CHARACTERISATION OF TiO$_2$ NANOPARTICLES INVOLVING TEM AND IMAGE PROCESSING ANALYSIS

Author: Carole Jonville
Supervisors: Prof. Guillermo Solórzano (at PUC Rio)
           Prof. Marcelo Portes de Albuquerque (at CBPF)

Projet de Fin d’Etudes for ENSPG (France) spent at PUC Rio and CBPF (Brazil) from 19/04/2004 to 12/09/04.
**ABSTRACT**

Interface phenomena, in general, and grain boundary diffusion, in particular, are of paramount importance in nano-scale polycrystalline materials as a whole. Aiming at studying transport properties in consolidated nanostructured oxides, a sample preparation has been started. A characterisation of TiO$_2$ nanoparticles by means of diffraction contrast transmission electron microscopy, based on Bright Field (BF) and High Resolution Electron Microscopy (HREM) images, has been carried out.

The BF images documented at 600000 magnifications showed agglomerations of nanoparticles. These micrographs were used to analyse the particles size repartition with an average size of 25.4 nm. The HREM pictures showed that the powder consists in almost defect free single crystalline particles.

A Radial Distribution Function (RDF) method is proposed to compute atomic lattice characteristics from HREM images, it calculates the histogram of the distances between all the atoms in a limited neighbourhood. The Lattice Parameter Analyser (LPA), an image processing program in Matlab, was developed using this technique. From a good quality HREM image, the LPA gives the value of the first neighbours’ distances and presents the results as a nanometre scaled plot showing peaks corresponding to those distances. The LPA is a powerful tool since it gives statistical results and may be used for any crystalline material HREM image.

**RESUME**

Les phénomènes d’interfaces et en particulier la diffusion aux joints de grains est d’une importance capitale pour les matériaux polycristallins nanostructurés. Dans le but d’étudier les propriétés de transport dans des oxides nanostructurés, une préparation d’échantillons a été débutée avec une poudre de nanoparticules de TiO$_2$. Une caractérisation de ces particules a été faite à partir d’images de Microscopie Electronique en Transmission (images en Champ Clair et images HREM : images de Haute Résolution).

Les premières, à un grandissement de 600 000, ont montré des agglomérations de particules. Elles ont été analysées pour déterminer la répartition de la taille des nanoparticules, la taille moyenne est de 25,4 nm. Les images HREM montrent que la poudre est constituée de particules monocristallines pratiquement absentes de défaut.

Une méthode de *Radial Distribution Function* (RDF) est proposée pour calculer les caractéristiques du réseau atomique des images HREM. Elle permet de calculer l’histogramme des distances entre tous les atomes à l’intérieur d’un voisinage. Le Lattice Parameter Analyser (LPA), un programme d’analyse digitale d’image développé sous Matlab, utilise cette technique. A partir d’une image HREM de bonne qualité, le LPA donne les distances entre premiers voisins et présente les résultats sous la forme d’un graphique montrant les pics correspondant à ces distances. Le LPA est un outil puissant puisqu’il donne des résultats statistiques et peut être utilisé pour n’importe quelle image HREM de matériaux cristallins.
ACKNOWLEDGEMENTS

First, I would like to thank Guillermo Solórzano, my supervisor at PUC Rio. He was the one who proposed this project and without his help I would not have had the opportunity to experience research in one of the most renowned Brazilian university. I thank him for introducing me to the world of academic research and for the experience I gained from the various experiments.

I am also very grateful to Paula Mendes Jardim who shared her experience with all the equipment for the TEM sample preparation, to “Carlão” (Carlos Augusto R. Queiroz) who showed me how to handle the hydraulic press in Casa XXI and Sidnei Pactornik who always took the time to answer my questions about image processing. Also this project would not have been possible without the help of Marcos Venicius Soares Pereira, Ronaldo Pedro da Silva and Paulo Fernando Costa.

