delta t} d_{ij}^*(\theta)\,d\theta > -\epsilon \tag{2}$$

where d_{ij}^* is the piecewise polynomial fit to d_{ij}. At each interval, the fitted quadratic function is checked for a bracketed positive root. If one is found, the interval is further refined using Brent's method [37] – a combination of bisection, secant, and

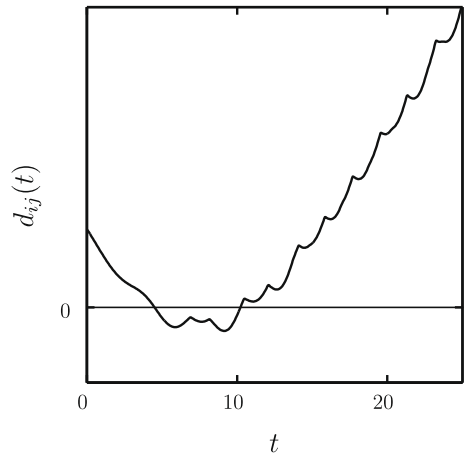

Fig. 2. Distance between two gelcaps (spherocylinders) as a function of time. The distance function $d_{ij}(t)$ resembles a quadratic function with superimposed oscillations. The quadratic-like portion (which is actually quartic) is due to the relative movement of the centroids, while the oscillations are due to the rotations as the non-spherical shapes tumble through space.

inverse quadratic interpolation methods. Thus, although small collisions may be missed due to finite ϵ, any scheduled collision will be predicted to high precision. As previously explained, this precision is critical to the success of the Lubachevsky–Stillinger algorithm.

We now show more detail of the black box in Algorithm 2, where we have deferred the shape copying and UPDATE(t) member functions into the function COMPUTE-DISTANCE(i,j,t) so that the function of the root solve is clear. Next, we explain our algorithm for the COMPUTE-DISTANCE(i,j,t) function, which exposes the details of our shape representation.

Algorithm 2. Details of the collision prediction scheme of our implementation. A piecewise quadratic fit is first formed by advancing copies of the particles forward in time. If a zero-crossing is located, Brent's method is used to refine the estimate for the root.

```
PREDICT-COLLISION(i, j, t_max)
  t_0 ← 0
  d_0 ← COMPUTE-DISTANCE(i, j, t_0)
  repeat
      t_1 ← t_0 + Δt/2
      t_2 ← t_0 + Δt
      d_1 ← COMPUTE-DISTANCE(i, j, t_1)
      d_2 ← COMPUTE-DISTANCE(i, j, t_2)
      Form piecewise quadratic fit d*_ij on interval [t_0, t_2]
      error ← ∫_t^{t+Δt} d*_ij(θ) dθ > −ε
      if error > target/2
        then Δt ← Δt/2
      elseif error < 2target
        then Δt ← 2Δt
             t_0 ← t_2
             d_0 ← d_2
      ▷ Check for bracketed root.
      if d_2 < 0
        then return BRENT-FIND-ROOT(i, j, t_0, t_2)
    until t_2 > t_max
  return ∞
```

3.1. Level set shapes

A level set (or isosurface) has some nice properties that make it useful for our purpose. The shape's surface $\Gamma(\boldsymbol{x},t)$ (in 3D) is represented implicitly by the zero level set of an auxiliary field $\phi(\boldsymbol{x},t)$

$$\Gamma(\boldsymbol{x},t) = \{\boldsymbol{x}|\phi(\boldsymbol{x},t) = 0\} \tag{3}$$

where the t dependence is included because our shapes translate, rotate, and grow in time. Our task is simplified if ϕ is the signed minimum distance to Γ

$$\phi(\boldsymbol{x},t) = \alpha \min_{\boldsymbol{y}}\{\|\boldsymbol{x}-\boldsymbol{y}\| | \phi(\boldsymbol{y},t) = 0\} \tag{4}$$

where we take the sign $\alpha = 1$ for points $\boldsymbol{x}$ that lie outside the shape and $\alpha = -1$ for points that lie inside the shape. Then we have $\|\nabla\phi\| \equiv 1$, and $\nabla\phi$ is a normal vector perpendicular to the surface Γ.

The shapes will in general be undergoing affine transformations in three dimensions and will also be growing according to the Lubachevsky–Stillinger algorithm. For a general affine transformation, we have

$$\boldsymbol{x} = \sigma(t)\boldsymbol{A}(t)\boldsymbol{x}_0 + \boldsymbol{b}(t) \tag{5}$$

where $\boldsymbol{A}(t)$ is a rotation tensor, $\boldsymbol{b}(t)$ is a translation vector, $\sigma(t)$ is the scaling factor, and $\boldsymbol{x}_0$ is the position in the reference frame at $t = 0$. Our shapes will be bounded by their minimum enclosing spheres, so we scale the level set so that $\sigma(t) = R(t)$, where $R(t)$ is the radius of the enclosing sphere. Unfortunately, the signed minimum distance fields are nonlinear and so there is no linear operator that can be applied to map $\phi(\boldsymbol{x}_0, t = 0) \to \phi(\boldsymbol{x},t)$. Instead, the transformation must be applied pointwise,

$$\phi(\boldsymbol{x},t) = \sigma(t)\phi(\sigma(t)\boldsymbol{A}(t)\boldsymbol{x}_0 + \boldsymbol{b}(t), t = 0) \tag{6}$$

which is very expensive.

With the Lubachevsky–Stillinger packing algorithm, we only deal with two shapes at a time, and as discussed in the previous section the kernel of the algorithm requires predicting the time of the next collision between those two shapes via Predict-Collision(i,j,t_{max}). The level set representation must provide a means to compute the instantaneous distance between two shapes at any time by the function Compute-Distance(i,j,t). We begin by transforming both shapes back to a reference configuration, computing the inverse of Eq. (5) for each shape. Then, we combine the shapes by taking their intersection. The intersection of the two level sets is just the maximum of the two fields ϕ_1 and ϕ_2

$$\phi_{ij}(\boldsymbol{x},t) = \max\{\phi_i(\boldsymbol{x},t), \phi_j(\boldsymbol{x},t)\} \tag{7}$$

and the resulting field $\phi_{ij}(\boldsymbol{x},t)$ is also a signed minimum distance field if the shapes are overlapping. Whether or not the shapes overlap, the intersection provides useful information because we can use it to calculate the signed minimum distance between the two shapes. To accomplish this, we compute the minimum of their intersection

$$d_{ij}(t) = 2\min_{\boldsymbol{x}} \phi_{ij}(\boldsymbol{x},t) \tag{8}$$

where a negative d_{ij} indicates that the shapes are overlapping and the factor of 2 accounts for the fact that the location of the minimum is halfway between the two particles' surfaces. Thus, as shown in Algorithm 3, Compute-Distance(i,j,t) is rather simple. Again, we have deferred most of the work to the function Get-Minimum$(i,j,\boldsymbol{x}_0)$, which attempts to locate the minimum of the intersected fields. Having found the minimum, we can use Eq. (8) to compute the signed minimum distance.

