ANDRZEJ DANILUK JANUARY 2022

rheed++ 2.0 User guide

The proposed framework approaches the modelling of RHEED experiments from two different points of view, i.e. the functional and data perspectives. The functional perspective focuses on the services provided by *rheed++ 2.0* whereas the data perspective (*growth++*) is used to characterize the data sources of the framework.
The perspective of the data (calculating the layer coverages as a function of their growth time) can be expressed by a completely different model introduced by the user.

To work properly, *rheed++ 2.0* requires two files containing input data. The first one (*inputRheedData.dat*) contains the simulation parameters of the RHEED experiment, the second one (*coverage.dat*) contains layer coverage values for the selected growth model. The *coverage.dat* file is generated by *growth++* program where no changes have been made in comparison to the earlier version: https://github.com/ElsevierSoftwareX/SOFTX_2020_278.

1. The growth++ code

The *growth++* code https://github.com/ElsevierSoftwareX/SOFTX_2020_278 is distributed in the form of *.cpp* file and one header file. The *growth++.cpp* contains all functionalities of the program. The header file contains:

- *tm.h* – aliases and templates used in the code.

The *growth++.cpp* is written in C++11, therefore one can use GNU g++ compiler or any C++11 compliant compiler to build the executable program. For example, under Linux use the g++ compiler: g++ -std=c++11 growth++.cpp -o growth++, and run the program: ./growth++

1.1. Input data

Input data are read from the *inputGrowthData.dat* text file where the user provides information on the parameters of the growth simulation, i.e. growth model:

0 - diffusive growth,

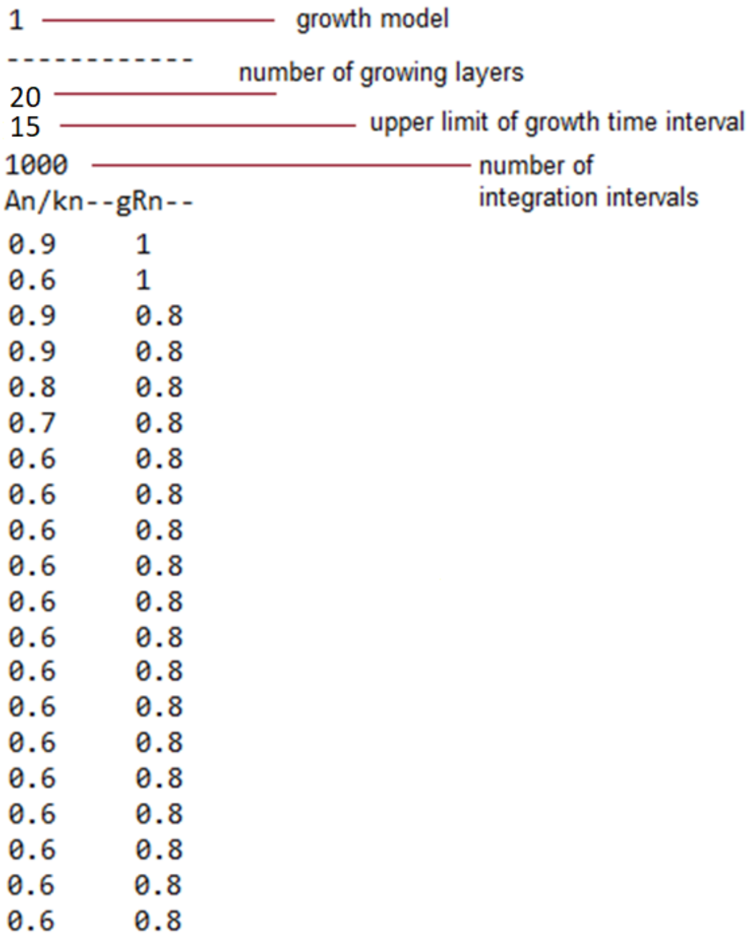
1 - distributed growth-variant 1,

2 - distributed growth-variant 2,

3 - distributed growth-variant 3,

the number of growing layers (*numLayers*), the upper limit of growth time interval (*tMax*), the number of integration intervals (*numIntervals*), the values of A_n (or k_n) parameters, and corresponding values of the gR_n (growth rate for each layer, See https://doi.org/10.1016/j.softx.2020.100593).

