Computer Physics Communications 273 (2022) 108260

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

A brute-force code searching for cell of non-identical displacement for CSL grain boundaries and interfaces ☆,☆☆

Yaoshu Xie, Kiyou Shibata, Teruyasu Mizoguchi *

Institute of Industrial Science, The University of Tokyo, 4-6-1, Meguro-ku, Tokyo, 13-8505, Japan

ARTICLE INFO

Article history:
Received 13 July 2021
Received in revised form 30 November 2021
Accepted 12 December 2021
Available online 23 December 2021

Keywords:
Synopsis
Interfaces
Grain boundaries
Bicrystallography
Coincident-site lattice
Displacement shift complete
Cell of non-identical displacement

ABSTRACT

Atomic simulation of coincidence-site lattice (CSL) grain boundaries (GBs) and interfaces is of importance for understanding GBs in polycrystalline materials and for making functional films. A common process of high-throughput simulation for CSL GBs and interfaces is to explore rigid body translation (RBT) of one crystal respect to the other. Cell of non-identical displacement (CNID) is the minimum cell including all non-identical RBTs in the GB or interface plane and is important for effective sampling. This work proposes an algorithm to compute the CNID of any two-dimensional CSL GB or interface based on a reciprocity relation between the displacement shift complete (DSC) and CSL.

Program summary

Program title: cnidcal
CPC Library link to program files: https://doi.org/10.17632/wzy67jd56d.1
Code Ocean capsule: https://codeocean.com/capsule/1648641
Licensing provisions: MIT license
Programming language: Python

Nature of problem

Determination of atomic structure of crystalline interfaces and grain boundaries is a key issue for understanding the properties of polycrystalline materials, where atomic simulation has made notable contribution in this field. Simulation of crystalline interfaces often applies a bicrystal model of a coincidence-site lattice (CSL) [1, 2] interface comprising two slabs of crystals which coincides in the interface plane. To determine the most stable structure of this interface model, a main task is often to tailor the initial bicrystal model to obtain many non-identical and non-relaxed structures for subsequent relaxation to explore the energy landscape of the system to find the 'global minimum' at 0 K, where one of the most important operations to tailor the model is applying rigid body translation (RBT) [1] of one crystal respect to the other by a vector confined in the interface plane. Because of symmetry of the two crystals, there exist periodicity of this translation vector which is reflected by a special cell called 'cell of non-identical displacement' (CNID) [1] which is the minimum cell including all non-identical choices of this vector and is key for effective sampling. Although CNID has a straightforward definition that it is a lattice including both the two overlapping two-dimensional lattices, finding a method capable to determine a primitive basis of CNID in general cases can be non-trivial and there has not been a reported programme capable to do so.

Solution method

CNID in nature is a displacement shift complete (DSC) lattice [2]. While there has not been effective reported programme to compute DSC, several programmes have been made to compute CSL [3, 4, 5]. Besides, Grimmer [6] has proposed a mathematical relationship between the DSC and the CSL. In this way, based on a previously reported programme computing the CSL by brute-force method [5] and the relationship between DSC and CSL proposed by Grimmer, we generated a python code to compute CNID. This code is available to compute the CNID of a CSL interface comprising any two lattices if only they

☆ The review of this paper was arranged by Prof. Weigel Martin.

☆☆ This paper and its associated computer program are available via the Computer Physics Communications homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author.
E-mail addresses: ysxie@iis.u-tokyo.ac.jp (Y. Xie), teru@iis.u-tokyo.ac.jp (T. Mizoguchi).

https://doi.org/10.1016/j.cpc.2021.108260

coincide even including complex cases involving heterogeneous lattices with difference in both shape and size.

References

[1] Sutton A.P., *Interfaces in Crystalline Materials,* Clarendon Press, Oxford (1995).
[2] Bollmann, W. General Geometrical Theory of Crystalline Interfaces. In: *Crystal Defects and Crystalline Interfaces*, pp. 143–185 (Springer Berlin Heidelberg, 1970).
[3] Ogawa, Hiroshi. GBstudio: a builder software on periodic models of CSL boundaries for molecular simulation. *Materials transactions* 47(11) (2006) 2706–2710.
[4] Cheng, Jianli, Jian Luo, Kesong Yang. Aimsgb: An algorithm and open-source python library to generate periodic grain boundary structures. *Computational Materials Science* 155 (2018) 92–103.
[5] Hadian, R., Grabowski, B., Neugebauer, J. GB code: A grain boundary generation code, *Review*, *Repository*, *Archive* (2018).
[6] Grimmer, H. A reciprocity relation between the coincidence site lattice and the DSC lattice. *Scr. Metall.* 8 (1974) 1221–1223.

1. Introduction

Interfaces are of importance in materials science not only because the interfaces consisting of heterogeneous materials can generate properties which does not belong to the bulk of each them, but also because its common existence in polycrystalline materials known as grain boundaries (GBs) plays a significant role in affecting many mechanical and functional properties [1–4]. Atomic-scale experimental observation has remarkably improved the understanding of interface structures and properties [5–10], while computational work has also made notable contribution especially in GBs. Examples of reported computational work for GBs include 1) investigating how the GB geometric parameters like disorientation between grains and GB plane orientation affect GB energy [11–14], GB segregation energy [15] and GB mobility [16]; 2) exploring GB phases and simulating phase transition processes [17–19]; 3) understanding the GB faceting behavior [20,21]; and 4) evaluating the stabilization effects of GB segregation [22,23].

Many studies on GBs apply a bicrystal model which combines two disoriented perfect identical crystals to form a *Coincident Site Lattice* (CSL) GB [1,24–26] where the two overlapping two-dimensional lattices lying in the GB plane coincides. Selecting such CSL GBs for simulation is suitable for applying periodic boundary condition. During the simulation process, a main task is often to tailor the initial bicrystal model to obtain many non-identical and non-relaxed GB structures for subsequent relaxation to explore the GB energy landscape and to obtain the most stable GB structure at 0 K. The operations to tailor the bicrystal often includes applying rigid body translation (RBT) of one crystal respect to the other and others like deleting atoms near the GB which are too close [27]. RBT is an important operation and it is necessary not only in traditional methods for GB structure optimization like the γ-surface method [1], but also in novel methods including Monte-Carlo method [28], generic algorithm [29], and machine learning [14,30,31].