Algorithm 3. Procedure for computing the instantaneous distance between two particles i and j. Copies of the particles are updated to the desired time t by moving them without collisions along their current trajectories. Then, the Get-Minimum$(i,j,\boldsymbol{x}_0)$ is used to find the distance.

```
Compute-Distance(i,j,t)
  ▷ Operate on copies of the shapes
  a ← i
  b ← j
  a.Update(t)
  b.Update(t)
  ▷ Form initial guess for location of minimum.
  x_0 ← (a.x + b.x)/2
  d_ij = 2 · Get − Minimum(i,j,x_0)
  return d_ij
```

We now discuss the process of locating the minimum of the intersected level set field $\phi_{ij}(\boldsymbol{x},t)$, which is a difficult task. Multivariate, nonlinear minimization is in general a hard problem for smooth functions, but it becomes almost impossible for functions that have strong discontinuities in the gradient [38]. Level sets and their intersections are functions of the latter type, and for intersections in particular, there are guaranteed to be surfaces of gradient discontinuity where the gradient – and thus the direction of steepest descent – changes abruptly. This effect is caused by the `min` functions in Eqs. (4) and (8) and is present whether or not the shapes are smooth. Furthermore, for an intersection, the minimum will always lie on one of these surfaces where the gradient does not exist. An additional complicating factor is that the fields are always locally linear in the direction of $\nabla\phi$. Thus, even if we travel downhill in the direction of steepest descent, we can never gain more infor-

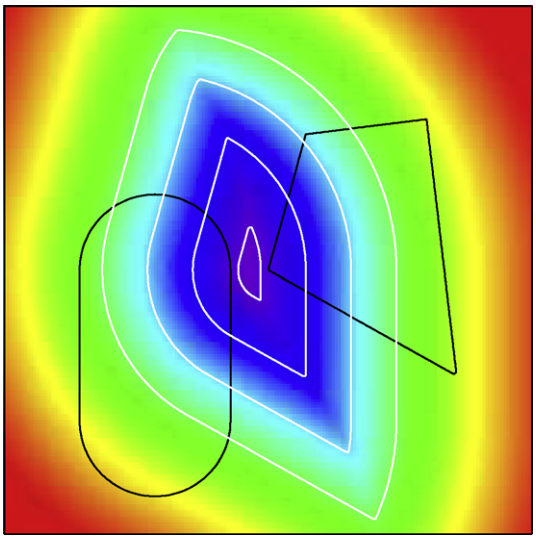

Fig. 3. Intersection field $\phi_{ij}(\boldsymbol{x}, t)$ of the signed minimum distance fields of two convex shapes. The outlines of the 2D shapes (the zero contours of $\phi_i(\boldsymbol{x}, t)$) are shown in black. The color field and white contours show the intersected field. When the shapes are both convex, a single, global minimum exists. However, note the complexity of the intersected field, particularly the presence of multiple ravines where the gradient jumps. The global minimum can be seen to lie in one of these ravines. In 3D, the ravines correspond to surfaces where the gradient does not exist.

mation about where the minimum will be located until we pass a location where $\nabla\phi$ actually changes. Locally, the fields have no characteristic length scale. Because of the linearity of the fields and discontinuous gradients, quadratic methods such as Newton–Raphson or Broyden–Fletcher–Goldfarb–Shanno (BFGS) and conjugate gradient methods will fail in this degenerate case.

An example of the two-dimensional intersected level set field $\phi_{ij}(\boldsymbol{x}, t)$ for two different convex shapes is shown in Fig. 3. Outlines of the two shapes are shown in black. The color[1] field and the white contour lines show the values of the intersected field $\phi_{ij}(\boldsymbol{x}, t)$. The signed minimum distance field has a smoothing effect far from the surfaces of the shapes, but this feature is of limited value in this context. Because of the `max` function in Eq. (7), note that the left side of the figure contains contours (and field values) for the polygon, while the right side of the figure contains the contours for the gelcap on the left side. These contours meet at the ravine passing vertically through the field. The global minimum lies in this ravine, inside the innermost contour, and the gradient does not exist at the minimum.

In this situation, it would seem that there are two options: (1) try a robust direct search method, or (2) design a new method. We have already achieved partial success with the former option, having applied the Nelder–Mead simplex method [39] as well as the Hooke and Jeeves direct search [40]. Of the two, the simplex method fared better and was able to find the general location of the global minimum quite reliably. The Hooke and Jeeves search tended to require many more function evaluations than Nelder–Mead to achieve the same level of tolerance. However, the Lubachevsky–Stillinger algorithm is not very tolerant of even very small errors in the collision prediction times, and neither method was able to consistently provide good estimates with relative tolerances below 10^{-3}. The poor performance for both methods occurs when the minimizer gets stuck on a surface of gradient discontinuity. It is easier to visualize the problem in 2D, where a height field indicates the value of the objective function. In 2D, the surfaces where the gradient is undefined become ravines where the slope changes abruptly. Both the Nelder–Mead and the Hooke–Jeeves methods move quickly downhill into the nearest ravine. Near the bottom of the ravine, they shrink their respective search templates rapidly to convergence, but are unable to make forward progress in the direction of the ravine. They are thus often unable to find the location of the true minimum.

Therefore, a major focus of this research was concerned with the second option: design of a new minimization method. In spite of the pitfalls discussed above, the signed minimum distance fields do have features amenable to minimization. One of these properties is the aforementioned linearity of the fields. If we perform a line search in the $-\nabla\phi$ direction, then we are guaranteed to travel downhill at a slope of unity until a discontinuity (in the gradient) is reached. In the remainder of this discussion, we will call these discontinuities "ravines" for brevity and because it is easier to visualize the process in 2D. At a ravine, the best search direction is the one that travels *in the direction of the ravine*. The location and direction of the ravine are both easily estimated by linearizing the field on either side of the ravine. We linearize the field separately at two locations $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ that straddle the ravine, dropping the time dependence for clarity

[1] For interpretation of color in Figs. 3–5 and 15, the reader is referred to the web version of this article.

$$\phi_a(\boldsymbol{x}) = \phi(\boldsymbol{x}_a) + \nabla\phi(\boldsymbol{x}_a)\cdot(\boldsymbol{x}-\boldsymbol{x}_a) \tag{9}$$

$$\phi_b(\boldsymbol{x}) = \phi(\boldsymbol{x}_b) + \nabla\phi(\boldsymbol{x}_b)\cdot(\boldsymbol{x}-\boldsymbol{x}_b) \tag{10}$$

The linearized ravine is a line in 2D (or a plane in 3D). To find an appropriate location to begin a search, we project a line in the downhill direction at x_a,

$$\boldsymbol{x}(t) = \boldsymbol{x}_a + t(\boldsymbol{x}_b - \boldsymbol{x}_a) \tag{11}$$

and find its intersection with the ravine. With some algebra, we find

$$t = \frac{\phi(\boldsymbol{x}_b) - \phi(\boldsymbol{x}_a) + \boldsymbol{n}_b\cdot(\boldsymbol{x}_b-\boldsymbol{x}_a)}{(\boldsymbol{n}_b-\boldsymbol{n}_a)\cdot(\boldsymbol{x}_b-\boldsymbol{x}_a)} \tag{12}$$

where the $\boldsymbol{n}_i = -\nabla\phi_i(\boldsymbol{x}_i)$ are the downhill normals. The estimate of the direction of steepest descent along the ravine is then

$$\boldsymbol{n}_{ab} = \frac{\boldsymbol{n}_a + \boldsymbol{n}_b}{2\|\boldsymbol{n}_a + \boldsymbol{n}_b\|} \tag{13}$$

Note that although the intersected field is linear with unity slope along the gradient direction, it is not in general linear in the direction of the ravine $\boldsymbol{n}_{ab}$. Also, with respect to the magnitude of the gradient, we know only that $\|\nabla\phi_{ij}\| \leqslant 1$ along the ravine.

We devised a method that uses these properties to find the true global minimum to high precision. Given a starting location $\boldsymbol{x}_0$, the method first moves downhill in the direction of steepest descent $-\phi_{ij}(\boldsymbol{x})/\|\phi_{ij}(\boldsymbol{x})\|$. When the minimizer encounters a ravine, the direction of the ravine is first estimated using Eq. (13), and then a line search is performed along that direction. The process then repeats.

One of the requirements for using this process (and a drawback to the overall level set shape representation) is that the shapes must be convex. The reason for this requirement is illustrated in Fig. 4, where we show the intersected level set field $\phi_{ij}(\boldsymbol{x})$ for two shapes that have concavities. The outlines of these shapes are shown in black, and the values of $\phi_{ij}(\boldsymbol{x})$ are again shown in color with white contour lines. The white contour lines show the existence of multiple minima in $\phi_{ij}(\boldsymbol{x})$ for these shapes. The method we have proposed could easily get stuck in a local minima and miss the global minimum, and so we require all of the shapes to be convex (but not strictly convex).

