Extracting dislocations and non-dislocation crystal defects from atomistic simulation data

2010 Modelling Simul. Mater. Sci. Eng. 18 085001

(http://iopscience.iop.org/0965-0393/18/8/085001)

Modelling Simul. Mater. Sci. Eng. **18** (2010) 085001 (13pp) doi:10.1088/0965-0393/18/8/085001

Extracting dislocations and non-dislocation crystal defects from atomistic simulation data

Alexander Stukowski and Karsten Albe

Institut für Materialwissenschaft, Technische Universität Darmstadt, Petersenstr. 32, D-64287 Darmstadt, Germany

E-mail: stukowski@mm.tu-darmstadt.de

Received 5 July 2010, in final form 24 August 2010
Published 30 September 2010
Online at stacks.iop.org/MSMSE/18/085001

Abstract

We describe a novel method for extracting dislocation lines from atomistic simulation data in a fully automated way. The *dislocation extraction algorithm* (DXA) generates a geometric description of dislocation lines contained in an arbitrary crystalline model structure. Burgers vectors are determined reliably, and the extracted dislocation network fulfills the Burgers vector conservation rule at each node. All remaining crystal defects (grain boundaries, surfaces, etc), which cannot be represented by one-dimensional dislocation lines, are output as triangulated surfaces. This geometric representation is ideal for visualization of complex defect structures, even if they are not related to dislocation activity. In contrast to the recently proposed *on-the-fly dislocation detection algorithm* (ODDA) Stukowski (2010 *Modelling Simul. Mater. Sci. Eng.* **18** 015012) the new method is extremely robust. While the ODDA was designed for a computationally efficient on-the-fly analysis, the DXA method enables a detailed analysis of dislocation lines even in highly distorted crystal regions, as they occur, for instance, close to grain boundaries or in dense dislocation networks.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Crystal plasticity and fracture of materials are closely linked to dislocation activity. Atomistic simulation methods, like molecular dynamics (MD), allow us to simulate the deformation behavior of virtual samples by numeric integration of atomic trajectories. While standard visualization techniques in principle allow us to identify deformation processes by visual inspection of atomic configurations, a quantitative analysis as well as a detailed investigation of dislocation lines and other crystal defects in atomistic simulation data remain a challenge.

It is the fundamental idea of the dislocation concept to treat certain kinds of crystal defects as continuous, one-dimensional lines, characterized by an elementary geometric property known as the Burgers vector [1]. Many phenomena associated with dislocations, such as their contribution to plastic slip, dislocation reactions and elastic stress fields, can solely be described in terms of this abstract line model. On the other hand, important aspects of lattice dislocations, such as their mobility, nucleation, and interactions with other crystal defects, depend directly on the discrete nature of the atoms constituting dislocation lines. To study phenomena related to dislocation activity numerically, while taking into account all fundamental processes, atomistic simulation methods like MD are generally used. These simulation methods do not model dislocation lines explicitly, that is, they do not keep track of the positions, shapes and Burgers vectors of dislocations in the simulated crystal. This implicit treatment of dislocations and other crystal defects gives atomistic models their unique strength. The interpretation and analysis of dislocation processes in such simulations, however, is severely hindered in the atomistic picture. Therefore, we now review the key methods and approaches that have been developed in the past to study dislocations in atomistic simulations.

In three-dimensional simulations of crystal plasticity, dislocations are usually occluded by the surrounding crystalline material. For observing dislocation mediated processes, the first analysis methods focused on the identification and removal of atoms in the undisturbed regions of the crystal (the 'good' material [2]), leaving behind the atoms that belong to crystal defects (the 'bad' material). The level of sophistication of these techniques ranges from very simple (e.g. coordination number, atomic excess energy [3]) to more elaborate (e.g. common neighbor analysis (CNA) [4], centro-symmetry parameter [5], Ackland's bond-angle method [6]). All these methods share a common weakness: they provide no information on the type of crystal defect the atoms form. It remains to the versed scientist to decide which defect structures constitute dislocation lines, while not knowing their Burgers vectors.

The determination of the Burgers vector of a dislocation segment by means of the well-known Burgers circuit method has been described by Frank [2]. Its application to atomistic simulation data usually requires the extraction of a planar cut from the good material region surrounding the dislocation and tracing a closed path around the dislocation core. This manual process, however, is far too laborious to analyze simulations involving more than a few dislocations. An automated method, that enables a semi-quantitative analysis of dislocations is the slip vector approach of Zimmerman *et al* [7]. The slip vector of an atom is calculated from the relative displacements of its nearest neighbors due to crystal slip. In the simple case of a single dislocation, all atoms above and below the glide plane in the wake of the dislocation exhibit a slip vector that is equal to the Burgers vector of the dislocation. Since the atomic slip vectors are not directly linked to the generating dislocation line, this approach is not suitable for characterizing dislocations in situations where multi-slip and dislocation reactions occur. Moreover, the slip vector technique can only detect the presence of gliding dislocations that produce slip; sessile lattice dislocations (e.g. in low-angle grain boundaries) cannot be analyzed by means of this method.