The effects from applying RBT of one crystal respect to the other on varying the interface structure was firstly discussed by Pond [5] on a tilt CSL GB of aluminium, where the RBT was decomposed to be expressed by two vectors $\mathbf{e}$ and $\mathbf{p}$. $\mathbf{e}$ is normal to the GB and $\mathbf{p}$ is confined in the GB plane. RBTs by $\mathbf{e}$ and $\mathbf{p}$ adjust the GB structure in different ways. While RBT by $\mathbf{e}$ simply adjusts the length of vacuum spacing the two crystals, RBT by $\mathbf{p}$ changes the overlapping two-dimensional pattern that it can introduce remarkable variation of the GB structure. According to Pond [5], if the GB structure possesses a two-dimensional periodicity in the GB plane, $\mathbf{p}$ has translational symmetry in cases that the translation by $\mathbf{p}$ does not relocate the position of GB plane so that it should be reduced into a two-dimensional minimum symmetric unit (MSU). When the GB plane contains only coincident sites (for symmetric tilt GBs), MSU is the Wigner-Seitz (W.-S.) cell of the two-dimensional atomic arrangement of one of the crystals' planes. For more general cases where the GB is not a symmetric tilt GB and its GB plane possesses non-coincident sites, MSU is a displacement shift complete (DSC) lattice [26] of the two overlapping two-dimensional lattices terminated at the GB plane. Note that in the former case, W.-S. cell is identical to the DSC lattice. Considering that the DSC is no larger than the Wigner-Seitz (W.-S.), this DSC lattice was also termed as cell of non-identical displacement (CNID) [1]. CNID is important to make effective simulation of CSL interfaces in general cases and it is also essential for understanding other interface-involved concepts and properties such as the GB kinetics [32].

Although CNID belonging to a kind of DSC lattice has a straightforward definition that it is a lattice including both the two disoriented lattices [26], finding a method capable to determine a primitive basis of CNID in general cases can be non-trivial. In some simple cases where the two lattices forming the interface are different only in orientation (e.g. for symmetric tilt and twist GBs), a primitive basis of CNID can be found by directly getting the two shortest non-collinear distance vectors between the two lattices. However, as will be shown in the next part, this direct method can be problematic in some complex cases and there exists another more effective method. In this work, we generated a python code applying a brute-force method to compute CNID based on a property of DSC lattice [33]. This code is available to compute the CNID of any two lattices which are capable to form a CSL lattice, which includes both simple cases like CSL GBs consisting of homogeneous crystals and complex cases of CSL interfaces comprising heterogeneous crystals with difference in shape and size. We will show that this method is more effective than directly using the definition of DSC, and that applying the computed CNID can dramatically promote the efficiency in simulation of many interfaces.

2. Theory and algorithm

Once a CSL interface is constructed, there exist two overlapping two-dimensional lattices lying in the interface plane with their basic vectors as $[\boldsymbol{a_1}, \boldsymbol{b_1}]$ and $[\boldsymbol{a_2}, \boldsymbol{b_2}]$ (see Fig. 1 (a)) and our task is to compute the DSC of these two lattices. Generally, the two primitive bases should be in 'good condition' which means the two primitive vectors should be short and as orthogonal as possible. Fig. 1 (a) illustrates an example of the two-dimensional DSC of which the primitive basis cannot be obtained directly by a simple form of difference vectors including $\boldsymbol{a_1} \pm \boldsymbol{a_2}$, $\boldsymbol{a_1} \pm \boldsymbol{b_2}$, $\boldsymbol{b_1} \pm \boldsymbol{a_2}$ or $\boldsymbol{b_1} \pm \boldsymbol{b_2}$, and it involves a complex form of linear combination of these four basic

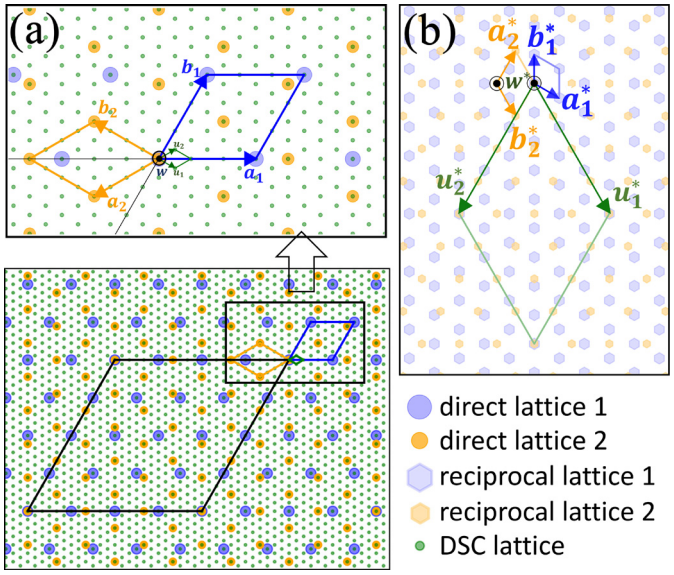

Fig. 1. Visualization of all the cells applied in the computation; (a) The two plane bases and the CSL and DSC of them ($[\boldsymbol{u}_1\ \boldsymbol{u}_2]$ have been LLL-reduced so that the angle between them is less than 90 degrees now); (b) The two reciprocal lattices and their CSL. This visualization is included in the code.

vectors, which is $[2\boldsymbol{a_1}-\boldsymbol{b_1}+2\boldsymbol{b_2}, \boldsymbol{a_1}+\boldsymbol{b_1}-2\boldsymbol{a_2}]$. From this example, we can see that using the definition of DSC to find a DSC primitive basis needs to explore, apart from the simplest ones, more complex linear combinations involving four vectors and to select the two shortest ones by ranking their lengths, which is straightforward but not effective. Firstly, after ranking all existing vectors, one has no idea whether there still exist other shorter vectors because when four basic vectors form linear combinations, those with large absolute value coefficients can be shorter than those with small absolute value coefficients (e.g. in this case $\boldsymbol{a_1}+\boldsymbol{b_1}-2\boldsymbol{a_2}$ is shorter than $\boldsymbol{a_1}\pm\boldsymbol{a_2}$, $\boldsymbol{a_1}\pm\boldsymbol{b_2}$, $\boldsymbol{b_1}\pm\boldsymbol{a_2}$ or $\boldsymbol{b_1}\pm\boldsymbol{b_2}$). Also, each time one extends the exploring range, when more than one new vectors are found to be shorter than the old minimum vector, a ranking operation needs to be done again. As well-known, ranking is a multi-step process which is time-consuming.

Apart from the definition, there exist other properties of DSC capable to be applied to determine it. Grimmer [33] proposed a reciprocity relation that the DSC formed by two sets of basic vectors $\wedge_1$ and $\wedge_2$ is the reciprocal lattice of the CSL lattice formed by $\wedge_1{}^*$ and $\wedge_2{}^*$, where * indicates the reciprocal lattice. This reciprocity relation can be applied for computing an arbitrary DSC because it brings a more 'well-defined' method which is to find the CSL of two reciprocal lattices. Here we present a brute-force method inspired by the code in the package gb_code[34] to use this reciprocity relation to compute a primitive basis of DSC of any two two-dimensional CSL-producing lattices $[\boldsymbol{a_1},\ \boldsymbol{b_1}]$ and $[\boldsymbol{a_2},\ \boldsymbol{b_2}]$. According to the definition, the computed DSC is also the CNID of the corresponding CSL interface made by two slabs with these two two-dimensional lattices lying in the interface.