Simplified pseudocode for the minimization process is shown in Algorithm 4. The Move-Downhill-Into-Ravine($\boldsymbol{x},\boldsymbol{n}$) procedure begins at location $\boldsymbol{x}$ and moves in the downhill direction $\boldsymbol{n}$ until the function evaluations $\phi_{ij}(\boldsymbol{x})$ no longer decrease as expected. This signals the location of a ravine. In the special case of signed minimum distance fields, the ravine may be found numerically by comparing the expected decrease in the field to the actual decrease in the field. Since signed minimum distance fields have a slope of unity in the search direction, the expected decrease is equal to the step size. The criterion for crossing a ravine is then

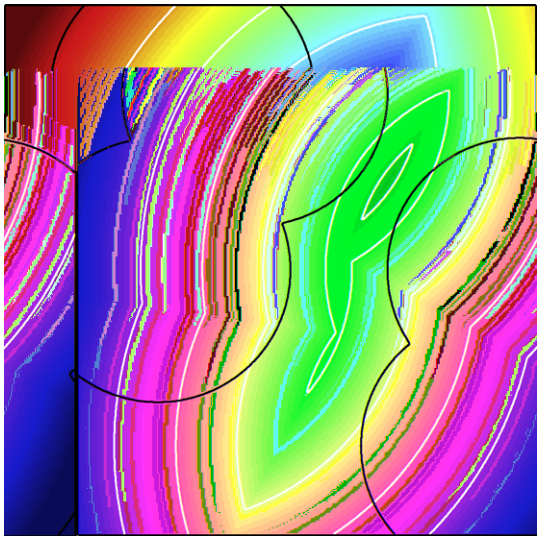

Fig. 4. Intersection field $\phi_{ij}(\boldsymbol{x},t)$ of the signed minimum distance fields of two non-convex shapes. If the shapes are not convex, the intersected field can contain multiple local minima, which reduces the chances that the minimization routine will converge to the correct global minimum.

Algorithm 4. Procedure for locating the minimum of the intersection of two level set fields that represent two convex objects. The procedure first attempts to locate a surface where the gradient is undefined (a ravine) and then performs a line minimization constrained to that surface.

```
GET-MINIMUM(i,j,x)
  f ← φ_ij(x)
  n ← −φ_ij(x)/‖φ_ij(x)‖
  count ← 0
  x_accel ← x
    repeat
      x_old ← x
      MOVE-DOWNHILL-INTO-RAVINE(x,n)
      q ← GET-RAVINE-DIRECTION(x)
      f ← LINE-SEARCH-FOR-MIN(x,q)
      count ← count + 1
      if count > accelCount
        then ACCELERATE-IN-DIRECTION(x − x_accel)
          count ← 0
          x_accel ← x
    until ‖x − x_old‖ < tolerance
  return f
```

$$(\phi_{ij}(\boldsymbol{x}+\Delta\boldsymbol{x},t)-\phi_{ij}(\boldsymbol{x},t))+\|\Delta\boldsymbol{x}\|>\epsilon \tag{14}$$

where Δx is the step and ϵ is some small tolerance. In practice, we have found $\epsilon = 0.05$ to work well. Larger values are also possible since it is not necessary to identify and search along ravines where the slope has only moderately changed.

Upon finding a ravine, the function will attempt to bracket the location of the ravine. MOVE-DOWNHILL-INTO-RAVINE is adaptive. If a particular step is successful, the step size is generally increased; likewise if the step is unsuccessful, the step size is decreased. However, we have found that there is a loss of efficiency if the function tries to find the location of the ravine to high precision. It is better to limit the step size adaptivity to say, 3 decreases, and then to move on to the line search rather than to attempt many function evaluations in this phase. At the end of this bracketing process, the closest points on each side of the ravine are chosen as $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ of Eq. (13).

LINE-SEARCH-FOR-MIN($\boldsymbol{x},\boldsymbol{q}$) begins at location $\boldsymbol{x}$ and performs a line search in the ravine direction $\boldsymbol{q}$. Brent's minimization method (a variation of the Golden Search) [37] is used to converge upon the location of the minimum along the line $\boldsymbol{y} = \boldsymbol{x} + t\boldsymbol{q}$. Brent's method requires a bracketed minimum, so before using the method, our algorithm attempts to quickly bracket the minimum by moving along the line with an adaptively increasing step size.

Acceleration is performed in the function ACCELERATE-IN-DIRECTION($\boldsymbol{x} - \boldsymbol{x}_{accel}$). Acceleration can be very useful in speeding up convergence to the minimum, especially in the (rather common) event that the intersected level set field $\phi_{ij}(\boldsymbol{x})$ has moderate curvature. If the curvature is significant, the downhill and line searches will tend to zig-zag towards the global minimum when it would of course be more efficient to move in an approximately straight line toward the goal. ACCELERATE-IN-DIRECTION thus projects a line search along the direction of a recent rolling average of the overall search progress, for example $\boldsymbol{x} - \boldsymbol{x}_0$, again performing an adaptive bracketing and successive Brent search.

As with almost all minimization methods, the method presented here greatly benefits from having a good initial guess for the location of the minimum. Since we already know that the bounding spheres of the pair of shapes are overlapping, a good initial guess can be quickly found by locating the center of the overlap region of the bounding spheres. For the shapes utilized in this work, the aspect ratios are reasonably close to unity, and so this initial guess method is adequate and efficient. Nevertheless, on rare occasions the algorithm does fail to find a minimum; in these cases, the pair of particles are treated as if they have collided immediately so that overlap of particles is avoided.

An example run of the minimization routine is shown in Fig. 5, where the search has been restricted to two dimensions for ease of visualization. Also note that for illustrative purposes, the initial guess is not as described above. Outlines of two gelcaps (spherocylinders) are shown in black, while the field values and contours of $\phi_{ij}(\boldsymbol{x})$ are again shown as color and white lines, respectively. The path of GET-MINIMUM($i,j,\boldsymbol{x}$) is shown in red. The initial guess was below the lower bound for y in the figure, so the path of the minimizer begins at the bottom of the figure. (We would normally start with an educated guess for the initial guess.) The function then moves in the downhill direction until it finds the first ravine. Then, it moves in the direction of the ravine, rapidly approaching the true point of contact. The inset presents a zoomed-in look at the action of the minimizer as the global minimum is approached, where the successive downhill-linesearch process can be seen.

The overall minimization method has been successful when used within the LS framework. The global minimum can typically by found with fewer than 100 function evaluations to a tolerance $\|\boldsymbol{x} - \boldsymbol{x}_{old}\| < 10^{-6}$. That is not to say that the minimization routine is optimal; it is not. In particular, although Brent's method can be second order, the linearization of the level

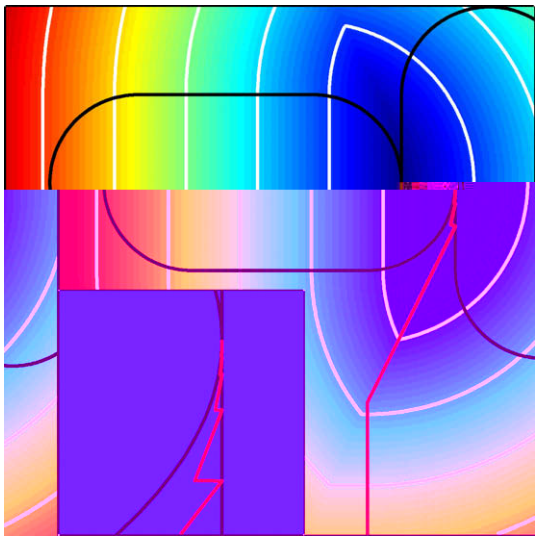

Fig. 5. Progress of the minimum search algorithm in 2D. The outlines of the two gelcaps (spherocylinders) are shown in black. The color field and white contour lines show the intersected field $\phi_{ij}(\boldsymbol{x}, t)$. Shown in red is the path of the minimizer. The inset shows the minimizer's path near the global min, zoomed in and expanded in the x-direction to show more detail.

set fields by Eqs. (9) and (10) causes the overall method to be at best first order. Nevertheless, as we will now show, the method does indeed work.

4. Results

4.1. Examples of packs

As mentioned previously, the method presented above can in principle be used to pack any convex shape for which one can compute the initial signed minimum distance field $\phi(\boldsymbol{x})$. Fortunately, these fields are also easy to calculate for many types of shapes including polyhedra, spherocylinders, cylinders, and of course spheres. We have computed the fields for these shapes and have successfully packed variants of each type.