I wish to acknowledge Márcio Portes de Alburquerque. He gave me the opportunity to work at CBPF where I gained another experience in a research centre. I thank Aline da Rocha who helped me with Matlab programming.

Finally, I am very grateful to Marcelo Portes de Albuquerque, my supervisor at CBPF. He has always been available to answer my questions and I benefited to a great extend from the numerous discussions about image processing programming. Without his precious help and every day encouragements I would not have succeeded in developing this project. He managed to pass on important experience about scientific research. I also enjoyed the discussions about the Brazilian culture.
PRESENTATION OF THE RESEARCH INSTITUTES AND SUPERVISORS

This project was realised at Pontifícia Universidade Católica of Rio de Janeiro (PUC Rio, Brazil) in the Material Science and Metallurgy Department (DCMM: www.dcmm.puc-rio.br/). This department is dedicated to the graduation and post graduation education as well as research and development. It counts on 16 professors. The infrastructure includes laboratories with last generation equipments for areas such as mineral reduction, environmental technology, thin films, fabrication of composite materials, mechanical behaviour, electronic microscopy, X-Ray diffraction…

My supervisor was Professor Ivan Guillermo Solórzano-Naranjo who obtained his PhD in material sciences at McMaster University (Canada) and worked a year at the LTPCM laboratory of INPG (France). He is the actual director of the Brazilian Material Research Society (SBPMat) and works in collaboration with the Department of Material Science of the MIT (USA).

At DCMM were done all the physical experiments and sample preparation.

A second part of the project was realised in the Brazilian Centre for Research in Physics (CBPF: http://www.cbpf.br/). It is a Research Institute of the Brazilian Ministry of Science and Technology and a pioneer institution in the development of scientific research in theoretical and experimental physics in Brazil.

The work was carried out in the Image and Signal Processing Laboratory (LPDSI: http://www.cbpf.br/cat/pdsi) which is part of the CBPF Technical Activities Department (CAT, responsible for the engineer operation of the academic internet network: Rio Metropolitan Network). The team consists of 5 permanent scientists and 4 students in Master or PhD. Image processing techniques using a new concept of entropy, Magnetic and Atomic Force Microscopy image analysis, DSPs applications for real time signal processing, neuronal networks for vision and pattern recognition systems are examples of research areas. It is a dynamic group that has developed several projects with private companies for technological applications.

The supervisor for this part of the project was Professor Marcelo Portes de Albuquerque, a senior scientist that obtained his PhD from INPG (France) working in the Institut Laue-Langevin of Grenoble.

All the image analysis and program developments were done at LPDSI.
TABLE OF CONTENTS

INTRODUCTION.................................................................................................................. 1
PART I: PHYSICAL EXPERIMENTS AND CHARACTERISATION TECHNIQUES ....2
  1. Composition.................................................................................................................... 2
    1.1. Experiment............................................................................................................... 2
    1.2. Results ..................................................................................................................... 2
    1.3. Conclusion............................................................................................................... 3
  2. TEM Pictures.................................................................................................................. 3
    2.1. Experiment............................................................................................................... 3
    2.2. Results ..................................................................................................................... 4
    2.3. Conclusion............................................................................................................... 5
  3. Sample Preparation......................................................................................................... 5
    3.1. Cold Pressing........................................................................................................... 5
    3.2. Sintering................................................................................................................... 6
    3.3. Grinding................................................................................................................... 7
    3.4. Ion Beam Thinning................................................................................................. 7
    3.5. Conclusion............................................................................................................... 7
  4. Conclusion...................................................................................................................... 7

PART II: PARTICLE’S CHARACTERISATION FROM THE BRIGHT FIELD IMAGES................................................................................................................................. 8
  1. Direct Method................................................................................................................. 8
    1.1. Method..................................................................................................................... 8
    1.2. Results ..................................................................................................................... 8
    1.3. Conclusion............................................................................................................... 9
  2. Size Repartition Automation ........................................................................................ 10
    2.1. Program Description.............................................................................................. 10
    2.2. Results ................................................................................................................... 12
    2.3. Conclusion............................................................................................................. 13
  3. Conclusion.................................................................................................................... 13