Hua and Hartmaier [8] recently employed the slip vector method in conjunction with the bond-angle method of Ackland and Jones [6]. They used the latter method to first identify atoms in the dislocation cores, while the former method was then applied for determining the corresponding Burgers vector from the local slip vector. The slip vector alone, however, is not sufficient to determine the sign of the Burgers vector, since the glide direction of the dislocation is not known *a priori*. Hua and Hartmaier resolved this ambiguity by taking into account the asymmetric core structure of edge dislocations in fcc crystals, which is revealed by a coordination analysis. A different method for characterizing single dislocations, which can be considered an alternative to the Burgers circuit method, has been devised by Hartley and

Mishin [9]. They have shown how the Nye tensor field associated with a dislocation can be derived from the discrete lattice distortions induced by an atomistically modeled dislocation. Then, by integrating the Nye tensor field over a sufficiently large area around the dislocation core, they were able to determine the Burgers vector of the dislocation with good accuracy.

All methods presented so far have in common that they do not provide a transition path from the atomistic dislocation picture to the discrete line representation. This restriction renders some important analyses impossible. For instance, without a discrete line representation, the local line direction is not properly defined, making it impossible to determine the screw/edge character of a dislocation. In the same way, exactly measuring the dislocation density and identifying dislocation nodes also requires the usage of the line concept. Another technical drawback of the atomistic representation is the large storage size as compared with a discrete line representation. This aspect is of particular importance for the investigation of rapid dislocation processes in large-scale MD simulations [10].

In a recent study [11], we have developed the *on-the-fly dislocation detection algorithm* (ODDA) that detects dislocation lines in atomistic simulation data, determines their Burgers vectors and transforms them into a discrete line representation. The ODDA is fully automated, parallelized, and can be efficiently performed within MD simulations. This allows one to study dislocation processes with picosecond time resolution, even in multi-million atoms MD simulations. So far, we have successfully applied the ODDA to measure dislocation densities as functions of strain in nanotwinned fcc metals [12] and nanocrystalline Pd–Au alloys [13]. In these studies, the dislocation analysis served as a very useful tool for measuring the respective fractions of different types of dislocations (i.e. Shockley partials, twinning partials and immobile stair-rod dislocations) as well as their preferred glide systems in a polycrystal.

Among the various processing steps of the ODDA, tracing of short Burgers circuits is the most important. In essence, the algorithm constructs a massive number of Burgers circuits around crystal defects, which have been identified with the CNA, to determine their Burgers vector. By ensuring that the circuits are tightly wrapped around dislocation cores, we can take their centers of mass as approximate points on the dislocation lines, which we want to reconstruct. In the last two processing steps, space points of equal Burgers vector are grouped and approximated by one-dimensional line segments, which are finally merged into a network by means of a heuristic matching method. The last two steps of the ODDA, however, can become unreliable in the case of complex and dense dislocation networks. Since the Burgers circuit search yields only disconnected and scattered space points (a so-called 'point cloud'), it is virtually impossible to make the reconstruction of the line network from this point set sufficiently reliable to ensure that the topology of the atomistic dislocation network is preserved in all cases. If, for example, the density of obtained points is too sparse for constructing a continuous line connecting these points, spurious dangling dislocations ending in the perfect crystal can occur, thus violating the Burgers vector conservation rule [14]. This problem plays a minor role if averaged quantities like the total dislocation density are of interest, but it can render an investigation of details of a complex and dense dislocation network difficult.

In this paper, we propose a novel algorithm, which follows a simple, but rigorous idea as illustrated in figure 1 and which does not show the shortcomings of the ODDA described above. The new method directly translates the network of disordered core atoms into a network of connected dislocation segments, thereby preserving its true topology. The output provided by this algorithm is a network of discrete dislocation lines and nodes, which entirely conforms to the Burgers vector conservation rule, and which reflects the shape of the underlying dislocation lines with atomic-scale precision. Moreover, the new algorithm delivers a geometric description of all other (non-dislocation) crystal defects (e.g. grain boundaries,

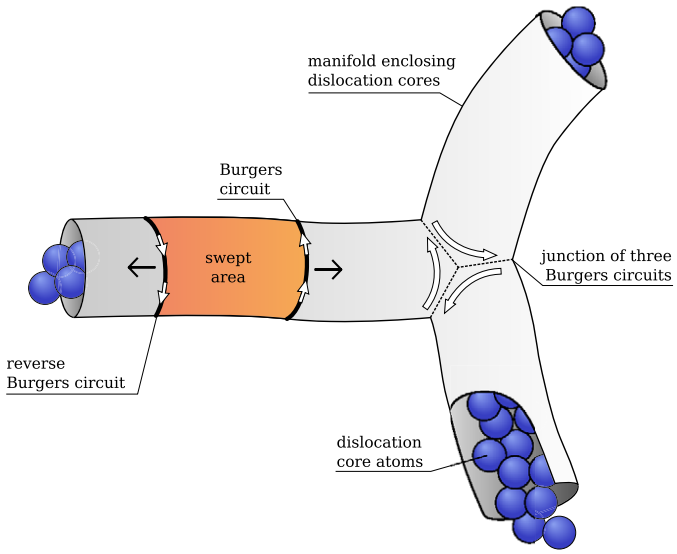

Figure 1. Schematic illustration of the new DXA. First a manifold surface enclosing the dislocation cores is constructed. Then elastic 'Burgers circuit bands' are swept over the manifold along each dislocation segment. Finally, nodal points are generated where multiple Burgers circuits meet on the manifold.

surfaces, pores, etc), which is very useful for visualization purposes and other applications. The basic algorithm requires only a single control parameter, making its usage extremely simple. In the following, we describe the algorithm in full detail and finally give an example.