Note that the method presented here is independent of the algorithm used to compute the signed distance fields. These fields may be computed on-the-fly for simple analytical shapes such as spheres and cylinders, or the signed distances may be pre-computed for more complex shapes and then stored as a mesh (a 3D array) in memory. The mesh method must be coupled with some sort of 3D interpolation technique so that the interpolated values vary smoothly between mesh nodes. We have applied both analytical and mesh representations of the shapes presented in this work. For the meshes, we used trilinear, tri-quadratic, and tri-cubic polynomial interpolation. In all cases, the method was found to work independently of the mesh resolution so long as the interpolated field values were continuous (but the shapes and packing fractions necessarily depend on the ability to resolve the original shape). Higher order interpolations are of course computationally expensive, but this was alleviated somewhat by pre-computing and storing the final polynomial coefficients at each location on the mesh. However, the relatively simple shapes presented in this work all have straightforward analytical solutions for the signed distance field, and these analytical methods were favored for their speed.

Shown in Fig. 6 is a pack of 1000 gelcaps (spherocylinders) with aspect ratio 2:1 (L/D) in a periodic cube configuration. The gelcaps were started with a scaled temperature $\Theta_0 = 10^3$ (see Ref. [27], [41], or [28] for a discussion of packing temperature). Gelcaps are smooth and the intersected level sets are relatively well-behaved, so it is easy for the minimization routine to locate the minimum. Therefore, these represent an almost optimal scenario for the method (spheres would be optimal, but uninteresting), and if only gelcaps are present the code is only about 10 times slower than the LS algorithm for spheres. Interestingly, the gelcaps pack to a higher packing fraction than spheres under the same conditions, $\rho = 0.70$ for gelcaps versus $\rho = 0.63$ for spheres [27]. This result is independent of the random seed to two significant digits; for example, under the same packing conditions with three different random seeds, the spherocylinders reached packing fractions of 0.696, 0.697, and 0.697. We can estimate an upper limit for the volume fraction of a pack of spherocylinders based on geometry considerations. If we assume that the gelcaps align themselves as an ordered stack of infinitely long cylinders, then a cross-section will reveal a pack of disks that have the maximum theoretical area fraction of 0.906. The gelcaps occupy a volume ratio of 5/6 of their bounding cylinders, so the maximum packing fraction would be $0.906 \times 5/6 = 0.755$, which is higher than the maximum volume fraction of ordered spheres (0.7405). Recently, a similar property was observed with packs of ellipsoids (M&M candies), where MRI experiments revealed that the ellipsoids pack up to $\rho = 0.739$, depending on the aspect ratio [42].

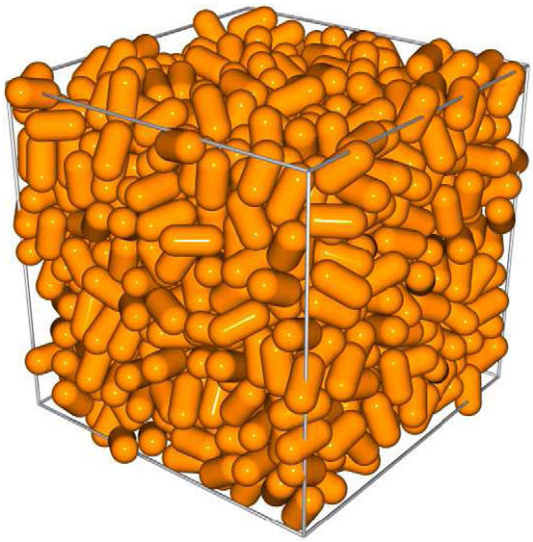

Fig. 6. A pack of 1000 gelcaps (spherocylinders) in a periodic cube, packed with $\Theta_0 = 10^3$. Interestingly, the gelcaps pack to a final packing fraction $\rho = 0.70$. A pack of spheres under identical conditions jams at only $\rho = 0.63$.

Gelcaps are interesting in their own right, but are not very similar to more the common propellants and energetic materials. Shown in Fig. 7 is a set of 500 right circular cylinders with aspect ratio 2:1, again packed with $\Theta_0 = 10^3$. This type of particle shape is similar to the 7-perforation grains used in modern cannon propellants [9], and the hope is that such a pack might be 1 day used to investigate convective burning in a packed gun propellant. Note that the edges of the cylinder are slightly rounded. The rounding is not strictly necessary but was found to increase the efficiency of the minimization drastically. Even a small amount of rounding can decrease the number of function evaluations by 10-fold, so most of the sharp edges of the particle shapes in this work have been slightly rounded.

A pack of 500 metabidiminished icosahedrons in a cubic container is shown in Fig. 8. This shape was chosen because unlike regular polyhedra, it resembles the irregular particle shapes seen in some energetic materials. Shown in Fig. 9 is a pack with icosahedra, gelcaps, and spheres, illustrating the method's capability of packing any combination of shapes at any size ratio. There were no code changes needed to handle the shapes together in the same pack. We are unaware of any other methods by which polyhedra may be packed alongside non-polyhedral shapes of any kind, including spheres. This is an important capability when modeling energetic materials such as PBX-9501 that consist of large, polyhedral shape particles

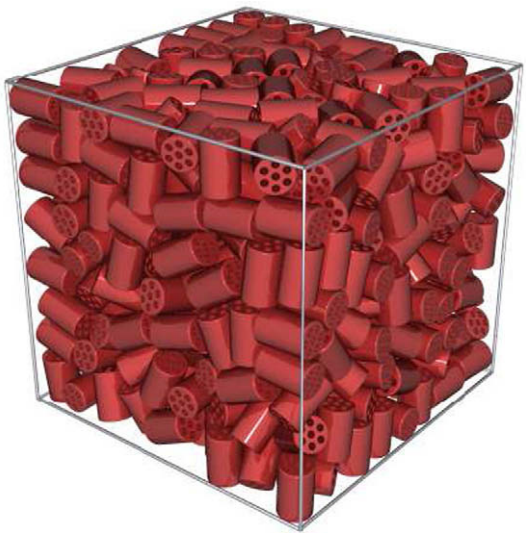

Fig. 7. A pack of 500 right circular cylinders rendered as 7-perforation gun propellant grains in a cubic container. The packing fraction of the cylinders is $\rho = 0.485$ (not accounting for the void fraction of the perforations).

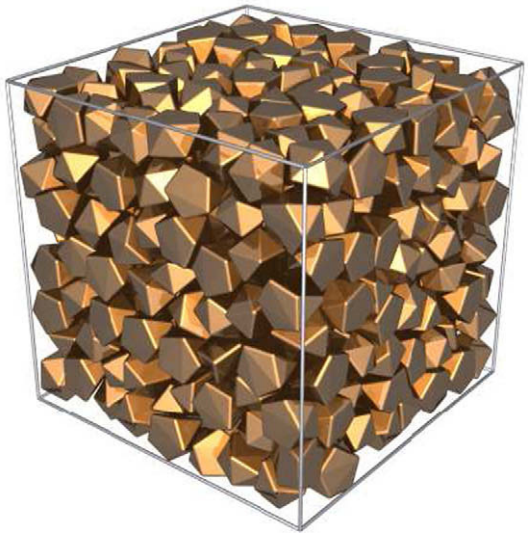

Fig. 8. A pack of 500 metabidiminished icosahedrons in a cubic container with packing fraction $\rho = 0.626$. Complex polyhedra such as these are desirable for modeling random packs of crystalline particles.

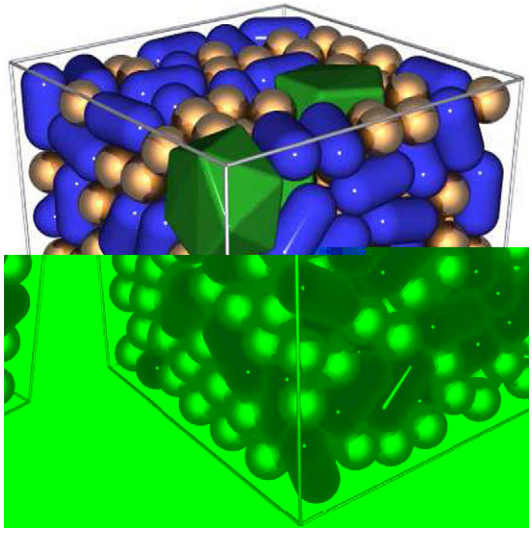

Fig. 9. A pack of spheres, gelcaps (spherocylinders), and metabidiminished icosahedrons, illustrating the capability of packing any kinds of convex shapes in any combination. All of these shapes use the same algorithm in the code; their only differences in the code are in their respective level set field values.

along with many smaller, spherical or cylindrical particles embedded in a rubberized binder (see Fig. 1 for a cutaway photo of the PBX-9501 microstructure).