PART III: LATTICE CHARACTERISTICS FROM HIGH RESOLUTION ELECTRON MICROSCOPY ANALYSIS................................................................................................................................. 14
  1. Direct Method................................................................................................................. 14
    1.1. Introduction ............................................................................................................ 14
    1.2. Results ................................................................................................................... 14
    1.3. Conclusion............................................................................................................. 16
  2. Automation of the Analysis of HREM Images ............................................................ 16
    2.1. Lattice Generator ................................................................................................. 16
    2.2. Lattice Parameter Analyser ................................................................................... 18
    2.3. LPA Results with Simulated Lattices .................................................................... 23
    2.4. LPA Results with the TiO₂ HREM Images ......................................................... 29
  3. Conclusion.................................................................................................................... 33

CONCLUSION OF THE INTERNSHIP ............................................................................. 34
REFERENCES .....................................................................................................................36
BIBLIOGRAPHY ..................................................................................................................36
APPENDIX A: LATTICE GENERATOR .................................................................37
  ReadMe .........................................................................................................................37
  Configuration File .........................................................................................................39
  Main_LatticeGenerator .................................................................................................40
  LGReadConfigFile ........................................................................................................42
  LGSquare .....................................................................................................................43
  LGHex ..........................................................................................................................44
APPENDIX B: LATTICE PARAMETER ANALYSER ........................................45
  ReadMe .........................................................................................................................45
  Configuration File .........................................................................................................48
  Main_LatticeParameterAnalyser ....................................................................................49
  ReadConfigFile .............................................................................................................51
  ProcessImage ...............................................................................................................52
  Circle ............................................................................................................................53
  DistMeasurement .........................................................................................................54
  FindMaxGauss ...............................................................................................................55
  SavingData ...................................................................................................................57
INTRODUCTION

The present work has been my *Projet de Fin d’Etude* (Final Project) for the ENSPG (Ecole Nationale Supérieure de Physique de Grenoble, France). A part has been done while studying at PUC Rio during the exchange semester of the last year of ENSPG. Then, when the courses ended, it became a full time work (spent both at PUC Rio and CBPF). My objectives in this internship were to confront my abilities, knowledge and competences with the demands of a scientific environment and to enlarge my international experience in an excellent university of South America and an eminent Brazilian research institute.

As I arrived at PUC, I searched an up to date laboratory involved in activities having an actual and growing scientific interest. I found the DCMM Laboratory where new materials featuring exceptional mechanical behaviours based on nanosized microstructures are being studied. A way to elaborate such material is to compact and consolidate powders of nanoparticles [ref 1]. With this method, it is possible to fabricate pellets of titanium oxide (TiO$_2$), obtained from cold pressing and sintering of a nanopowder. Because of the nanoscale of the grains constituting the produced material, the density of grain boundaries is greater than in conventional materials. Therefore, interesting properties may arise from this characteristic and my project was the characterisation of the TiO$_2$ nanopowder. We are interested in measuring, with great accuracy, the size of the powder’s particles. It is also important the identification of its composition and the crystalline parameters.

Conventional characterisation techniques, such as X-ray diffraction and TEM imaging were used. Direct methods of image analysis were applied to extract characteristics from the TEM images. To increase the accuracy of the measurements, an automatic image processing technique is more appropriate.

Part I describes the physical experiments and characterisation techniques involved in this project. The method and results of a direct analysis of the Bright Field images showing the nanoparticles can be found in part II. An attempt to automate the calculation of statistical values extracted from these images is also demonstrated. Finally, Part III presents the analysis of the HREM images and explains a Matlab image processing algorithm in the aim of computing statistical values for the atomic lattice characteristics.
This part was realized at the DCMM laboratory of PUC Rio.