Firstly, we introduce an auxiliary vector with $\mathbf{w} = 1/|\boldsymbol{a_1}|$ $(\boldsymbol{a_1}\times\boldsymbol{b_1})$ to form two lattices $\boldsymbol{P_1}=[\mathbf{w},\boldsymbol{a_1},\boldsymbol{b_1}]$ and $\boldsymbol{P_2}=[\mathbf{w},\boldsymbol{a_2},\boldsymbol{b_2}]$ and compute their reciprocal lattices $\boldsymbol{P_1^*}=[\mathbf{w^*},\boldsymbol{a_1^*},\boldsymbol{b_1^*}]$ and $\boldsymbol{P_2^*}=[\mathbf{w^*},\boldsymbol{a_2^*},\boldsymbol{b_2^*}]$ (see Fig. 2 (a, b)). Then, we compute the CSL of $\boldsymbol{P_1^*}$ and $\boldsymbol{P_2^*}$ by the following routine (all the vectors involved here are column vectors).

1) Generate a set of non-zero vectors $\{\boldsymbol{t_i}\}$ spanned by basis $[\boldsymbol{a_1^*},\boldsymbol{b_1^*}]$, and get a set of coefficient vectors $\{\boldsymbol{p_i}\}$ to express $\{\boldsymbol{t_i}\}$ in the $\boldsymbol{P_2^*}$ frame.

$$\boldsymbol{t_i} = y_i\boldsymbol{a_1^*} + z_i\boldsymbol{b_1^*}(y_i, z_i \in \mathbb{Z}, |y_i|+|z_i|\neq 0)$$

$$\boldsymbol{p_i} = \boldsymbol{P_2^*}^{-1}\boldsymbol{t_i}$$

The number of generated vectors can be limited by a *lim* where $|y_i| < lim$, $|z_i| < lim$.

2) Get subset $\{\boldsymbol{p_j}\}$ in $\{\boldsymbol{p_i}\}$ so that $\{\boldsymbol{p_j}\}$ only have integer elements, and get a set of vectors $\{\boldsymbol{q_j^*}\}$ that

$$\boldsymbol{q_j^*} = \boldsymbol{P_2^*}\boldsymbol{p_j}$$

$\{\boldsymbol{q_j^*}\}$ is a set of vectors belonging to CSL of $\mathbf{P_1^*}$ and $\mathbf{P_2^*}$. From $\{\boldsymbol{q_j^*}\}$, we find two shortest non-collinear vectors $\boldsymbol{u_1^*}$, $\boldsymbol{u_2^*}$ by ranking the length of $\boldsymbol{q_j^*}$. Finally, with $[\boldsymbol{w},\boldsymbol{u_1},\boldsymbol{u_2}] = [\mathbf{w^*},\boldsymbol{u_1^*},\boldsymbol{u_2^*}]^*$, $[\boldsymbol{u_1},\boldsymbol{u_2}]$ is a desired primitive basis of the DSC (CNID). After finding this primitive basis, the Lenstra–Lenstra–Lovász (LLL) [35] lattice reduction algorithm is applied to make the two primitive vectors short and as close to orthogonal as possible (see Fig. 1 (a)).

As can be seen, compared with the method directly finding two shortest vectors which involves ranking the length of linear combination of four basic vectors, this method only involves ranking the length of $\boldsymbol{q_j^*}$ which are in a subset of linear combination of two reciprocal primitive vectors, and apart from this ranking operation, all other operations involved are one-step operations of matrix computation or parallel logic computation. Besides, the process finding integer vectors $\boldsymbol{p_j}$ can help to judge whether an enough number of vectors by linear combination of $\boldsymbol{a_1^*}$ and $\boldsymbol{b_1^*}$ have been generated. In this way, one does not need to at the beginning generate large number of vectors $\boldsymbol{t_i}$ by a large *lim* for a successful searching. Instead, one can increase *lim* gradually until the integer vectors $\boldsymbol{p_j}$ appear, which is fast because all the computations before doing ranking are one-step. Also, because $\boldsymbol{q_j^*}$ is linear combination involving only two primitive reciprocal vectors which are often in good condition, once two shortest vectors in $\boldsymbol{q_j^*}$ are found within a small *lim*, it is not likely to find new smaller vectors within a larger *lim*. This assures that the basis found by this method are primitive. Therefore, although this reciprocal method involves more processes than the direct method, it is more effective in terms of promoting speed of computation and assuring primitive basis.

For a convenient usage, we developed a code based on this computation for DSC allowing users to obtain the CNID of any interfaces by inputting some parameters easily accessible including

1). Matrices with columns as primitive vectors of Lattice 1 $\boldsymbol{L_1}$ and Lattice 2 $\boldsymbol{L_2}$;

2). Rotation matrix R applied to $\boldsymbol{L_2}$ to make a CSL;

3). Miller indices of the interface plane expressed in the lattice 1.

The flowchart of the code is shown in Fig. 2. In this code, we also made an algorithm to convert Miller indices expressed in different lattice bases (see S3), and in the step 2, we utilized the algorithm in [36] to find the two plane bases in the interface plane. For users only knowing the Miller indices expressed in the conventional lattice but want to compute the CNID of the primitive lattices, they can use the algorithm of Miller indices conversion to convert that Miller indices to be expressed in the primitive lattice. We also provide computation for the CSL lattice, by which one is capable to compute the CNID of any interfaces lying in a CSL lattice plane. For example, one can specify a interface to lie in the (3 2 1) plane of the CSL lattice, and convert these indices to be expressed in the primitive lattice 1 and input this converted indices to compute the CNID of this CSL interface, which is quite flexible.