Relevant to the same problem are the "HMX crystals" shown in Fig. 11. This model shape is currently being used in packs that resemble crystalline energetic materials and was modeled after the large HMX crystals that are being grown at the Los Alamos National Laboratory, shown in Fig. 10 [43]. The model crystal was constructed by beginning with a hexagonal prism and moving some of the vertices to emulate the structure of the real crystalline shapes. A virtual pack using the crystal model is shown in Fig. 12. The periodic pack consists of small cubes, small and large icosahedrons, and small and large HMX model polyhedra at a packing fraction of $\rho = 0.710$. An identical pack of spheres with the same size ratios, number of particles, and scaled temperature reached a packing fraction $\rho = 0.709$.

4.2. Statistics

For more detailed quantification of the properties of these packs, we turn to multi-point probability distributions. We show here only a few statistics with the simple justification that we cannot discuss the results with respect to experimental

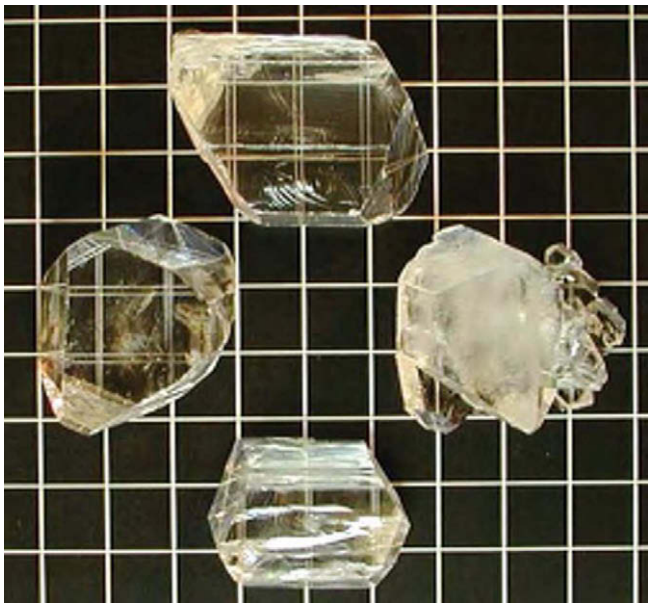

Fig. 10. Large HMX crystals grown at Los Alamos National Lab. These are single crystals photographed on a 1 cm grid. (Photo courtesy of D. Hooks at LANL [43].)

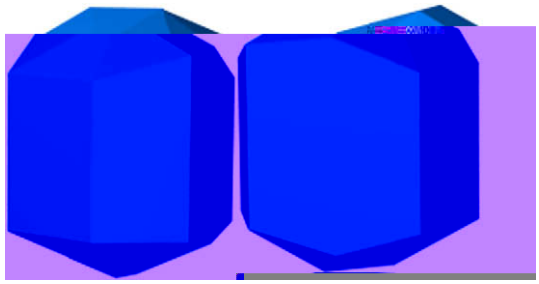

Fig. 11. A model of an HMX crystal used in the packing. The shape was modeled from the photos of the large HMX crystals grown at Los Alamos National Lab [43].

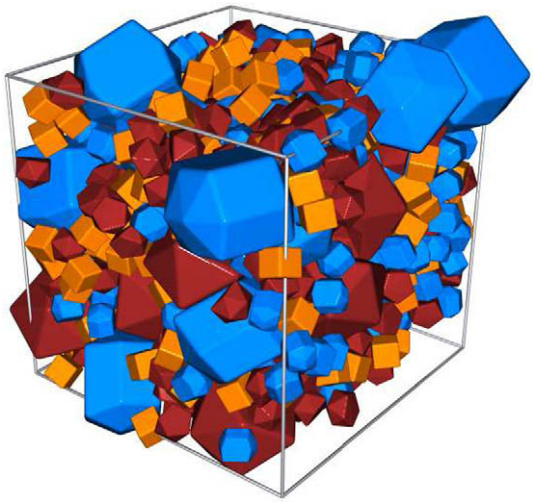

Fig. 12. A virtual model of a periodic pack of crystalline HMX. The pack has 200 each of icosahedrons, cubes, and "HMX crystals" of relative size 1 with 10 more icosahedrons and 20 more HMX crystals of relative size 3. The pack reached a packing fraction $\rho = 0.71$ with $\Theta_0 = 10^3$. Only particles whose centroids lie in the periodic domain (shown by the bounding box) are pictured.

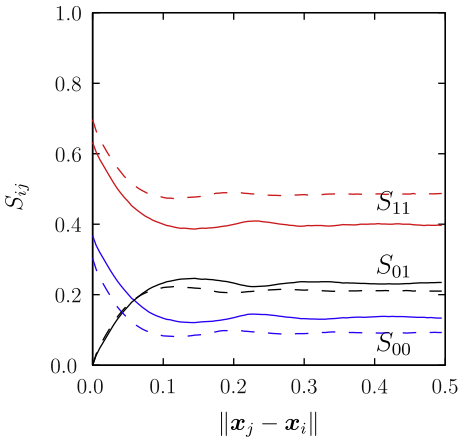

Fig. 13. Second order statistics S_{ij} for the gelcap pack of Fig. 6, shown in dashed lines. The solid lines correspond to S_{ij} of a pack of spheres that was jammed under the same conditions. The gelcap length is 0.162 (aspect ratio 2:1) and the sphere radius is 0.106. S_{11} for the gelcaps begins at a higher probability than the spheres because of the higher packing fraction (0.70 for gelcaps versus 0.63 for spheres). Note also that the first peaks for the gelcaps are moved closer to $\|\boldsymbol{x}_j - \boldsymbol{x}_i\| = 0$ than the peaks of the spheres, indicating that the gelcap is in contact with closer neighbors. This is the expected result, since the gelcap has a smaller radius than a sphere of equivalent volume.

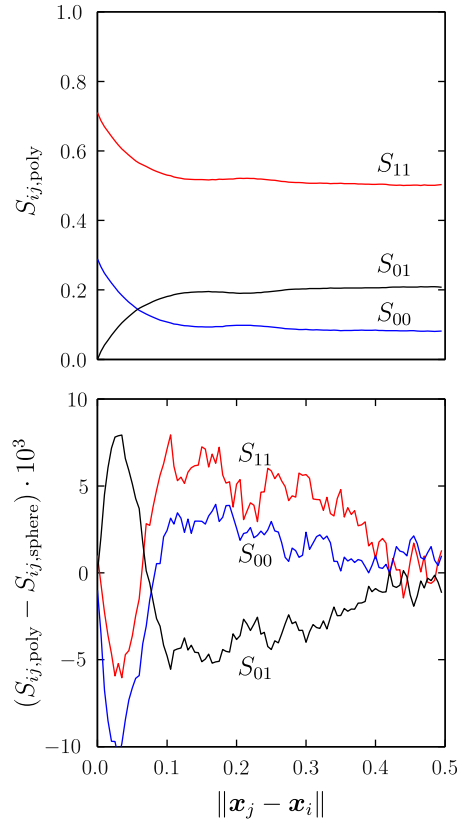

Fig. 14. (top) Second order statistics S_{ij} for the HMX model pack of Fig. 12. (bottom) Difference of the statistics relative to S_{ij} of a pack of spheres that was jammed under the same conditions. Note that although the spheres are jammed at $\rho = 0.71$, the crystal pack is not jammed. The longest dimension of the largest polyhedra is ≈0.79; the diameter of the largest spheres is ≈0.57.

results. XCT scans of various particle packs are generally not available, and although packs of spheres generated by *Rocpack* have been validated against published experimental data [27], the same is not possible here.