This project included a characterisation part which is the study of the TiO₂ nanoparticles from the original material. Another aim of my work was to begin the sample preparation for the study of the transport properties in general and grain boundary diffusion in particular, in a nanostructured material. Thus pellets were prepared from the nanopowder. It consisted in a compaction of the powder and a sintering. No further heat treatments were done because grain growth would occur and the material would lose its nanostructure properties.

The titanium oxide (TiO₂) powder used for this study was synthesized in the Department of Material Science of MIT (USA), the method used was not revealed to me because it has not been published yet by the elaborating team.

1. Composition

The titanium oxide has 3 possible structures: anatase (space group 141 I4₁₃md [ref 2]), rutile (space group 136 P4₂/nm [ref 3]) and brookite (space group number 61 Pcab [ref 4]). The anatase structure is metastable and is transformed into the rutile structure with heat treatments [ref 1]. The values of the lattice parameters given by the International Centre for Diffraction Data (ICDD) can be found in Table 2.

To check the purity and the amount of anatase, rutile and brookite phases present in this powder, an X-ray analysis was carried out.

1.1. Experiment

A powder diagram was done using directly the TiO₂ powder using a θ/2θ diffractometer (Siemens D5000). The software used for the qualitative analysis was Eva 9.0 (to detect the phases present) and Topas 2.1 for the quantitative analysis (of the identified phases) using the Rietveld method [ref 5], both from Bruker AXS GmbH. The data used is from the ICDD. The angle was set from 24° to 80°.

1.2. Results

The indexed peaks can be found on the powder diagrams obtained in Figure 1. No peak for the brookite structure was identified in the qualitative analysis and therefore was not set in the parameters for the Rietveld analysis. No impurity was found.

The mass percentage obtained with the quantitative analysis of the phase composition of the powders can be found in Table 1. The Rietveld analysis also permits a refinement of the lattice parameters [ref 5]; the results are shown in Table 2.

<table>
<thead>
<tr>
<th></th>
<th>Anatase</th>
<th>Rutile</th>
<th>Brookite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass %</td>
<td>81.71 %</td>
<td>18.29 %</td>
<td>0 %</td>
</tr>
</tbody>
</table>

Table 1: Results of the quantitative analysis.
Figure 1: Powder diagram of the TiO$_2$ powder sample.

![TiO$_2$ (P25) Powder Diagram](image)

<table>
<thead>
<tr>
<th>System</th>
<th>Space group</th>
<th>a parameter of ICDD (Å)</th>
<th>c parameter of ICDD (Å)</th>
<th>a parameter refined (Å)</th>
<th>c parameter refined (Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anatase</td>
<td>I$<em>{4}$$</em>{1}$/a md (141)</td>
<td>3.7852</td>
<td>9.5139</td>
<td>3.7820</td>
<td>9.4956</td>
</tr>
<tr>
<td>Rutile</td>
<td>P$<em>{4}$$</em>{2}$/m nm(136)</td>
<td>4.5933</td>
<td>2.9592</td>
<td>4.5886</td>
<td>2.9552</td>
</tr>
</tbody>
</table>

Table 2: Crystallographic data of the ICDD of the tetragonal polymorphs of TiO$_2$ and value of the refined parameters from the Rietveld analysis.

1.3. Conclusion

This X-ray diffraction analysis permitted to know the powder composition. The TiO$_2$ elaborated in MIT is very pure and consists in a majority of anatase particles.

2. TEM Pictures

The characterisation was based on TEM pictures since the material to study was a powder of nanosized particles. No other microscope technique can offer a sufficient resolution.