Sample contents of the input file *inputGrowthData.dat*, for variant 1 of the distributed growth model:

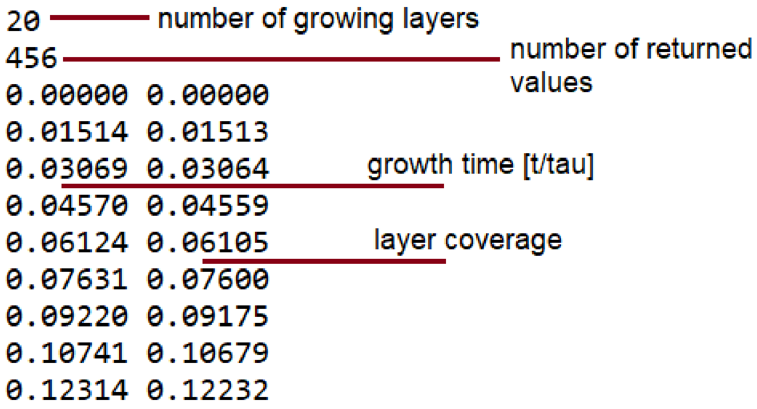


```
1
------------
20
15
1000
An/kn--gRn--
0.9     1
0.6     1
0.9     0.8
0.9     0.8
0.8     0.8
0.7     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
0.6     0.8
```

1.2. Output data

Output data are written on a disc in the current directory in the form of *coverage.dat* text file. The data in this file are saved in two columns: the first column stores the values of growth time, the second – the values of layer coverages.

Sample contents of the output file *coverage.dat*:



```
20
456
0.00000 0.00000
0.01514 0.01513
0.03069 0.03064
0.04570 0.04559
0.06124 0.06105
0.07631 0.07600
0.09220 0.09175
0.10741 0.10679
0.12314 0.12232
```

1.3. The diffusive growth

The process of the diffusive growth is described by the following set of coupled differential equations:

$$\frac{d\theta_n}{dt} = gR_n\left(\theta_{n-1}-\theta_n\right)+k_n\left(\theta_{n+1}-\theta_{n+2}\right)\left(\theta_{n-1}-\theta_n\right)-k_n\left(\theta_n-\theta_{n+1}\right)\left(\theta_{n-2}-\theta_{n-1}\right), \qquad (1)$$

subject to the initial conditions: $\theta_0(t)=1$, $\theta_n(0)=0$, and $\theta_\infty(t)=0$.

In these equations θ_n is the coverage ratio of the *n*th surface layer. k_n is the parameter that determines the coverage profile of the growing film surface. Large values of k_n give rise to an effective mass transport from higher layers to lower layers. For large values of k_n the diffusive growth model can be used for modelling the growth of the Frank–Van der Merwe (FV) type. The growth rate *gR*$_n$ is *1/τ*$_n$, where τ_n is the deposition time of n*th* layer.

1.4. The distributed growth

In this model, the outer edges of the films are taken into consideration, and the atoms which reach the area of the layers will be divided according to the number of possible places of interaction. The process of the distributed growth is described by the following set of coupled differential equations:

$$\frac{d\theta_n}{dt} = gR_n\left[\left(\theta_{n-1}-\theta_n\right)+\alpha_n\left(\theta_n-\theta_{n+1}\right)-\alpha_{n-1}\left(\theta_{n-1}-\theta_n\right)\right], \qquad (2)$$

subject to the initial conditions: $\theta_0(t)=1$, $\theta_n(0)=0$ for $n \geq 1$, and $\theta_\infty(t)=0$.