2. Dislocation network extraction

The *dislocation extraction algorithm* (DXA) presented in this paper consists of three principal steps:

(i) The CNA method [4] is used to identify crystalline atoms (referred to as 'good' material by Frank [2]). We call the remaining atoms, which form crystal defects, 'disordered atoms'.
(ii) A closed, orientable, two-dimensional manifold (referred to as interface mesh) is constructed that separates the crystalline atoms from the disordered ones.
(iii) For each dislocation segment, an initial Burgers circuit is generated on this manifold. The closed circuit is moved in both directions to the two opposing ends of the dislocation segment to capture its shape (figure 1). While the circuit is advanced in each direction, a one-dimensional line representing the dislocation segment in the output is constructed.

Note that the DXA is in most parts a generic algorithm, independent of the underlying crystal structure. To give a concrete example whenever necessary, we will describe its implementation for body-centered cubic (bcc) crystals. The adaptation of the basic algorithm to other crystal structures requires only minor modifications.

2.1. Establishing lattice directions

As part of the first step of the DXA, crystalline atoms (having a particular arrangement of neighbors) are identified by means of the standard CNA method. The CNA takes into account

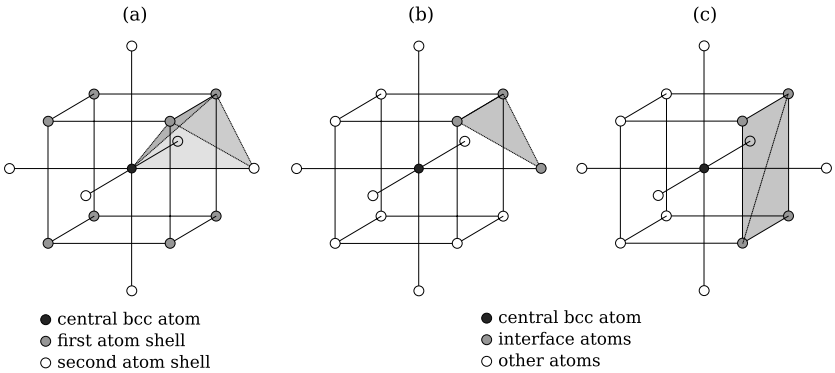

Figure 2. (*a*) The first and second shell of neighbors of a bcc atom. Tetrehedra like the one shown are used to carry the local lattice orientation over to nearest neighbor atoms (see section 2.1). (*b*) and (*c*) Two different types of triangular facets that are generated to construct a closed mesh around crystal defects.

the bonds between nearest neighbors to identify crystalline atoms. This allows us to make certain assumptions on the interconnectivity of nearest neighbors given that an atom has been classified as crystalline by the CNA. For instance, a bcc atom has 14 neighbors of which eight are in the first neighbor shell. The CNA guarantees that each of these eight nearest neighbors has exactly three neighbors that belong to the first shell of the central bcc atom, and three neighbors that belong to the second shell (see figure 2(*a*)).

As noted by Frank [2], the 'good' crystal region around dislocations is characterized by the possibility to uniquely define crystal lattice directions throughout the whole region—even in the presence of elastic distortions—by constructing a series of tetrahedra that link neighboring atoms together and pass over the local orientation information. Frank's idea allows us to devise a method to unambiguously associate a vector of the imaginary reference lattice with each atomic bond in the real crystal. To this end, we use the following scheme in the bcc crystal structure: as shown in figure 2(*a*), for each bcc atom, 24 tetrahedra can be constructed that each connect the central bcc atom with two of its nearest neighbors and one of its next-nearest neighbors. If we assume that we already know the lattice vectors corresponding to the three neighbor bonds of the central bcc atom[1], then we can directly calculate the lattice vectors of the three remaining tetrahedron edges. This information is sufficient to determine the lattice vectors of all respective neighbors of the two gray corner atoms in figure 2(*a*). Finally, we can repeat the tetrahedron construction at the newly visited lattice sites, and ultimately map all bonds in the 'good' crystal region to lattice vectors of the imaginary reference crystal.

A crystal may contain several isolated 'good' regions (we call them crystallite clusters), which are completely separated by 'bad' regions (e.g. grain boundaries), that is, no continuous path can be constructed that connects two of these regions. In the following, each of these clusters can be processed independently from the other clusters, since, according to Frank's definition, no lattice dislocation can exist in between such clusters. That is, all good atoms enclosing a dislocation form a single cluster.