If the microstructures of interest are ergodic, isotropic, and homogeneous, we can use simplified one-, two-, and three-point probability functions to describe the microstructure. A complete discussion of these probability functions, their basis, and the reduction to their simple forms when the above requirements are met is beyond the scope of this work (see [44] for a more detailed discussion). We describe the functions only briefly here for completeness.

For the purposes of our brief description, we will assume that the materials of interest have only two phases, are ergodic, are isotropic, and are homogeneous. If we sample the microstructure (our packed bed of spheres or other shapes) at random locations $\boldsymbol{x}_q$, we can compute the one-, two-, and three-point probability functions S_i, S_{ij}, and S_{ijk}. $S_i(\boldsymbol{x}_q)$ is the probability that $\boldsymbol{x}_q$ lies in the ith phase, $S_{ij}(\boldsymbol{x}_q, \boldsymbol{x}_r)$ is the probability that $\boldsymbol{x}_q$ lies in the ith phase while $\boldsymbol{x}_r$ simultaneously lies in the jth phase, and $S_{ijk}(\boldsymbol{x}_q, \boldsymbol{x}_r, \boldsymbol{x}_s)$ is defined similarly for three points. Fortunately, when the material meets the requirements listed above, these probability functions can be simplified significantly to

$$S_i(\boldsymbol{x}_q) = \rho_i \tag{15}$$
$$S_{ij}(\boldsymbol{x}_q, \boldsymbol{x}_r) = S_{ij}(\|\boldsymbol{x}_r - \boldsymbol{x}_q\|) \tag{16}$$
$$S_{ijk}(\boldsymbol{x}_q, \boldsymbol{x}_r, \boldsymbol{x}_s) = S_{ijk}(\|\boldsymbol{x}_r - \boldsymbol{x}_q\|, \|\boldsymbol{x}_r - \boldsymbol{x}_s\|) \tag{17}$$

Note that we can take the limits of these functions for both small and large $\|\boldsymbol{x}_r - \boldsymbol{x}_s\|$ to obtain bulk properties

$$S_{ij}(0) = \delta_{ij}\rho_i \quad \text{(no sum)} \tag{18}$$
$$S_{ij}(\infty) = \rho_i\rho_j \tag{19}$$

with similar consistency checks available for S_{ijk} (see Ref. [45] for general random two-phase media). In the following discussion, we use the subscript 0 to denote sample locations inside the matrix material (or voids) and the subscript 1 to denote sample locations inside the particles. For example, $S_{110}(\boldsymbol{x}_q, \boldsymbol{x}_r, \boldsymbol{x}_s)$ is the probability of simultaneously finding a point $\boldsymbol{x}_q$ inside a particle, $\boldsymbol{x}_r$ inside a particle, and $\boldsymbol{x}_s$ inside the matrix.

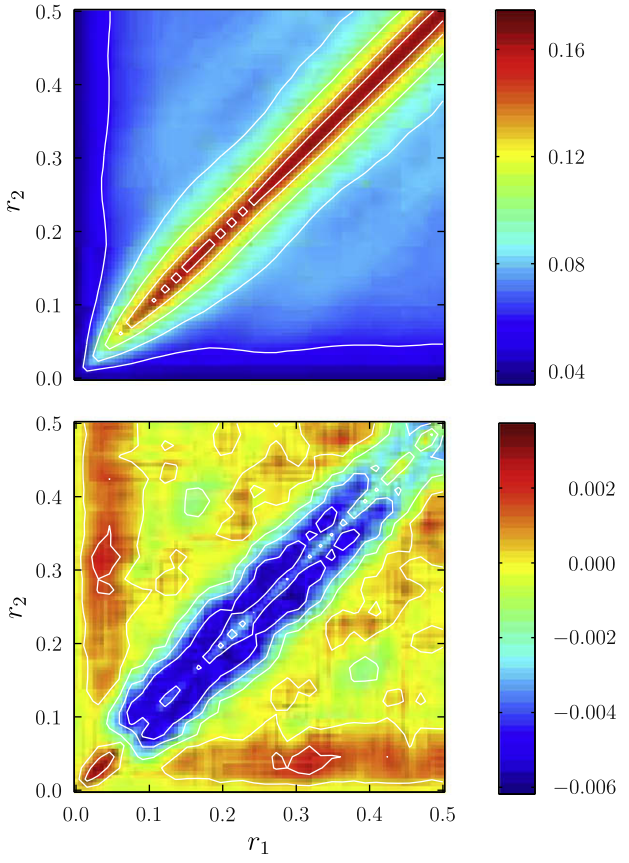

Fig. 15. (top) Three-point probability function S_{011} at $\theta = 0^\circ$ for the pack in Fig. 12. (bottom) The difference of S_{011} relative to an equivalent pack of spheres ($S_{011,\text{poly}} - S_{011,\text{sphere}}$). Here, $r_1 = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, $r_2 = \|\boldsymbol{x}_k - \boldsymbol{x}_i\|$, and θ is the angle between the two vectors $\boldsymbol{x}_j - \boldsymbol{x}_i$ and $\boldsymbol{x}_k - \boldsymbol{x}_i$. Red corresponds to high probability, while blue corresponds to low probability.

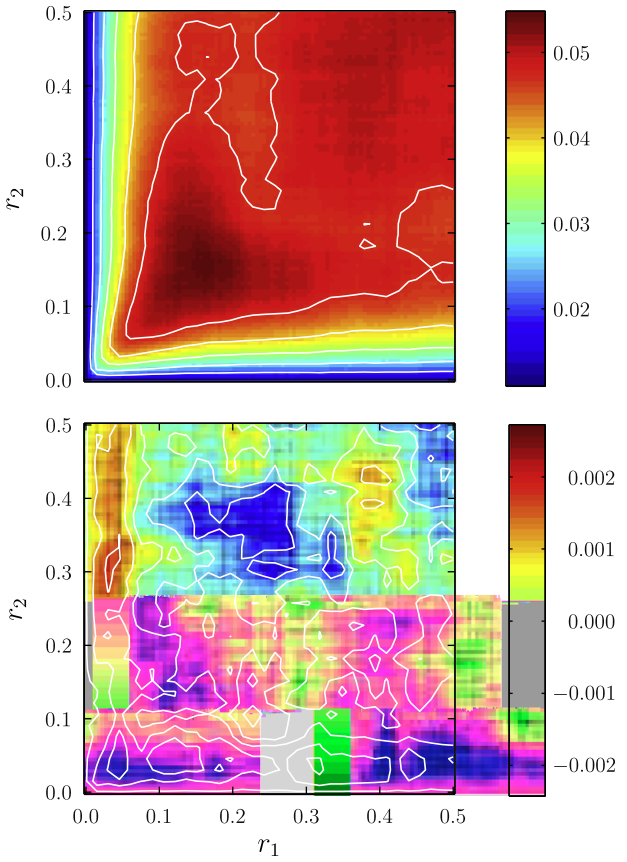

Fig. 16. (top) Three-point probability function S_{011} at $\theta = 72^\circ$ for the pack in Fig. 12. (bottom) The difference of S_{011} relative to an equivalent pack of spheres ($S_{011,\text{poly}} - S_{011,\text{sphere}}$).

We first consider the results of the gelcap pack of Fig. 6. Recall that this pack jammed at a packing fraction of $\rho = 0.70$, which is 10% higher than the jamming density of spheres under similar conditions. The second order statistics S_{ij} are compared to S_{ij} for an equivalent pack of spheres in Fig. 13. The dashed lines correspond to the gelcap pack while the solid lines correspond to S_{ij} of a pack of spheres that was packed under the same conditions. The characteristic lengths for the two particle shapes are an overall length of 0.162 for the gelcaps and a radius 0.106 for the spheres. The gelcaps have an aspect ratio of 2:1, so their radii are 0.081. The final packing fraction differences can be clearly seen in the figure because they are equal to the intercept of the S_{11} functions. Note also that the first peaks for the gelcaps are moved closer to $\|\boldsymbol{x}_j - \boldsymbol{x}_i\| = 0$ than the peaks of the spheres. This is an expected result, indicating the presence of touching neighbors at closer distances than in the sphere pack and is due to the fact that the gelcap has a smaller radius than a sphere of equivalent volume.