We need to mention that our computation for CNID only considered the translational symmetry and does not include point symmetry. The computed CNID might be further reduced by its point symmetry (Pond and Bollmann, 1979 [38], Pond and Vlachavas, 1983 [37]). For multiple-motif systems, different terminating planes at the interface can result in different point symmetry of CNID and thus producing a family of CNIDs (see examples

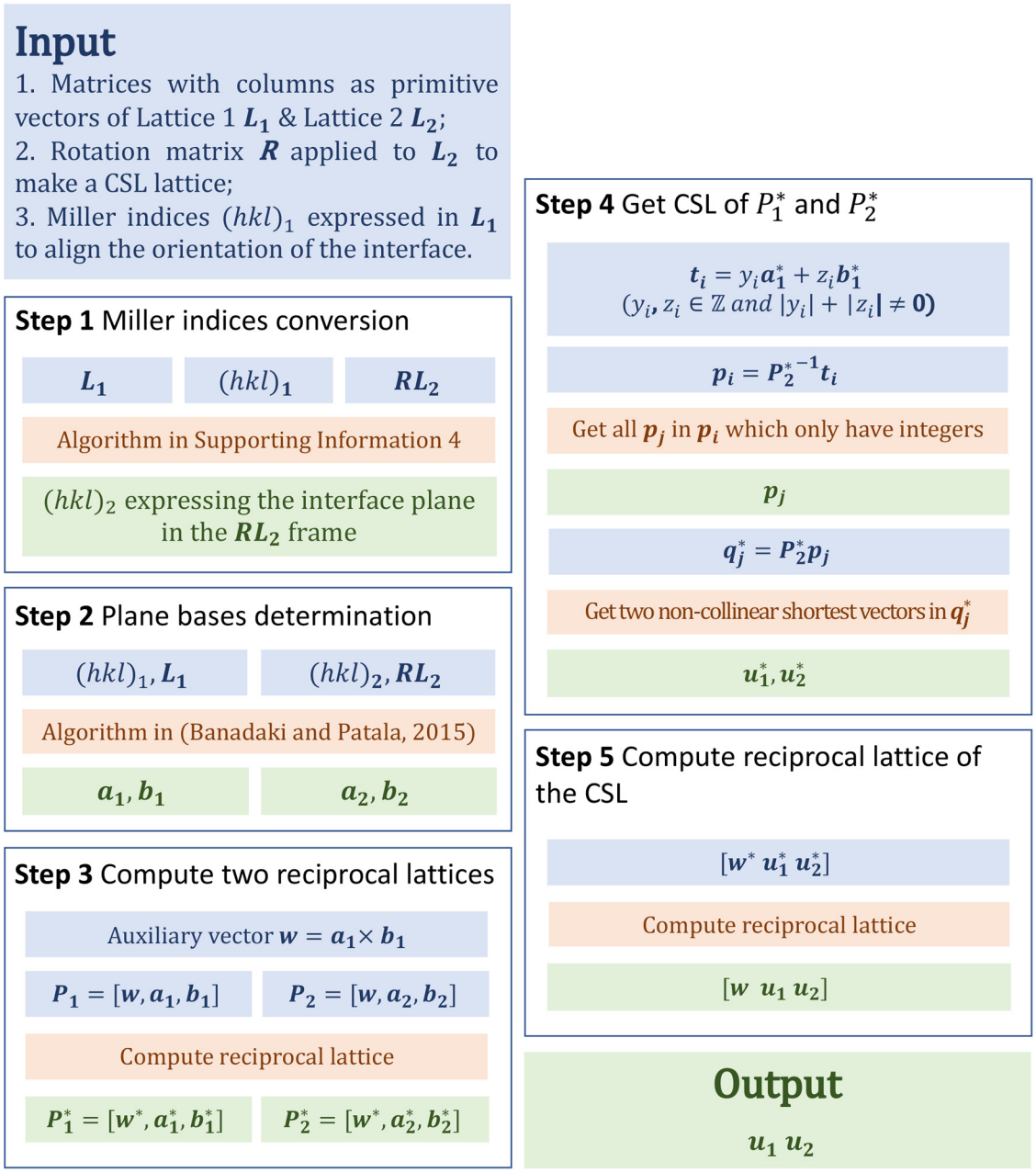

Fig. 2. Algorithm flowchart for computing CNID, with blue blocks as inputs, orange blocks as algorithms or operations and green blocks as outputs. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

in Pond et al., 1991 [39]). One can refer to the literature to further reduce the CNID computed by our code. However, as mentioned later, in some cases applying the non-reduced CNID has already made a notable reduction of time need to explore RBT.

3. Results and discussion

3.1. CNID of cubic GBs

To evaluate the efficiency of applying CNID for exploring RBT in simulation of GBs, we computed the CNID for a series of GBs based on fcc and bcc lattices. Some of our computed CNIDs are identical to those reported in Pond and Bollmann, 1979 [38], Pond and Vlachavas, 1983 [37]. For convenience, we firstly settle two identical perfect crystals in the global cartesian coordinates system with its conventional lattice $a\boldsymbol{e}$ ($\boldsymbol{e}$ is a 3×3 unit matrix and a is the lattice constant). Then, by rotating one crystal respect to axis <$\boldsymbol{g}$> by a certain angle, we determined disorientations up to Σ 99: <111> (Σ 3, Σ 7, Σ 19, Σ 21, Σ 31, Σ 39, Σ 43, Σ 49, Σ 57, Σ 67, Σ 79, Σ 91, Σ 93), <100> (Σ 5, Σ 13, Σ 17, Σ 25, Σ 29, Σ 37, Σ 41, Σ 53, Σ 61, Σ 65, Σ 73, Σ 85, Σ 89, Σ 97) and <110> (Σ 9, Σ 11, Σ 17, Σ 19, Σ 27, Σ 33, Σ 41, Σ 43, Σ 51, Σ 57, Σ 59, Σ 67, Σ 73, Σ 81, Σ 83, Σ 89, Σ 97, Σ 99). We applied the concept of 'plane orientation fundamental zone' [40] to determine a series of twist, symmetric tilt, asymmetric tilt and mixed GBs. Specifically, for each disorientation, we specify a CSL lattice $\mathbf{C} = [\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{g}]$, ($|\boldsymbol{t_1}| \leq |\boldsymbol{t_2}|$), where $\boldsymbol{t_1}$ and $\boldsymbol{t_2}$ are two minimum non-collinear primitive CSL vectors normal to the rotation axis $\boldsymbol{g}$), and then we construct CSL GBs by specifying the GB plane normal vector as a linear combination of $\boldsymbol{t_1}$, $\boldsymbol{t_2}$ and $\boldsymbol{g}$, which ended up with 100 CSL GBs which were divided into 7 different groups as showed in Table 1.

The simulation cell of a bicrystal model of each specified GB can be orthogonal, or monoclinic depending on whether the two

Table 1
Computed GBs.