Note that the actual atomic input coordinates are required only for determining the bonds between atoms during the first phase of the CNA. All subsequent computations, including the actual CNA as well as the complete DXA described in the following, are based on this

[1] For the very first atom, we are free to chose any crystal lattice orientation and assign corresponding lattice vectors to the 14 neighbor bonds of this 'seed' atom.

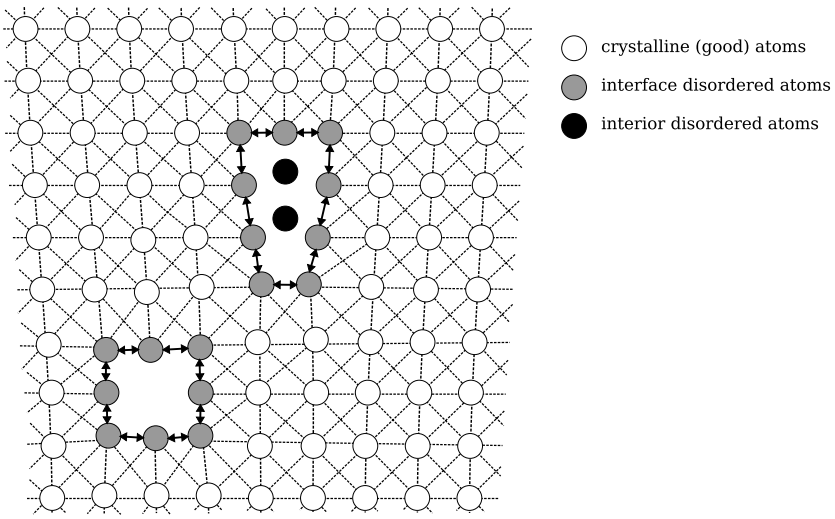

Figure 3. Illustration of the classification of atoms. The depicted crystal contains an edge dislocation and a vacancy. The CNA identifies fully coordinated crystalline atoms (open circles). The set of disordered atoms is further sub-divided into the so-called interface atoms (gray circles), which have at least one bond to a crystalline atom, and interior defect atoms (black circles), which are completely surrounded by other disordered atoms. The interface atoms constitute the nodes of the interface mesh, which is schematically depicted by arrows. The interface mesh completely encloses all crystal defects.

interconnectivity information only; hence, they are effectively independent of the atomic input coordinates! Consequently, the DXA inherits all characteristics of the CNA pertaining to the reliability of identification of crystalline structures. That is, the actual detection of dislocations is not affected by, for instance, thermal displacements or elastic strain fields in any way, as long as they do not impair the identification of interatomic bonds.

In summary, the outcome of the first analysis step is as follows:

(i) Each input atom has been classified as being either crystalline or disordered.
(ii) The crystalline atoms have been divided into independent crystallite clusters.
(iii) For each crystalline atom we have obtained
- a list of its neighbors (14 in the case of bcc),
- the relative lattice vector (from the cluster's imaginary reference crystal) corresponding to each of the 14 bonds.

2.2. Construction of the interface mesh

We first point out a specific feature of the CNA, which we will take advantage of in the construction of the interface mesh: the CNA classifies each atom as being either crystalline (having a complete coordination shell) or disordered. At the interface between crystalline and disordered regions, however, one finds a single layer of atoms, which have been classified as 'disordered' by the CNA, but which are located on well-defined lattice sites (figure 3). We refer to these atoms as 'interface atoms', and they can be characterized as having at least one crystalline atom among their neighbors.

The construction of the interface mesh, which separates the 'good' material from the 'bad' material, is done in the next step. The interface mesh is actually a triangulation of the

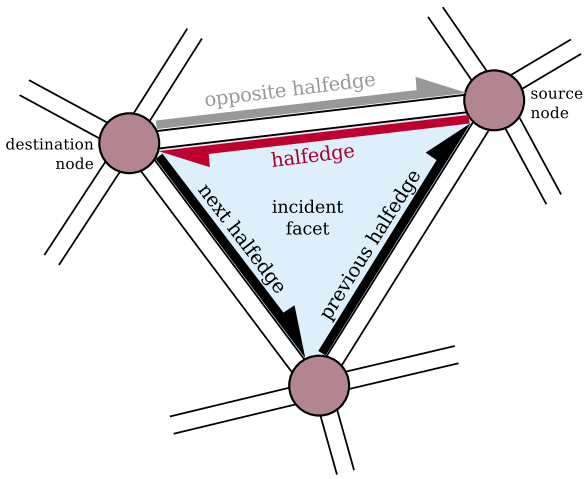

Figure 4. Illustration of the halfedge data structure used to store the interface mesh. Nodes of the interface mesh represent interface atoms as depicted in figure 3.

two-dimensional manifold depicted in figure 1 and is stored as a *halfedge* structure [15, 16]. This combinatorial data structure consists of nodes, halfedges and triangular facets. The following list describes the relationships between these entities, which are illustrated in figure 4.