2.1. Experiment

The samples were prepared from the TiO$_2$ powder for the Transmission Electron Microscope (TEM). A small amount of powder was put in isopropanol and the suspension was left 10 minutes in an ultrasonic bath to avoid forming agglomerations of particles. Two drops of the suspension were placed on a 3mm diameter carbon-coated copper grid and the alcohol evaporated completely in air.
The samples were then analysed with a JEOL 2010 at 200kV. The photos were made on a film, then developed and digitalized. Several types of images were obtained:
- Bright Field micrographs (BF)
- High Resolution Electron Microscopy images (HREM)
- Selected Area Diffraction Pattern (SADP)

### 2.2. Results

The BF images obtained show the particles of TiO\textsubscript{2}. Figure 2 is an example of the BF type of images at a magnification of 600 000 that were used to do the particle size repartition analysis (see part II). Figure 3 is an example of the HREM pictures that were used for the analysis of the lattice characteristic. They were also used for the elaboration of the Lattice Parameter Analyser (see part III).

![Figure 2: Example of TEM photograph (mf9422) of the powder obtained.](image1)

![Figure 3: Example of TEM image (mf9415) that was used for the lattice analysis.](image2)

Other images, such as SADP and images showing Moiré fringes were obtained, but they were not analysed.

It is important to note that I did not succeed in obtaining these pictures alone. Therefore to able me to move forward with the project, it is my supervisor that made these pictures while he was working at the Department of Material Science and Engineering in MIT.
2.3. Conclusion

Good quality TEM pictures were obtained in different conditions. Enough were made for the size distribution analysis and to determine the atomic lattice characteristic of some particles.

3. Sample Preparation

My project was a preliminary work for the study of the grain boundary diffusion in consolidated nanostructured oxides. To anticipate the work of the following student, an important part of the time was spent for the sample preparation. It consisted in the preparation of compacted pellets of TiO$_2$ powder for TEM analysis since what was to be observed was grain boundaries.

The preparation of TEM samples is not straightforward because in the end, the sample must present, over an extended area, a thickness as small as 10 to 50 nm. The different steps were:

- Cold pressing of the nanopowder
- Sintering
- Mechanical grinding
- Ion beam thinning

The material used was a powder of titanium oxide, thus the samples obtained were ceramic. The samples manipulated were therefore extremely fragile and because it was made from a powder, they were also very flaky. That is why, in each step of the sample preparation, a lot of samples broke into pieces.

This sample preparation required a lot of care and meticulous experimentation and was not simple for non-familiarized person.

3.1. Cold Pressing

Introduction

There are different types of pressing possible to compact powders: cold pressing and hot pressing, that both can be unidirectional or isostatic pressing. In addition to that, there is the possibility of doing the compacting and sintering at the same time [ref 1]: hot isostatic pressing, sinter forging, transformation-assisted consolidation, plasma pressure compaction…

In particular, the transformation-assisted consolidation has been used for TiO$_2$ nanopowders [ref 6]. The powder is pressed for 1 hour at the high pressure of 1.5 GPa and in a temperature that ranged from 400°C to 600°C. It takes advantage of the anatase to rutile phase transformation that occurs with a reduction of volume. Because of this, the nucleation barrier is reduced, as well as being already enhanced by the application of external pressure. Because of the low temperature (only 0.2 to 0.3 times the melting temperature compared to the usual 0.5 to 0.8 times for sintering temperatures), the diffusion is minimized. This combination helps producing a sintered product with a grain size as small as the starting material. This would have been perfect for the study of grain boundary diffusion since the sintered material would have had a great density of grain boundary.

However, this technique could not be tested because the only available equipment was a simple unidirectional manual hydraulic press. Therefore compaction was obtained by a cold pressing.
What is usually done to prepare pellets for TEM analysis [ref 7] is to process pellets as thick as a centimetre, slice them with a low speed wafering saw, do the sintering, then operate a mechanical grinding until obtaining a thickness of less than 1 µm and finally using the ion milling to obtain the thin thickness required to use the TEM.

But because no wafering saw was available at DCMM, the aim was to produce pellets of more or less 0.1 mm of thickness directly after the cold pressing. Then, the only steps left would be grinding using an automatic dimple grinder and the ion milling.