Group	rotation axis	GB normal	GB type
1	<100>	$\mathbf{t}_1$	
2	<100>	$\mathbf{t}_1 + \mathbf{t}_2$	symmetric tilt (STGB)
3	<111>	$\mathbf{t}_1$	
4	<110>	$2\mathbf{t}_1 \pm \mathbf{t}_2$	asymmetric tilt (ASGB)
5	<100>	$\mathbf{g}$	twist (TSTGB)
6	<111>	$\mathbf{g}$	
7	<100>	$2\mathbf{t}_1 + \mathbf{t}_2 + \mathbf{g}$	mixed (MXDGB)

non-collinear CSL vectors in the GB plane are perpendicular or not. We specify the two minimum perpendicular CSL vectors in the GB plane as $\mathbf{v}_{Rt1}$, $\mathbf{v}_{Rt2}$ and two minimum non-collinear ones as $\mathbf{v}_{Ol1}$, $\mathbf{v}_{Ol2}$ ($\mathbf{v}_{Ol1}$, $\mathbf{v}_{Ol2}$ can be perpendicular or non-perpendicular). From these two sets of vectors, we obtain one rectangular cell **Rt** = [$\mathbf{v}_{Rt1}$, $\mathbf{v}_{Rt2}$], and one oblique cell **Ol** = [$\mathbf{v}_{Ol1}$, $\mathbf{v}_{Ol2}$]. Many previous studies conducting GB simulation sampled RBTs in **Rt** or **Ol**, which can be non-effective because they can include many identical structures. To show the difference between sampling RBTs in **Rt, Ol** and CNID, we computed these three different cells belonging to every CSL GB listed in Table 1. The areas of CNID, **Rt** and **Ol** are specified as A_c, A_r and A_o, obviously with $A_c \leq A_r \leq A_o$. The details of all the computed GBs are listed in S1. Fig. 3 shows some examples of different cases with (a) $A_c = A_r = A_o$, (b) $A_c < A_r = A_o$, (c) $A_c = A_r < A_o$, and (d) $A_c < A_r < A_o$. The ratio of A_o/A_r depends on the shape of **Ol**; and the ratio of A_c/A_o is proportional to the ratio of the number of coincident sites to the number of all the sites (denominated as GB sites) in the two lattice planes lying in the GB. Fig. 4 compares the A_c, A_r and A_o of all the fcc GBs listed in Table 1. For the <100> symmetric-tilt GBs (group 1 & group 2), CNID is identical to **Ol** because all the GB sites are coincident sites. Some <111> symmetric-tilt GBs (group 3) and <110> asymmetric-tilt GBs (group 4) present a smaller area of CNID than **Ol** as there exist non-coincident GB sites. The CNID of GBs in group 5, 6 and 7, with twist components, is much smaller than the **Ol** and **Rt**, where most of the GB sites are non-coincident sites. Similar results were also found in the computed bcc GBs (see S1). According to the results, to increase the efficiency of high-throughput simulation applying RBT, applying CNID can dramatically reduce the time cost especially for asymmetric-tilt, twist and mixed GBs.

To verify whether the CNIDs have been correctly computed, we computed the non-relaxed GB energy distribution of all the listed GBs by LAMMPS [41] of fcc Al, bcc W and diamond-structured Si (applied CNID of its fcc lattice) in a 2 × 2 unit cell of CNID vectors which is divided into a 50 × 50 mesh. We applied a EAM potential for Al [42], a cut-off modified Tersoff potential for Si [43] and a EAM potential for W [44]. The CSL GBs were built with the gb_code package [34], where we did some modification to make it available for generating monoclinic simulation cells. Every resulting computed pattern of energy landscape has 2 × 2 periodicity which verified the accuracy of the computed CNIDs (see S2).

This example of usage for these many fcc and bcc GBs shows an important application of the code to help research demanding in simulating large number of GBs. Relevant topics include exploring the structure-property relationships exploring the five macroscopic degrees of freedom [12,17] and machine learning applied in GBs [14,15] which has attracted increasing interests recently. These topics require to do structure optimization for many GBs to explore the whole GB space and to make enough training data. This code can effectively accelerate research on them because although one can find the CNID manually for a single simple case, the manual work is cumbersome when large number of GBs were to be computed without programming. This code provides much convenience simply using the same parameters required by a traditional programme making CSL GBs. This is even important for complex

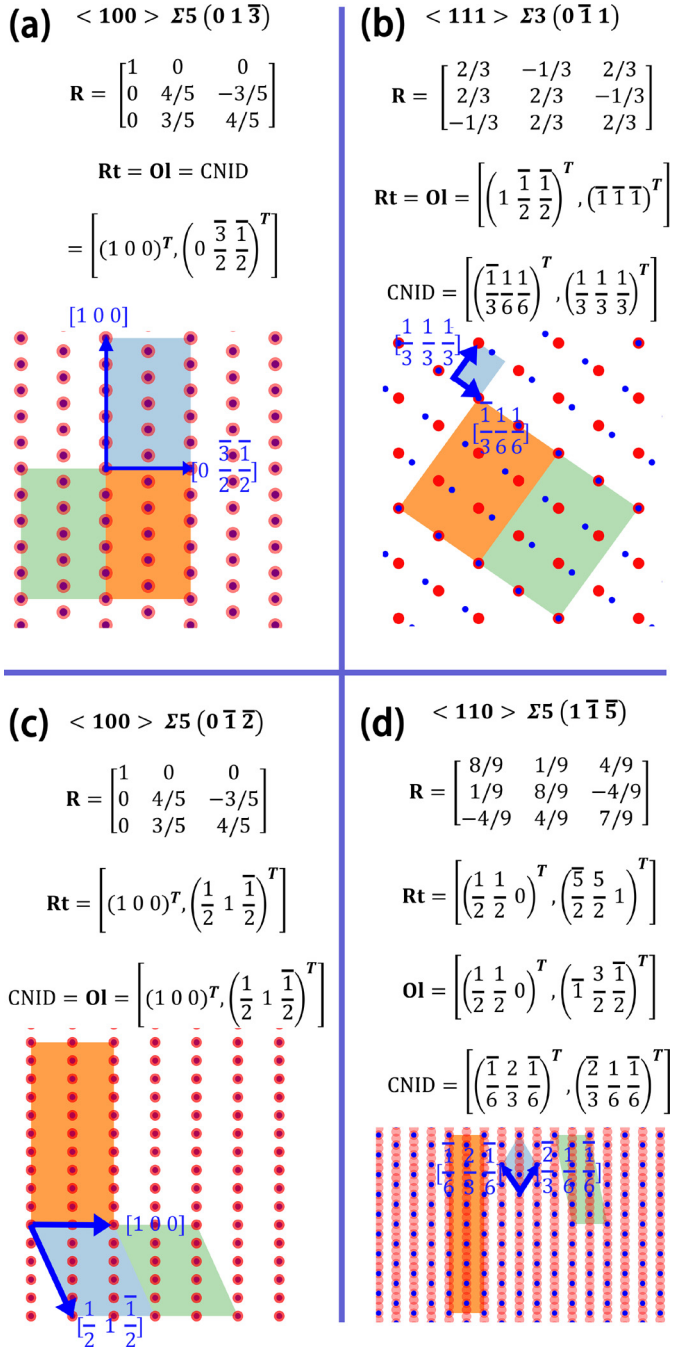

Fig. 3. Examples of computed **Rt** (rectangular cell, orange), **Ol** (oblique cell, green) and CNID (blue).