(i) Triangular facets store a circular sequence of three halfedges.
(ii) Each halfedge connects a source node with a destination node and is associated with exactly one incident facet, which it borders.
(iii) Each halfedge is paired with an opposite halfedge that points from its destination node to its source node.
(iv) Each node stores a list of halfedges that point away from the node.

The nodes of the interface mesh are constituted by the interface atoms, which are located at the boundary of crystal defects as discussed above (see figure 3). Pairs of halfedges between nodes represent bonds between these interface atoms. Since interface atoms are positioned on well-defined lattice sites, we will be able to assign a lattice vector from the imaginary reference crystal to each halfedge as will be described below.

For constructing the interface mesh in bcc crystals, we process each crystalline atom by searching for groups of three or four of its nearest neighbors, which can be connected with a triangular or quadrangular facet as shown in figures 2(*b*) and (*c*). All vertices of such a facet must have previously been classified as interface atoms. Quadrangular facets are split up into two triangular facets. Since we already computed the lattice vectors pointing from the black central atom to the gray interface atoms in the first analysis step, we can now easily calculate the lattice vectors of the halfedges of the interface mesh.

In certain cases, more than one mesh node must be associated with an interface atom. Imagine, for example, a planar crystal defect that consists of a single layer of atoms as shown in figure 5. Here, two mesh nodes per crystal defect atom have to be created, one for each side of the defect, to obtain a topologically valid manifold. In order to determine whether an existing node is shared by two parts of the manifold, one picks an arbitrary facet incident on that node. Then, based on the halfedge data structure, all adjacent facets are traversed in clock-wise order around the node until one arrives at the start facet again. If, during this

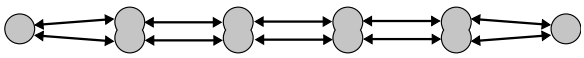

Figure 5. A planar crystal defect consisting of a single layer of interface atoms, which requires the creation of two nodes per interface atom.

traversal, some incident facets have not been visited, they must form a separate part of the manifold and the shared node is duplicated to fix the mesh topology.

After the interface mesh has been constructed, the following information is at our disposal:

(i) A list of mesh nodes and a list of triangular mesh facets connecting these nodes.
(ii) Each mesh facet is bounded by three halfedges, and normal vectors point toward the crystalline region, away from the crystal defects.
(iii) The interface mesh is closed and possesses the properties of an orientable manifold, i.e. each facet has exactly three neighboring facets with their normals pointing to the same side.
(iv) Each halfedge of the mesh is associated with a lattice vector from the imaginary reference crystal.

Figure 6(*a*) exemplarily displays the interface mesh enclosing a dislocation core. Note that from now on, the original input atoms are no longer needed. The actual tracing of dislocation lines will be performed exclusively on the basis of the interface mesh.

2.3. Tracing of dislocations

Each halfedge e of the interface mesh is associated with a vector $\vec{l}(e)$ from the reference crystal lattice. Given some closed path on the interface mesh, consisting of a circular sequence of n halfedges $\{e_1, e_2, ..., e_n\}$, one can calculate the path's Burgers vector by summing up the lattice vectors of the halfedges: $\vec{b} = \sum_{i=1}^{n} \vec{l}(e_i)$. We are only interested in paths (Burgers circuits) that enclose a dislocation segment, i.e. for which $\vec{b} \neq 0$. Such paths can be found using a recursive breath-first search algorithm that traverses all possible halfedge circuits. The search is limited to a small recursive depth N_r, and it is invoked at each node of the mesh until a first circuit with a non-zero Burgers vector is found[2].

The well-known Burgers vector conservation rule states that the Burgers vector cannot change along a dislocation segment, and that a dislocation segment must either merge into a dislocation node or end at an outer surface of the crystal. What happens when an elastic Burgers circuit reaches the end of a dislocation segment as illustrated by figure 7? If the segment ends in a dislocation node then the closed circuit cannot overcome the furcation of the interface mesh at that node. It is inseparably tied to its primary dislocation segment for topological reasons as shown in figure 7. We impose a maximum length on the elastic band to prevent it from overexpanding, that is, the Burgers circuit is no longer advanced after it has reached its maximum expansion. The same applies to the case when a dislocation segment ends in an outer crystal surface or a grain boundary: if the elastic band would be made extremely expandable, then it would move along the outer surface indefinitely (but still yielding the same Burgers vector of the primary dislocation it started from!). Thus, one has to impose an upper limit on the length of the elastic Burgers circuit to stop it when the enclosed crystal defect widens from a thin dislocation core to an extended defect, as for example a surface or a grain boundary.

[2] The maximum search depth, N_r, determines the maximum length of the generated test Burgers circuits, which is given by $n_\mathrm{max} = 2N_\mathrm{r} + 1$. In bcc and fcc crystals, a value of $N_\mathrm{r} = 3$ is sufficient to detect all dislocation segments having a typical core diameter.