The parameter α_n is defined by:

$$\alpha_n = A_n \frac{d_n(\theta_n)}{d_n(\theta_n)+d_{n+1}(\theta_{n+1})}. \qquad (3)$$

$d_n(\theta_n)$ is the perimeter of the *n*th film, and A_n is the phenomenological parameter that measures the net rate of transfer from one layer to the next. For A_n close to 1.0 this model gives perfect layer-by-layer growth (FV), and for A_n close to 0 it gives non-diffusive growth (growth of the Volmer–Webber type).

1.4.1. Variant 1

In this model we can choose the following dependence:

$$d_n(\theta_n)=\theta_n\sqrt{(1-\theta_n)}, \qquad (4)$$

which corresponds to a growth mode in which both the number and size of the nucleation sites change during film growth.

1.4.2. Variant 2

In this model we can choose the following dependence:

$$d_n(\theta_n)=\theta_n(1-\theta_n), \qquad (5)$$

which corresponds to a growth mode that depends solely on the number of filled and empty places on the surface.

1.4.3. Variant 3

Assuming that in the case of the surface which is less than half filled, there is a definite number of islands and each of them has the same perimeter, and also that in the case of the surface which is more than half filled there is a definite number of empty sites with the same perimeter, the following can be written:

$$d_n(\theta_n) = \theta_n^{p_1} \quad \text{for } \theta_n \langle \theta_c \, , \tag{6a}$$

$$d_n(\theta_n) = (1-\theta_n)^{p_2} \quad \text{for } \theta_n \rangle \theta_c \, , \tag{6b}$$

with:

$$\theta_c^{p_1} = (1-\theta_c)^{p_2} \, , \tag{6c}$$

and $p_1 = p_2 = 0.5$, $\theta_c = 0.5$.

In this model one can notice a considerable flexibility, which is dependent upon the way in which the perimeter of the film is linked to the coverage of this particular film.

2. The rheed++ 2.0 code

The version 2.0 of the *rheed++* code is distributed in the form of *.cpp* file and three header files. The *rheed++.cpp* contains all functionalities of the program. The header files contain:

- *dtp.h* - Doyle and Turner constants parameters of the analytic representation of the electron scattering factors,

- *uc.h* - values of universal physical constants and constants for the crystals under investigation,

- *tm.h* – aliases and templates used in the code.

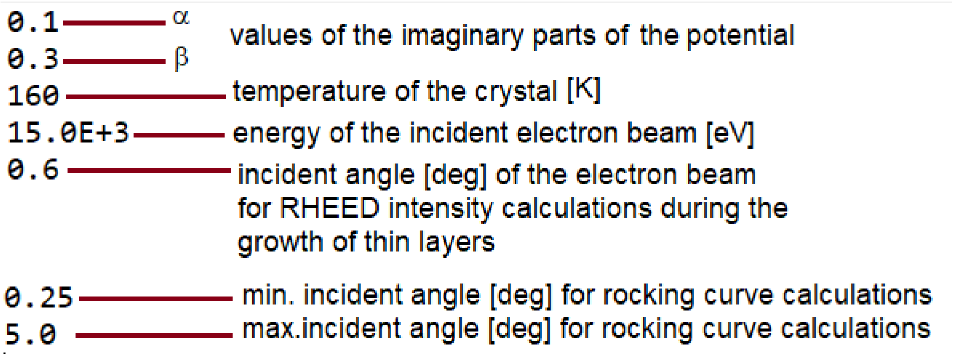
The version 2.0 of *rheed++ .cpp* code is written in C++11, therefore one can use GNU g++ compiler or any C++11 compliant compiler to build the executable program. For example, under Linux use the g++ compiler: g++ -std=c++11 rheed++.cpp -o rheed++, and run the program ./rheed++

2.1. Input data

The presented algorithm allows the user significant freedom in implementation, so (for simplicity of code analysis) the parameters for the crystals under investigation are defined in *uc.h* and *dtp.h* header files.