**Experiment**

The steel mould used was composed of a cylindrical matrix and a cylindrical awl that fitted in the hole of 1.5 cm diameter. The pressure used was 5 t/cm², it was chosen to be as high as possible to obtain the best compaction without deforming the mould and the quantity used was 0.45 g of TiO₂ for each pellet.

**Results**

Several samples roughly 0.1 mm thick were prepared. Figure 4 shows a general view of a pellet obtained, it presents a cracking. Figure 5 was obtained with an optical microscope equipped with a digital camera. It shows, at a magnification of 50, the imprints of the matrix’s or awl’s lines of manufacturing.

![Figure 4: Typical pellet obtained after the cold pressing.](image1)

![Figure 5: Black and white image of a pellet’s surface at a magnification of 50.](image2)

### 3.2. Sintering

Sintering is the bonding of adjacent surfaces of particles in a mass of powder obtained by heating. Sintering strengthens a powder mass and normally produces densification.

**Experiment**

The sintering was done for 1 and 2 hours at 1200ºC in air in a conventional oven. Another oven was used to do a sintering for 1 hour at 1400ºC also in air.

**Result**

After this step, the pellets were noticeably more resistant. However, the originally white pellets turned yellowish in both cases and black at some places for the 1400ºC sintering.
3.3. Grinding

Grinding is the first step in the thinning of the sample for the MET analysis. The aim is to remove as much material as possible to facilitate the next step of ion beam thinning, which is very slow.

Experiment

A GATAN Model 656 Dimple Grinder was used. It is an instrument used for grinding circular dimples of spherical profile. It is supposed to be able to obtain a thickness as small as 5 µm. The principle is to mount the sample on a magnetic turntable and to apply a diamond-polishing compound (2-4 µm) that will take away material when a wheel in contact with the sample rotates.

It was necessary to cut the pellets in small pieces. Indeed, the samples for the TEM must be disks of 3mm in diameter and no special equipment was available to do the cutting. So the method was to try and break the samples in pieces of such size and form that it would fit on special copper rings. Then, because the samples were fragile and presented cracks from the handling before they were sintered, they often broke during the grinding process.

Result

This step was not easy and patience and care were required. However, samples from all the different sintering times and temperatures were successfully prepared. The ones with a correct shape were selected and glued to the copper rings.

3.4. Ion Beam Thinning

The final thinning step was done with the ion beam technique. The instrument was a GATAN Duomill model 600. Two canons at an angle of 15º (one thinning the top, the other, the bottom) sputtering plasma-induced argon ions were used. A voltage of 5 kV and a total argon ion current of 1 mA were set.

The sample preparation had to be stopped there. Indeed, we ran out of the special rings and new ones had to be ordered. Moreover, a problem occurred with the machine: the sample holder could not go down until the canons. Thus, the ion beam thinning could not be continued.

3.5. Conclusion

The sample preparation for the TEM observation of grain boundaries was well advanced and represents a considerable gain of time for the next student. What is left to do is to proceed with the last ion beam thinning step. All the difficult steps of compaction of the powder and mechanical grinding were successfully done.

4. Conclusion

This first part of the project included sample preparation, composition determination based on X-ray diffraction and the making of the TEM images that will be analysed in the following parts. Nanostructured TiO₂ samples were made from a nanopowder and then were prepared for TEM analysis. It was a meticulous task but it resulted in almost finished preparation as well as in a great experience in manipulation.
PART II: PARTICLE’S CHARACTERISATION FROM THE BRIGHT FIELD IMAGES

Because this project aims at preparing the study of transport properties in consolidated nanostructured oxides and particularly of the grain boundary diffusion, it is crucial to know the size and form of the nanoparticles. Those characteristics are indeed related to the size and shape of the grains that constitute the consolidated material fabricated from compaction and sintering of the nanopowder.

A direct image analysis of the Bright Field images from the TEM is presented and the results obtained are discussed. Then, in order to try to calculate automatically statistical values extracted from these images, an image processing algorithm is described.