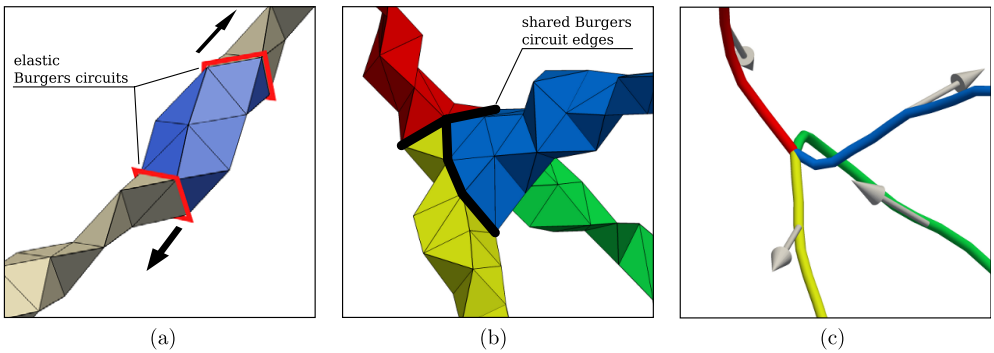

Figure 6. (*a*) Close-up view of the interface mesh enclosing a dislocation. The two Burgers circuits are advanced, facet by facet, in opposite directions to sweep the dislocation segment. (*b*) Interface mesh of a 4-segment node. (*c*) The obtained lines and Burgers vectors representing the junction.

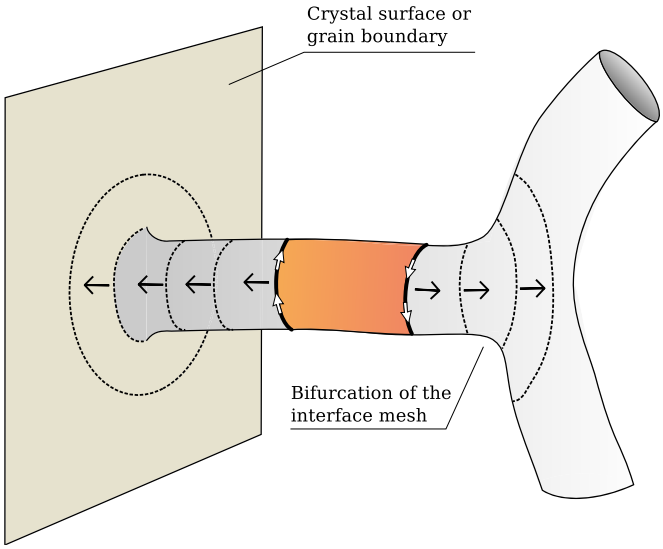

Figure 7. The two elastic Burgers circuits that scan the dislocation segment in both directions hit either a planar crystal defect (e.g. a surface) or a dislocation node, where the interface mesh bifurcates.

2.4. Transition to a network of one-dimensional lines

The diameter of a typical dislocation core fluctuates along the dislocation line, and the Burgers circuit being swept over it has to stretch at the bulky points. The elastic Burgers circuit is moved one or two facets at a time. The set of five elemental replacement operations depicted in figure 8 ensures that the circuit remains as short as possible. A one-dimensional line representation of the dislocation segment is obtained by recording the current center of mass of the Burgers circuit each time the circuit is advanced by one step.

During the advancement phase, it is ensured that Burgers circuits do not interpenetrate other circuits from different dislocation segments, by marking the visited facets of the mesh. This allows us to identify dislocation nodes simply by searching for two or more blocked Burgers circuits that completely cover each other as shown in figure 6(*b*). If multiple circuits