The second order statistical metrics S_{ij} for the pack of Fig. 12 are shown in Fig. 14 along with the difference relative to S_{ij} for a very similar pack of spheres. The pack of spheres contains the same number of particles and the same size ratios as the crystal pack. However, although the spheres are jammed at $\rho = 0.71$, the crystal pack is not jammed at that density. Also, because spheres have a larger volume to length ratio, the longest dimension of the largest polyhedra is greater than the diameter of the large spheres: ≈0.79 for the polyhedra versus ≈0.57 for the spheres. Nevertheless, the S_{ij} functions for the two packs are very similar (within 1%).

The statistical differences in the two packs are more apparent in the higher-order statistical functions. The three-point probability function S_{011} for the same pack is shown in Fig. 15. The distances between the three randomly thrown points x_i, x_j, and x_k are indicated on the axes by $r_1 = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$ and $r_2 = \|\boldsymbol{x}_k - \boldsymbol{x}_i\|$. $\theta = 0^\circ$ is the angle between the two vectors $\boldsymbol{x}_j - \boldsymbol{x}_i$ and $\boldsymbol{x}_k - \boldsymbol{x}_i$. The large probability along the diagonal (red is high probability) is expected because when $r_1 = r_2$, $S_{011}(r_1, r_2, \theta) = S_{01}(r)$. There is a small, but noticeable difference in the width of the central band; it is wider for spheres. But even with this third-order statistic, the packs seem similar. More differences can be seen in Fig. 16, where $\theta = 72^\circ$ and Fig. 17, where $\theta = 180^\circ$, and the contours are moderately different. Despite what seem to be small differences in the packs, there are successful ways to compute, for example, statistically optimal unit cells based on small, but quantifiable differences in these probability functions [44].

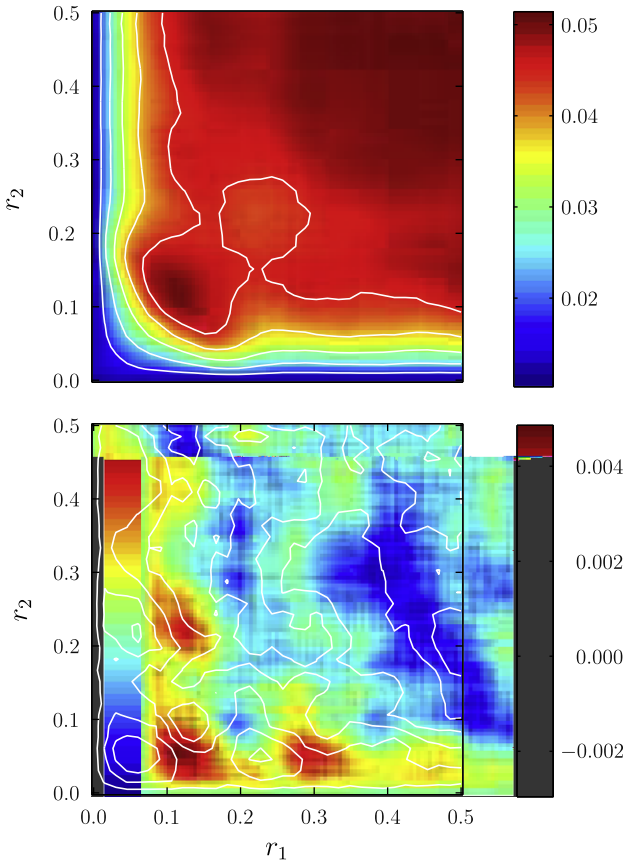

Fig. 17. (top) Three-point probability function S_{011} at $\theta = 180^\circ$ for the pack in Fig. 12. (bottom) The difference of S_{011} relative to an equivalent pack of spheres $(S_{011,\text{poly}} - S_{011,\text{sphere}})$.

5. Conclusions

In this article we have introduced a new method for the creation of virtual packs of arbitrary shapes of particles. The method uses a level set (or isosurface) representation of the shape, which has the advantages of being able to accommodate almost any shape or any combination of shapes in the same framework. We are unaware of any other use of level sets or other type of isosurface in the packing literature. The level set representation also allows us to efficiently predict the next collision between any two particles because we can compute a smooth, and well-behaved distance function $d_{ij}(t)$ for the signed minimum distance between two particles. We also showed why standard minimization algorithms cannot be used with this method and subsequently introduced a new minimization procedure that takes advantage of the specific properties of intersected level sets.

We demonstrated several packs with various shapes that were created using the new method. Some of the shapes of interest to us are gelcaps, cylinders, and irregular polyhedra. We demonstrated the capability of packing these shapes together, in the same container, with no change to the method. In particular, we demonstrated packs that visually resemble HMX crystals and that also have similar packing fractions. We also found that gelcaps, much like ellipsoids, jam at significantly higher packing fractions than spheres.

To reiterate, the primary advantages of this method are that it can pack any combination of convex shapes together and that – thanks to the LS framework – high packing fractions can be achieved. We are unaware of any other available commercial or research tools that have this capability.

Unfortunately, we were unable to validate the results of the packing algorithm by comparing to real materials – except by visual comparisons. The only way to truly quantify how closely our virtual packs resemble the intended microstructures would be to compare them to microstructure data from real materials. For example, second- and third-order statistics could be computed for tomography experiments from XCT or MRI and quantitatively compared to statistics from our packs. At this time, XCT data is difficult to obtain, but XCT experiments at the Center for Simulation of Advanced Rockets (CSAR) at the University of Illinois are being conducted in conjunction with the Beckman Institute with the goal of quantifying the statistics of both real and simulated energetic materials. Current experiments are focused on gathering information about the packing characteristics of bidisperse glass spheres.

It is of course most helpful to have information on the real materials of interest, and so later experiments are planned for conducting scans of industrial propellant formulations. These experiments should be carefully designed so that their usefulness is maximized with respect to validation of the packing algorithm. It would be tempting to attempt the largest scan possible so that a fully-representative density plot can be demonstrated. However, to be useful for validation of the packing algorithm, many XCT scans of a single type of material should be obtained so that the variation between samples can be quantified. Also, since the level of polydispersity in real materials is very large, a single scan that attempts to capture many large particles is inappropriate because it will not resolve the details of the small particles. Rather, it would be better to capture many smaller samples that may include only a few of the largest particles (as in Fig. 1) so that smaller particles can be adequately resolved by the XCT equipment. Ergodicity can then be confirmed by comparing the high order statistics of these real samples with the same metrics from virtual packs. This method is also more appropriate for obtaining many small scans that can be directly compared to the statistically optimal unit cell construction algorithms [16].

However, even lacking explicit validation, many computational studies of interest in this field can be carried out without fully understanding how accurately the virtual packs reflect reality. For example, numerical combustion studies have indicated that packs of spheroids produce approximately the same burn rates as packs of spheres in AP/HTPB composite propellants [12]. The same null hypothesis may be true for other phenomena, and such studies could be used to give us further confidence in results obtained with packs of spheres. Furthermore, the development of methods that characterize or estimate the thermal and mechanical properties of real materials can carry on independently of whether or not experimental data sets exist because we can now create virtual packs to test these methods. For example, theories for obtaining bounds on heterogeneous solid material properties such as bulk moduli can be developed by testing done on virtual packs [14], now including packs of non-spherical shapes.

Also, even though bulk burn rate is apparently independent of the particle shape, it is still unclear whether or not this is true for other properties such as the acoustics in the rocket grain. Further studies can answer this question. Furthermore, since the method presented in this work uses level sets, there is a unique ability to smoothly morph any given shape into an equivalent spherical shape. Using this method, we cannot only answer the question of whether the particle shapes influence a given property, but we can also predict at what point the sphericity becomes an important factor. Using this technique, we plan to re-visit the problem of burning spheroidal AP particles in HTPB binder [12], but instead use more realistic shapes such as polyhedra for the burn computations. Since the shapes of some materials are obviously more spherical than other shapes, this type of study is applicable to other areas as well.

One such area is in prediction of pressure rise times in confined packs of loose propellant particles. These predictions are useful when designing solid rocket igniters and gun cartridges [6,7]. Numerical tools for simulating burn rates in porous materials are not yet widely available (see Ref. [46] for an example where the propellant has simulated cracks), but it is now possible to begin looking at porous burning of packs of realistic shapes of particles like those of Fig. 7.