GBs where the CSL is large and CNID is small because the traditional programmes building CSL GBs often only compute the **Rt** or **Ol** cell and because the CNID in this case becomes difficult to determine manually, one might choose to conduct sampling the RBT in the whole **Rt** or **Ol** cell whose area is often many times of the CNID and thus involving much unnecessary cost. When the CNID becomes quite narrow, the GB structure is not likely to remarkably vary through doing RBT and in this case RBT becomes less important. In this way, the sampling cost should be proportional to the CNID area with a specific grid size. With this code, one can conveniently achieve a smart and consistent sampling convention by applying a two-dimensional grid dividing the two computed CNID vectors (such as 0.1 angstrom of grid length for each CNID vector) to automatically handle the difference in the size of CNIDs.

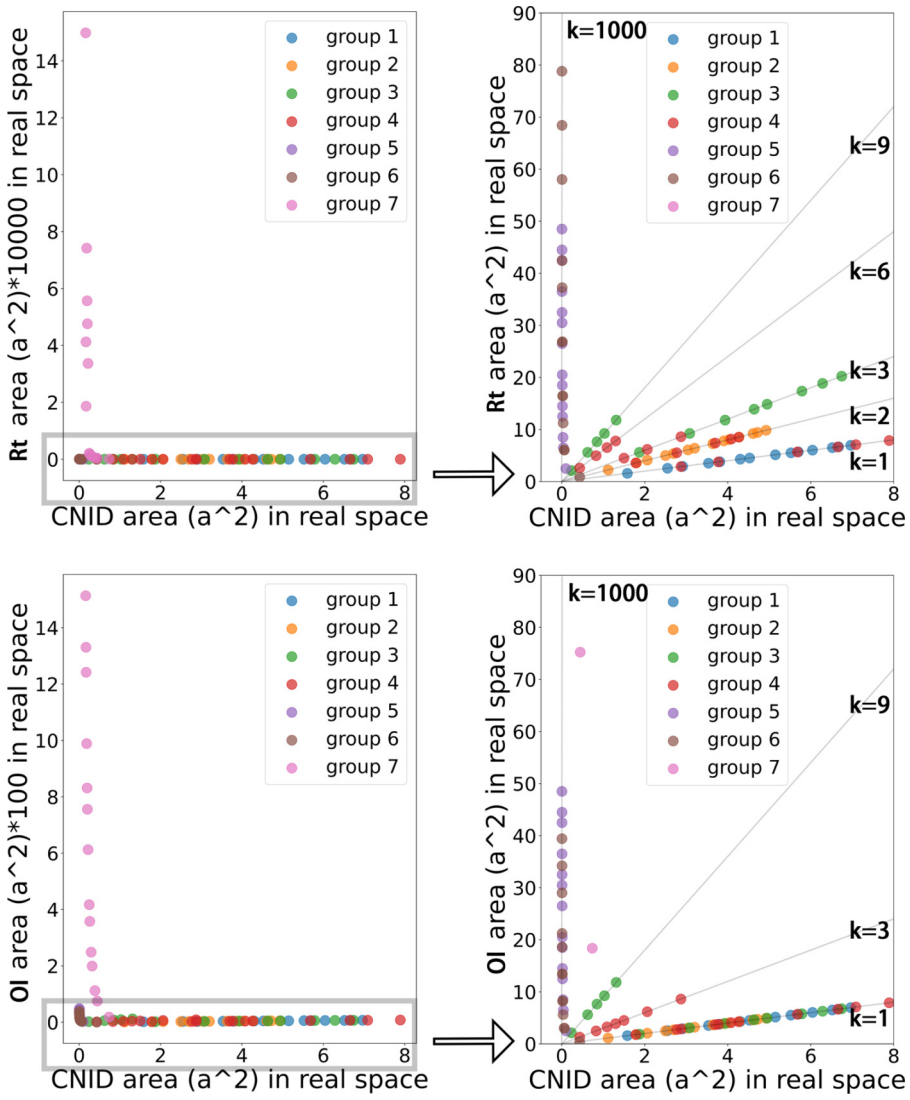

Fig. 4. Areas of **Rt**, **OI** and CNID of fcc GBs; black lines are guides for eyes and the value k is the slope corresponding to the factor of the area relative to the area of CNID.

3.2. CNID of non-cubic interfaces made by non-identical lattices

An advantage of our code is that it can be applied for any two identical lattices if only they form a two-dimensional CSL lattice. Here we show an example computing the CNID of an Si(111)/SiC(0001) approximate CSL interface. Si(111)/SiC(0001) interface has been experimentally observed [45,46] which is likely to act as the transistor wafer with high thermal conductivity [47], and to determine the performance of C/SiC ceramic matrix composite [48]. Fig. 5 (a) shows that an approximate CSL bicrystal is built for this interface by combining two supercells made from the primitive cells of Si and SiC (provided by the cif files in S4). To make this approximate CSL interface, the two vectors in the basal plane of the hcp SiC were deformed by a factor of 0.9619. Fig. 5 (b) illustrates the lattice energy of the whole bicrystal without relaxation corresponding to RBTs in a 2×2 expansion of the computed CNID, which shows a 2×2 periodicity and thus verifying that the CNID is correct. Computation of this energy landscape is done by using LAMMPS with a Tersoff potential [49]. The computed CNID is $[\frac{1}{18}[1\bar{2}1], \frac{1}{18}[\bar{2}11]]$ expressed in the fcc-diamond conventional lattice of Si and $[\frac{1}{4}[21\bar{1}0], \frac{1}{4}[1\bar{2}10]]$ expressed in the conventional hcp lattice of SiC. Compared with sampling in the minimum repeating cell of the Si lattice or SiC lattice lying in the interface, using a non-reduced CNID can increase the speed

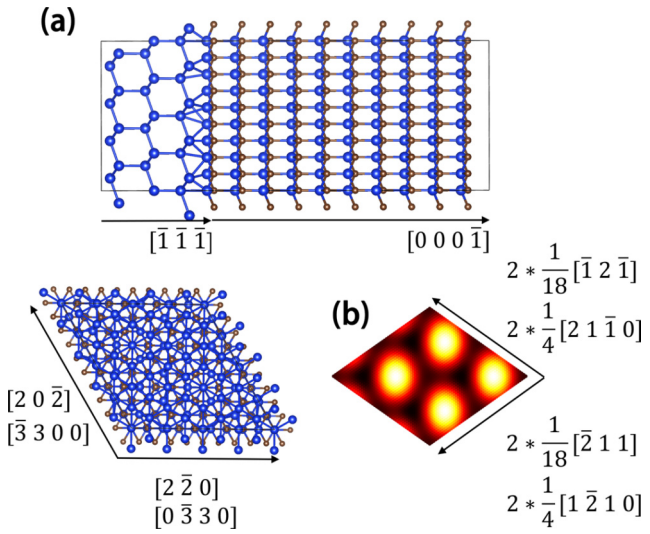

Fig. 5. (a) the bicrystal of an approximate CSL interface of (111)Si/(0001)SiC; (b) the potential energy of the non-relaxed bicrystal with RBT in the 2×2 CNID. The three-element indices express one side of the cell by a vector in the conventional fcc-diamond lattice and the four-element express one by the conventional hcp lattice. White colour represents high energy.