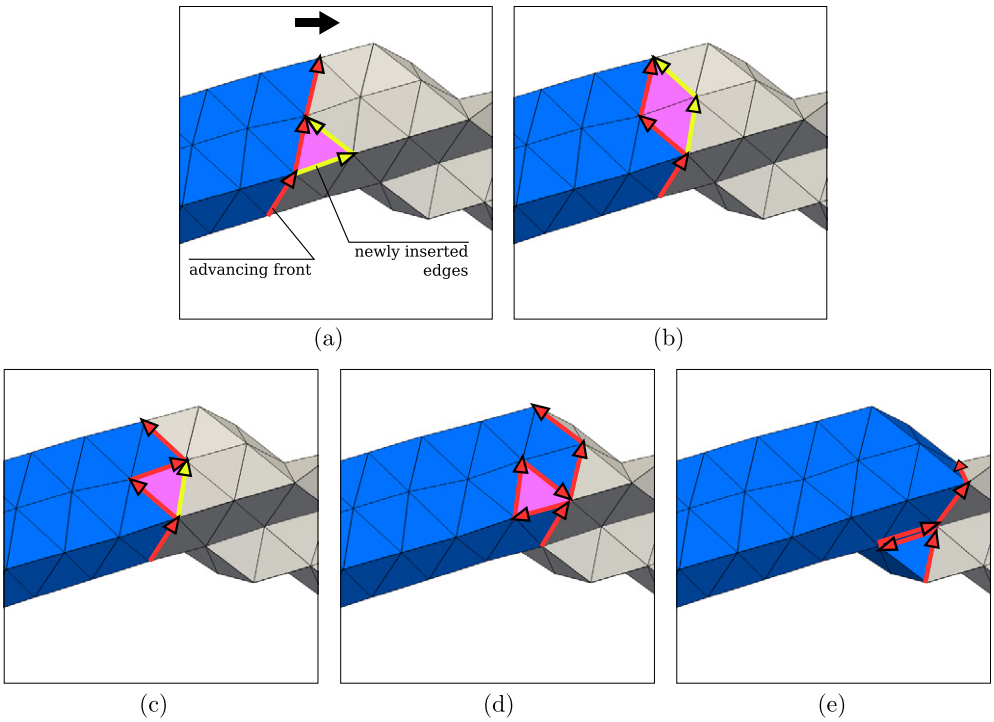

Figure 8. Five elemental replacement operations that are performed to advance the Burgers circuit along the interface mesh enclosing a dislocation segment. Operations that reduce the total number of edges in the circuit are given precedence over operations that extend the circuit. (*a*) Replacing one edge with two new edges (+1), (*b*) Replacing two edges with two other edges (±0), (*c*) Replacing two edges with a single edge (−1), (*d*) Removing three edges (−3) and (*e*) Removing two edges (−2).

have run into each other at a dislocation node then the corresponding one-dimensional line segments are connected in the output as shown in figure 6(*c*). Note that the way we constructed the elastic Burgers circuits guarantees that Burgers vector conservation is fulfilled at every output node.

2.5. Extraction of other crystal defects

Since we impose the maximum Burgers circuit length criterion, we will likely end up with some parts of the interface mesh that have not been swept by a Burgers circuit. These parts form extended crystal defects like surfaces or grain boundaries, which we would not want to be represented by one-dimensional lines. Nevertheless, they are in many cases important for the interpretation of the simulation. That is why we retain these parts of the interface mesh for visualization, whereas all facets, which have been swept by Burgers circuits, get replaced by the corresponding one-dimensional lines. In fact, the interface mesh turns out to be very useful, even for the case of a completely dislocation-free crystal. It can be regarded as a volumetric representation of crystal defects, which could be used for analyses that are not feasible in the atomistic picture.

For visualization purposes, one can smooth the defect surface as well as the dislocation lines using a volume/curvature-preserving low-pass filter algorithm [17].

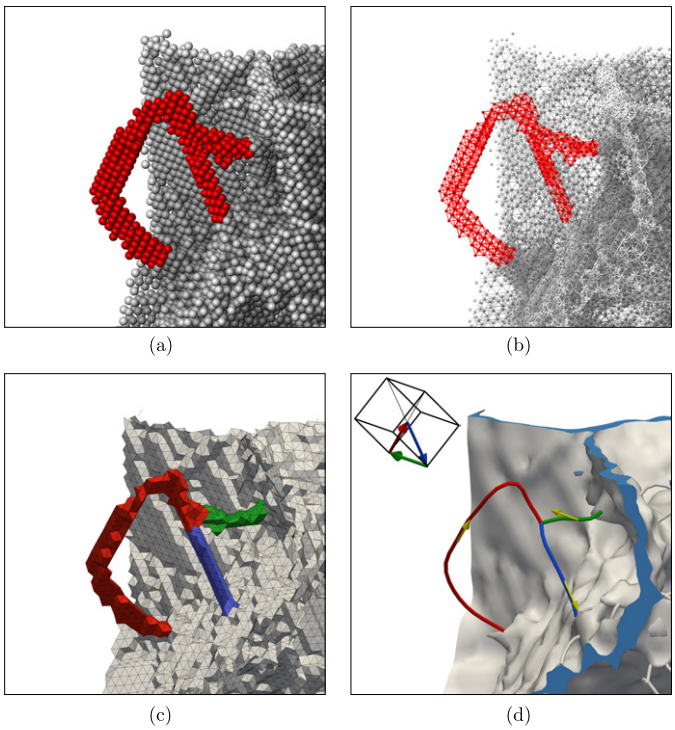

Figure 9. Stepwise conversion of three atomistic dislocation cores into a geometric line representation. (*a*) Atomistic input data. (*b*) Bonds between disordered atoms. (*c*) Interface mesh. (*d*) Smoothed output.

3. Example

Figure 9 shows an example demonstrating the complete transition process from the atomistic input data to the final dislocation lines. The input is a snapshot taken from an MD simulation of a nanocrystalline tungsten structure, which is being deformed in a simulated tensile experiment. The first picture of figure 9 displays a section of the atomistic input, which contains three dislocation segments (red atoms) that have been nucleated from a nearby grain boundary junction (gray atoms). Note that the CNA has been applied to remove ideal bcc atoms in this picture, and dislocation core atoms have been highlighted manually for the sake of clarity. For the CNA, the cutoff radius $R_{\mathrm{CNA}} = 3.8145$ Å was used, which is halfway between the second and third neighbor shells of tungsten.