Yet another direction for future research is the broad area of pack characterization. Packs of spheres have been well-characterized in terms of convergence of the first and second order statistics (volume fraction and $g(r)$, respectively) with respect to packing algorithm variables such as initial scaled temperature Θ_0, number of particles N, and the distance to jamming δ [27,28]. These convergence properties are also of interest for packs of arbitrary shapes, but have only been studied for simple shapes such as ellipsoids. Future work with packs created using the algorithms presented here could quantify these properties for packs of monodisperse and polydisperse polyhedra, spherocylinders, etc. present either individually or together in the same pack.

This is by no means an exhaustive list of possible topics for future research. Indeed, our discussion has been almost entirely limited to materials that are used as propellants or energetic materials. Many more opportunities for porous or heterogeneous modeling exist in chemical engineering, materials science, and medicine. It is our hope that the tool presented here will enable researchers to tackle the sorts of computational problems that have been previously impractical because of the difficulty in modeling the complex morphology.

Acknowledgments

This work was supported by the US Department of Energy through the University of California under subcontract number B523819. Dr. Jackson was also partially supported by the Air Force Research Laboratory under contract FA9550-06-C-0078, program manager Dr. A. Nachman.

References

[1] M. Baer, C. Hall, R. Gustavsen, D. Hooks, Isentropic loading experiments of a plastic bonded explosive and constituents, Journal of Applied Physics (2007).
[2] R. Armstrong, W. Elban, Materials science and technology aspects of energetic (explosive) materials, Materials Science and Technology (2006).
[3] T. Davis, The Chemistry of Powder and Explosives, Chapman and Hall, 1941.
[4] D. Adams, Igniter performance in solid-propellant rocket motors, Journal of Spacecraft and Rockets (1967).
[5] T. Davis, K. Kuo, Experimental study of the combustion processes in granular propellant beds, Journal of Spacecraft and Rockets (1979).
[6] D. Greatrix, J. Gottlieb, T. Constantinou, Numerical model for pellet-dispersion igniter systems, Journal of Propulsion and Power (1988).
[7] K. Kuo, R. Vichnevetsky, M. Summerfield, Theory of flame front propagation in porous propellant charges under confinement, stinet.dtic.mil (1971).
[8] B. Asay, S. Son, J. Bdzil, The role of gas permeation in convective burning, International Journal of Multiphase Flow (1996).
[9] D. Carlucci, S. Jacobson, Ballistics: Theory and Design of Guns and Ammunition, CRC Press, 2007.

[10] S. Kochevets, J. Buckmaster, T. Jackson, A. Hegab, Random packs and their use in modeling heterogeneous solid propellant combustion, Journal of Propulsion and Power (2001).
[11] G. Knott, T. Jackson, J. Buckmaster, Random packing of heterogeneous propellants, AIAA Journal (2001).
[12] X. Wang, J. Buckmaster, T. Jackson, Burning of ammonium-perchlorate ellipses and spheroids in fuel binder, Journal of Propulsion and Power (2006).
[13] T. Jackson, F. Najjar, J. Buckmaster, New aluminum agglomeration models and their use in solid-propellant-rocket simulations, Journal of Propulsion and Power (2005).
[14] R. Lipton, Variational methods, bounds and size effects for two-phase composites with coupled heat and mass transport processes at the two-phase interface, Journal of the Mechanics and Physics of Solids (1999).
[15] R. Hill, Elastic properties of reinforced solids: some theoretical principles, Journal of the Mechanics and Physics of Solids (1963).
[16] N. Kumar, K. Matouš, P. Geubelle, Reconstruction of periodic unit cells of multimodal random particulate composites using genetic algorithms, Computational Materials Science (2008).
[17] I. Macdonald, M. El-Sayed, K. Mow, F. Dullien, Flow through porous media – the Ergun equation revisited, Industrial and Engineering Chemistry Fundamentals (1979).
[18] M. Keyser, M. Conradie, M. Coertzen, J. Van Dyk, Effect of coal particle size distribution on packed bed pressure drop and gas flow distribution, Fuel (2006).
[19] J. Shepherd, D. Begeal, Transient compressible flow in porous materials, NASA STI/Recon Technical Report No. 88:22324, January 1988.
[20] J. Bernal, J. Mason, Packing of spheres: co-ordination of randomly packed spheres, Nature (1960).
[21] T. Gruhn, P. Monson, Isobaric molecular dynamics simulations of hard sphere systems, Physical Review E (2001).
[22] I. Volkov, M. Cieplak, J. Koplik, J. Banavar, Molecular dynamics simulations of crystallization of hard spheres, Physical Review E 66 (2002) 061401.
[23] M. Alam, S. Luding, Rheology of bidisperse granular mixtures via event-driven simulations, Journal of Fluid Mechanics (2003).
[24] R. Johansson, H. Thunman, B. Leckner, Influence of intraparticle gradients in modeling of fixed bed combustion, Combustion and Flame (2007).
[25] L. Massa, T. Jackson, M. Short, Numerical solution of three-dimensional heterogeneous solid propellants, Combustion Theory and Modelling (2003).
[26] M. Wackenhut, S. McNamara, H. Herrmann, Shearing behavior of polydisperse media, The European Physical Journal E-Soft Matter (2005).
[27] F. Maggi, S. Stafford, T. Jackson, J. Buckmaster, Nature of packs used in propellant modeling, Physical Review E (2008).
[28] S. Stafford, Random packs and their use in modeling high speed porous flow, Ph.D. Thesis, University of Illinois at Urbana-Champaign, 2008.
[29] A. Donev, S. Torquato, F. Stillinger, Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles. II. Applications to ellipses and ellipsoids, Journal of Computational Physics (2005).
[30] A. Donev, J. Burton, F. Stillinger, S. Torquato, Tetratic order in the phase behavior of a hard-rectangle system, Physical Review B (2006).
[31] R. Rowe, P. York, E. Colbourn, S. Roskilly, The influence of pellet shape, size and distribution on capsule filling—a preliminary evaluation of three-dimensional computer simulation using a Monte-Carlo technique, International Journal of Pharmaceutics (2005).
[32] C. Neff, Finding the distance between two circles in three-dimensional space, IBM Journal of Research and Development (1990).
[33] X. Jia, R. Williams, A packing algorithm for particles of arbitrary shapes, Powder Technology (2001).
[34] X. Jia, M. Gan, R. Williams, D. Rhodes, Validation of a digital packing algorithm in predicting powder packing densities, Powder Technology (2007).
[35] J. Reaugh, Checking out the hot spots, Science and Technology Review (2003).
[36] Eran Guendelman, Robert Bridson, Ronald Fedkiw, Nonconvex rigid bodies with stacking, ACM Transactions on Graphics 22 (3) (2003) 871–878.
[37] R. Brent, Algorithms for Minimization without Derivatives, Prentice-Hall, 1973.
[38] W. Press, B. Flannery, S. Teukolsky, W. Vetterling, Numerical Recipes in C, Cambridge University Press, 1988.
[39] J. Nelder, R. Mead, A simplex method for function minimization, Computer Journal (1965).
[40] R. Hooke, T. Jeeves, Direct search solution of numerical and statistical problems, Journal of the ACM 8 (1961) 212–229.
[41] A. Donev, S. Torquato, F. Stillinger, Pair correlation function characteristics of nearly jammed disordered and ordered hard-sphere packings, Physical Review E (2005).
[42] W. Man, A. Donev, F. Stillinger, M. Sullivan, Experiments on random packings of ellipsoids, Physical Review Letters (2005).
[43] D. Hooks, Growth and characterization of explosive single crystals at Los Alamos National Laboratory, CPIAC Bulletin 33 (2) (2007) 1.
[44] B. Collins, Reconstruction of statistically optimal periodic unit cells of multimodal particulate composites, M.S. Thesis, University of Illinois at Urbana-Champaign, 2008.
[45] S. Torquato, G. Stell, Microstructure of two-phase random media. I. The *n*-point probability functions, The Journal of Chemical Physics (1982).
[46] X. Wang, T. Jackson, L. Massa, Numerical simulation of heterogeneous propellant combustion by a level set method, Combustion Theory and Modelling (2004).