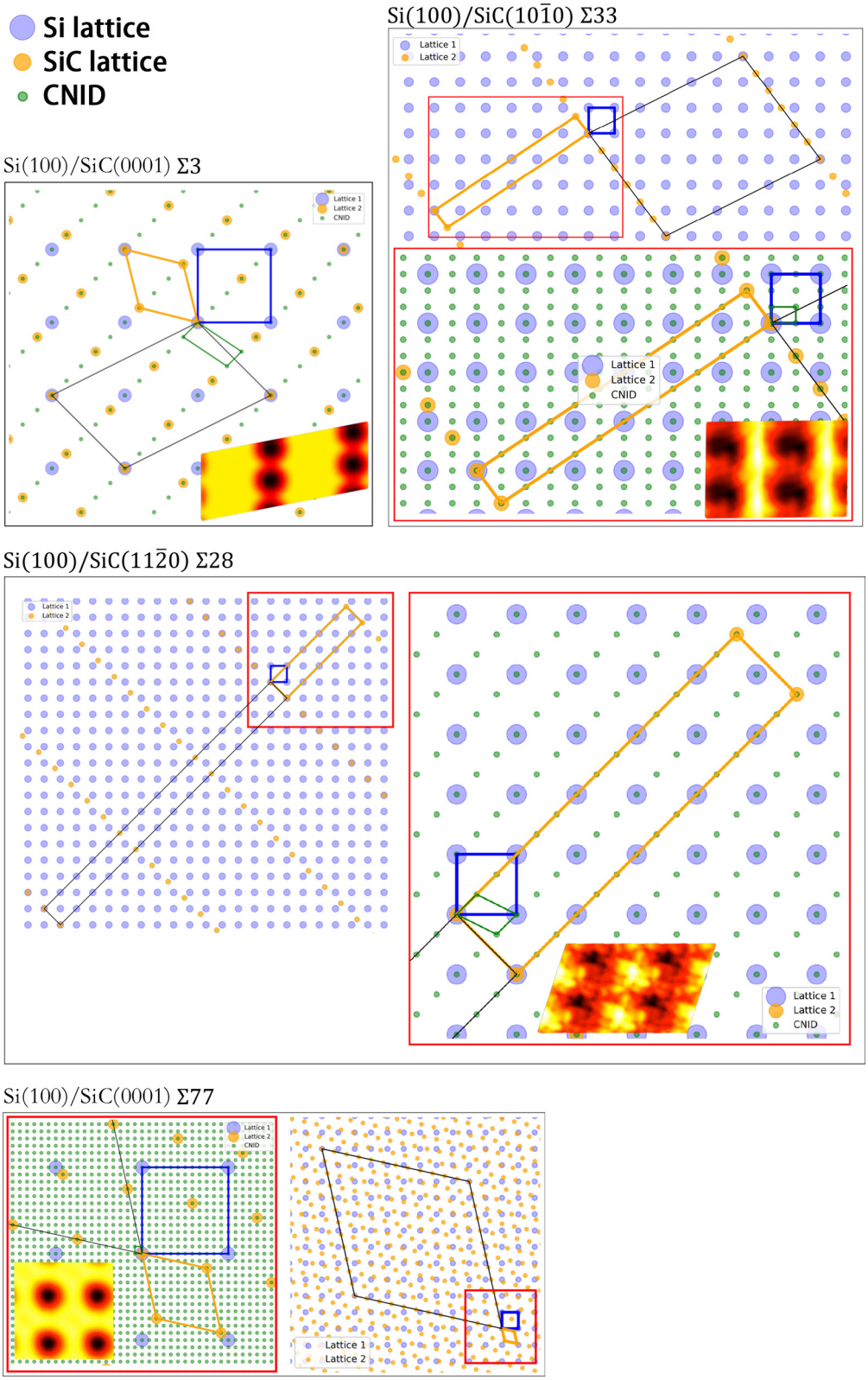

Fig. 6. Usage examples on Si/SiC interfaces. Blue frames represent the primitive silicon lattice; orange frames represent the primitive SiC lattice; black frames represent the CSL. Colormap is the non-relaxed total lattice energy sampled with RBT in the 2x2 CNID. White colour represents high energy.

by 27 to 16 times. Fig. 6 illustrated more examples of the computed CNIDs of the Si/SiC interfaces with different orientations. As can be seen, determining their CNIDs manually can be complex. The input generating these CNIDs are provided in the Tutorial folder of the *cnidcal* repository published in Github https://github.com/nmdl-mizo/cnidcal.

Here we have shown an example of this code being applied to interfaces comprising two different crystals. In fact, this code in principle can be applied to any general cases of CSL interfaces. For those desiring to simulate non-CSL interfaces, one effective way is to make an approximate CSL interface by introducing deformation to the two crystals, as what we have done for the presented Si(111)/SiC(0001) interface where the SiC was shrunk to form a CSL interface with Si. One can regard the approximate CSL interface as an approximate projection of the structure of the infinite non-CSL interface to a periodic structure. When the deformation is small enough, the deformed model should reflect similar properties with the original one [26], and it is also an appropriate model for perfectly deposited epitaxial films. Besides the application on epitaxial films, another reason one might prefer an approximate CSL model to an exact non-CSL model is that for complex interfaces one often needs to rely on first-principle simulation which is costing to simulate large supercells and thus requiring at least a two-dimensional period boundary condition to be applied in the interface plane. For non-CSL interfaces, the CNID shrinks to a single point because the interface structure cannot vary by RBT confined in the interface plane; while for CSL interfaces, the CNID always spans a non-zero area and one needs to explore RBT in this CNID for structure optimization. Therefore, for non-CSL interfaces, one can still apply a CSL model to simulate it and utilize this code to compute the corresponding CNID for structure exploration. For this purpose, we have also developed another package which is capable to build an approximate CSL interface of two arbitrary materials within allowed deformation. We will make that code available for public usage once this article is accepted.