Figure 9(*b*) displays the bonds between disordered atoms, which are used to generate the interface mesh. The final interface mesh is shown in figure 9(*c*). The colored parts of the mesh have been swept by Burgers circuits during the dislocation analysis, i.e. they have been identified as dislocation segments. Their colors correspond to the three different Burgers vectors as depicted inside the bcc unit cell in figure 9(*d*). The last figure shows the final results of the analysis algorithm: the one-dimensional dislocation lines, which merge at a nodal point; their Burgers vectors (transformed into the coordinate system of the real crystal) and the defect surface, which represents the surrounding grain boundaries.

4. Summary

We have described a novel method for extracting dislocation lines from atomistic simulation data in a fully automated way. The DXA yields a geometric description of the dislocation network contained in an arbitrary crystal. The network of one-dimensional continuous dislocation lines generated by the algorithm is guaranteed to have the same topology as the real dislocation network. In particular, it always conforms to the Burgers vector conservation rule. The shape of dislocation loops is resolved with up to atomic precision.

All other crystal defects (grain boundaries, surfaces, etc), which cannot be represented by dislocation lines, are converted into a polyhedral representation by the algorithm. This representation is ideal for visualization purposes, even for applications which do not involve dislocations.

The DXA was designed to reliably divide the wealth of information stored in a snapshot of an atomistic simulation into relevant and dispensable parts. The complexity of the relevant part, the crystal defects, is reduced as far as possible by transforming them into a higher-level description, which consists of only dislocation lines (and their Burgers vectors) and defect surfaces. In summary, the DXA provides a simple-to-use and robust method to analyze and visualize atomistic simulations of crystal plasticity.

An implementation of the described algorithm is available from the authors on request.

Acknowledgments

This work was performed with financial support of the Deutsche Forschungsgemeinschaft (FOR714) and computing time grants from HHLR Darmstadt, FZ Jülich and bwGRiD, member of the German D-Grid initiative. Test datasets have been kindly provided by J Schäfer. Three-dimensional visualizations have been created with OVITO [18] and PARAVIEW [19].

References

[1] Burgers J M 1940 Geometrical considerations concerning the structural irregularities to be assumed in a crystal *Proc. Phys. Soc.* **52** 23

[2] Frank F C 1951 LXXXIII. Crystal dislocations—elementary concepts and definitions *Phil. Mag. Ser. 7* **42** 809–19

[3] Bulatov V V and Cai W 2006 *Computer Simulations of Dislocations* (Oxford: Oxford University Press) p 63 chapter 3.3

[4] Honeycutt J D and Andersen H C 1987 Molecular dynamics study of melting and freezing of small Lennard-Jones clusters *J. Phys. Chem.* **91** 4950–63

[5] Kelchner C L, Plimpton S J and Hamilton J C 1998 Dislocation nucleation and defect structure during surface indentation *Phys. Rev.* B **58** 11085

[6] Ackland G J and Jones A P 2006 Applications of local crystal structure measures in experiment and simulation *Phys. Rev.* B **73** 054104

[7] Zimmerman J A, Kelchner C L, Klein P A, Hamilton J C and Foiles S M 2001 Surface step effects on nanoindentation *Phys. Rev. Lett.* **87** 165507

[8] Hua J and Hartmaier A 2010 Determining Burgers vectors and geometrically necessary dislocation densities from atomistic data *Modelling Simul. Mater. Sci. Eng.* **18** 045007

[9] Hartley C S and Mishin Y 2005 Characterization and visualization of the lattice misfit associated with dislocation cores *Acta Mater.* **53** 1313–21

[10] Abraham F F, Walkup R, Gao H, Duchaineau M, Diaz De La Rubia T and Seager M 2002 Simulating materials failure by using up to one billion atoms and the world's fastest computer: work-hardening *Proc. Natl Acad. Sci. USA* **99** 5783–7

[11] Stukowski A and Albe K 2010 Dislocation detection algorithm for atomistic simulations *Modelling Simul. Mater. Sci. Eng.* **18** 025016

[12] Stukowski A, Farkas D and Albe K 2010 Nanotwinned fcc metals: strengthening vs. softening mechanisms *Phys. Rev.* B submitted

[13] Schäfer J, Stukowski A and Albe K 2010 Plastic deformation of nanocrystalline Pd–Au alloys: on the interplay of grain boundary solute segregation, fault energies and grain size *Acta Mater.* submitted
[14] Hirth J P and Lothe J 1982 *Theory of Dislocations* (New York: Wiley) 2nd edn
[15] Weiler K 1985 Edge-based data structures for solid modeling in curved-surface environments *IEEE Comput. Graph.* **5** 21–40
[16] Mäntylä M 1987 *An Introduction to Solid Modeling* (New York: Computer Science Press)
[17] Taubin G 1995 A signal processing approach to fair surface design *Computer Graphics Proc. SIGGRAPH 95 (Los Angeles, CA)*
[18] Stukowski A 2010 Visualization and analysis of atomistic simulation data with OVITO—the Open Visualization Tool *Modelling Simul. Mater. Sci. Eng.* **18** 015012 Software available at http://ovito.org/
[19] Kitware, Inc. ParaView data analysis and visualization application