Input data are read from the *inputRheedData.dat* text file where the user provides information on the parameters of the RHEED simulation, i.e.: values of the parameters α and β that describe the imaginary parts of the potential, temperature of the crystal [K], energy of the incident electron beam [eV], incident angle [deg] of the electron beam for RHEED intensity calculations during the growth of thin layers, and the range of the incident angles (from min. to max.) [deg] of the electron beam for RHEED rocking curve calculations.

Sample contents of the input file *inputRheedData.dat*:



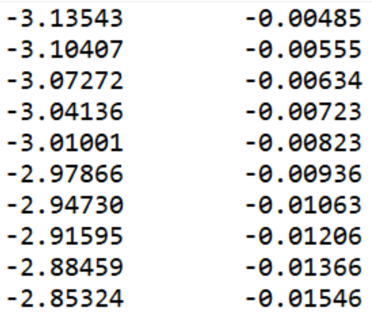
```
0.1
0.3
160
15.0E+3
0.6
0.25
5.0
```

α
β
values of the imaginary parts of the potential
temperature of the crystal [K]
energy of the incident electron beam [eV]
incident angle [deg] of the electron beam for RHEED intensity calculations during the growth of thin layers
min. incident angle [deg] for rocking curve calculations
max.incident angle [deg] for rocking curve calculations

2.2. Output data

The execution generates five text files with results of the calculations:

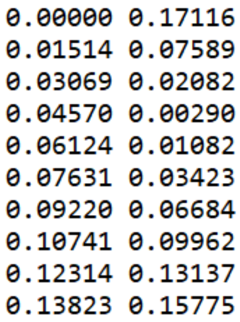
- *potential.dat*, which stores values of the real part of the crystalline potential. The data are organised in two columns: the first column stores the values of the z-axis (expressed in [Å]), the second – the values of the crystal potential (expressed in [eV] units):



```
-3.13543         -0.00485
-3.10407         -0.00555
-3.07272         -0.00634
-3.04136         -0.00723
-3.01001         -0.00823
-2.97866         -0.00936
-2.94730         -0.01063
-2.91595         -0.01206
-2.88459         -0.01366
-2.85324         -0.01546
```

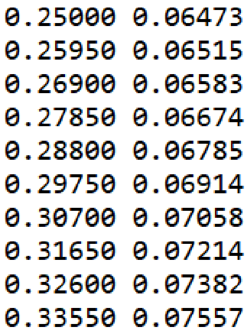
- Files containing the results of the computations.

RHEED intensity oscillations as a function of layer growth time are saved in a file with a name containing the simulation parameters. Data in this file are organised in two columns: the first column stores the values of the growth time, the second – the amplitudes of the reflected wave.



```
0.00000 0.17116
0.01514 0.07589
0.03069 0.02082
0.04570 0.00290
0.06124 0.01082
0.07631 0.03423
0.09220 0.06684
0.10741 0.09962
0.12314 0.13137
0.13823 0.15775
```

Glancing-angle dependences of the RHEED intensity oscillations are saved in a file with the prefix RC- (rocking curves). Data in this file are organised in two columns: the first column stores the values of the glancing angle, the second – the amplitudes of the reflected wave.



```
0.25000 0.06473
0.25950 0.06515
0.26900 0.06583
0.27850 0.06674
0.28800 0.06785
0.29750 0.06914
0.30700 0.07058
0.31650 0.07214
0.32600 0.07382
0.33550 0.07557
```

The normalized data are saved in the files with the prefix norm-.

3. How to run the simulation for the first time

1. Prepare the contents of the *inputGrowthData.dat*

2. Prepare the contents of the *inputRheedData.dat*

3. Compile and run the *growth ++* program:

https://github.com/ElsevierSoftwareX/SOFTX_2020_278

4. Compile and run the *rheed ++* program