4. Conclusions

The objective of this article is to compute the CNID of two coplanar primitive two-dimensional lattices $\mathbf{L_1} = [\mathbf{a_1}, \mathbf{b_1}]$ and $\mathbf{L_2} = [\mathbf{a_2}, \mathbf{b_2}]$ lying in a two-dimensional CSL GB plane. Based on a reciprocity relation between the CSL and DSC lattices, a numerical searching algorithm by brute-force method is generated for computation. The computed CNIDs were verified to be accurate both for cubic GBs consisting of homogeneous crystals and for non-cubic interfaces consisting of heterogeneous crystals. This code is available for any two lattices if only they form a CSL lattice. Through the present study, we have demonstrated that applying the CNID computed by the present code dramatically reduces the number of necessary operations of RBT on a bicrystal model to explore all non-identical configurations, especially for twist, asymmetric tilt and mixed CSL GBs; and heterogeneous CSL interfaces.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Yaoshu Xie was supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT) Scholarship with Embassy Recommendation for Chinese students. This work was supported by JST-PRESTO (JPMJPR16NB 16814592) and the (MEXT); Nos. 17H06094, 19H00818, and 19H05787.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cpc.2021.108260.

References

[1] P.A. Sutton, in: Interfaces in Crystalline Materials, Clarendon Press, Oxford, 1995.
[2] G.S. Rohrer, J. Mater. Sci. 46 (2011) 5881–5895.
[3] T. Watanabe, J. Mater. Sci. 46 (2011) 4095–4115.
[4] S. Patala, Comput. Mater. Sci. 162 (2019) 281–294.
[5] R.C. Pond, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 357 (1977) 471–483.
[6] Y. Ikuhara, P. Thavorniti, T. Sakuma, Acta Mater. 45 (1997) 5275–5284.
[7] W.D. Kaplan, Acta Mater. 46 (1998) 2369–2379.
[8] A. Avishai, C. Scheu, W.D. Kaplan, Acta Mater. 53 (2005) 1559–1569.
[9] H. Hojo, et al., Nano Lett. 10 (2010) 4668–4672.
[10] K. Inoue, M. Saito, Z. Wang, M. Kotani, Y. Ikuhara, Mater. Trans. 56 (2015) 1945–1952.
[11] M.A. Tschopp, D.L. McDowell, Philos. Mag. 87 (2007) 3871–3892.
[12] D.L. Olmsted, S.M. Foiles, E.A. Holm, Acta Mater. 57 (2009) 3694–3703.
[13] E.R. Homer, S. Patala, J.L. Priedeman, Sci. Rep. 5 (2015) 1–13.
[14] H. Oda, S. Kiyohara, T. Mizoguchi, J. Phys. Mater. 2 (2019) 34005.
[15] L. Huber, R. Hadian, B. Grabowski, J. Neugebauer, npj Comput. Mater. 4 (2018) 64.
[16] D.L. Olmsted, E.A. Holm, S.M. Foiles, Acta Mater. 57 (2009) 3704–3713.
[17] T. Frolov, D.L. Olmsted, M. Asta, Y. Mishin, Nat. Commun. 4 (2013) 1899.
[18] T. Frolov, et al., Nanoscale 10 (2018) 8253–8268.
[19] Q. Zhu, A. Samanta, B. Li, R.E. Rudd, T. Frolov, Nat. Commun. 9 (2018) 1–9.
[20] R. Hadian, B. Grabowski, M.W. Finnis, J. Neugebauer, Phys. Rev. Mater. 2 (2018) 43601.
[21] A.D. Banadaki, S. Patala, Comput. Mater. Sci. 112 (2016) 147–160.
[22] F. Abdeljawad, S.M. Foiles, Acta Mater. 101 (2015) 159–171.
[23] J.L. Priedeman, G.B. Thompson, Acta Mater. 201 (2020) 329–340.
[24] D.G. Brandon, Acta Metall. 14 (1966) 1479–1484.
[25] S. Ranganathan, Acta Crystallogr. 21 (1966) 197–199.
[26] W. Bollmann, in: Crystal Defects and Crystalline Interfaces, Springer, Berlin Heidelberg, 1970.
[27] J. Hickman, Y. Mishin, RAPID Commun. Phys. Rev. Mater. 1 (2017) 10601.
[28] A.D. Banadaki, M.A. Tschopp, S. Patala, Comput. Mater. Sci. 155 (2018) 466–475.
[29] Q. Zhu, A. Samanta, B. Li, R.E. Rudd, T. Frolov, Nat. Commun. 9 (2018) 1–9.
[30] S. Kiyohara, H. Oda, K. Tsuda, T. Mizoguchi, Jpn. J. Appl. Phys. 55 (2016) 45502.
[31] H. Oda, S. Kiyohara, K. Tsuda, T. Mizoguchi, J. Phys. Soc. Jpn. 86 (2017) 123601.
[32] J. Han, S.L. Thomas, D.J. Srolovitz, Prog. Mater. Sci. 98 (2018) 386–476.
[33] H. Grimmer, Scr. Metall. 8 (1974) 1221–1223.
[34] R. Hadian, B. Grabowski, J. Neugebauer, J. Open Sour. Softw. 3 (29) (2018) 900.
[35] A.K. Lenstra, H.W. Lenstra, L. Lovász, Math. Ann. 261 (1982) 515–534.
[36] A.D. Banadaki, S. Patala, J. Appl. Crystallogr. 48 (2015) 585–588.
[37] R.C. Pond, D.S. Vlachavas, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 386 (1983) 95–143.
[38] R.C. Pond, W. Bollmann, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 292 (1979) 449–472.
[39] R.C. Pond, D.J. Bacon, A. Serra, A.P. Sutton, Metall. Trans. 22 (1991) 1185–1196.
[40] E.R. Homer, S. Patala, J.L. Priedeman, Sci. Rep. 5 (2015) 15476.
[41] S. Plimpton, J. Comput. Phys. 117 (1995) 1–19.
[42] Y. Mishin, D. Farkas, M.J. Mehl, D.A. Papaconstantopoulos, Phys. Rev. B, Condens. Matter Mater. Phys. 59 (1999) 3393–3407.
[43] G.P.P. Pun, Y. Mishin, Phys. Rev. B 95 (2017) 224103.
[44] M.C. Marinica, et al., J. Phys. Condens. Matter 25 (2013) 395502.
[45] M. Yoshimoto, R. Araki, T. Kurumi, H. Kinoshita, ECS Trans. 50 (2013) 61.
[46] L. Li, Heterojunct. Nanostruct. (2018) 67.
[47] H. Shinohara, H. Kinoshita, M. Yoshimoto, Appl. Phys. Lett. 93 (2008) 122110.
[48] R.H. Kraft, J.F. Molinari, Acta Mater. 56 (2008) 4739–4749.
[49] P. Erhart, K. Albe, Phys. Rev. B, Condens. Matter Mater. Phys. 71 (2005) 035211.