1. Direct Method

In this part are presented the method used and results obtained concerning the analysis of the BF images. The aim was to extract information about the form and size of the TiO₂ particles of the nanopowder.

Seven Bright Field pictures at different magnifications featuring the nanoparticles were used to determine the characteristics. The software used was the Carl Zeiss Vision’s ActionVision4.

1.1. Method

The technique consisted in delimiting the particles with the mouse on the digitalized photos. Then ActionVision 4 calculated the area in pixel. With the scale of the pictures, the spatial resolution (value of the distance represented by one pixel) was known and the area values were automatically converted in square nanometres. It calculated the equivalent diameter and the perimeter of each particle. Therefore, after delimiting particles, obtaining the size distribution was possible.

The software calculated the diameter of the circle that has the same area as the particle (expression 1, where \( d \) is the equivalent diameter and \( A \) is the area). To obtain the perimeter, the Crofton method is used by ActionVision4. If the border pixels of the delimited particle share edges, it counts 1 pixel of distance, if the pixels share only corners, it counts \( \sqrt{2} \) pixel of distance.

\[
d = 2\sqrt{\frac{A}{\pi}} \quad (1)
\]

\[
f = \frac{4\pi A}{P^2} \quad (2)
\]

A form factor was computed for each particle (expression 2, where \( A \) is the area and \( P \) is the perimeter). This non dimensional factor gives an idea of the roundness of the particles. Indeed if the values of \( f \) are close to 1, it means that the particles tend to be spherical.

1.2. Results

The photos obtained from the TEM are of a very good quality (see contrast and focus on Figure 2) but despite the careful sample preparation, the particles formed agglomerations. This is probably due to the small size of the particles: the Van der Waals forces are not negligible and
the attraction between particles is enough to create the tendency to form agglomerations. This turned the particle identification difficult. On the images, not all the particles were measured, only the ones that were not too superposed were taken into account. This might have introduced a deviation from the real size distribution.

The accuracy of the values obtained for the equivalent diameter and from factor is considerable. Indeed the error made by the measurement is of more or less one pixel (for the mf9422 picture of Figure 2, 1 pixel represents 0.66 nm). In addition, the error is reduced when calculating an average.

Figure 6 shows the histogram obtained for the particles’ equivalent diameter from seven BF pictures. In total 339 particles have been identified and taken into account for the analysis. The repartition is approximately Gaussian and the average particle size is 25.4 nm with a standard deviation of 7.5 nm. It means that the repartition in size is quite large (15% of extension over the average value).

The histogram of the form factor can be found in Figure 7. The average value was found to be 0.77 with a standard deviation of 0.09. It means that the particles are not spherical but not very stretched either. However the histogram is far from being a Gaussian distribution. The distribution is large, meaning that there is a significant difference of form between particles.

1.3. Conclusion

This first method to obtain characteristics of the nanoparticles included a tedious step: the identification and contour delimitation “by hand” of 339 particles. It led to the estimation of the average particle size and form factor.
2. Size Repartition Automation

The rest of the project was realised in the Brazilian Centre for Research in Physics (CBPF) in the LPDSI laboratory.

To avoid the need of the particle contour determination, it was attempted to develop an automatic tool. The objective was thus to create a program that detected the particles in the BF image and calculated their characteristics (e.g. area, size, form factor...). It also aimed at leaving a tool that other scientists could use in similar case of nanopowder characterisation. Such program would turn the statistic analysis of the TEM pictures fast and automatic. A Matlab algorithm was elaborated with those aims.

2.1. Program Description

Digital image processing is divided into basic steps:
- Acquisition of the image: capturing the image and digitalisation if necessary.
- Pre processing: preparing the image before the segmentation.
- Segmentation: obtaining a black and white image separating the objects from the background.
- Attributes extraction: computing of the desired characteristics of the image.
- Interpretation.

In the following paragraphs, more details are given about the different steps. To illustrate the program an image showing rice grains will be used (Figure 8).

Pre Processing

Background illumination correction:

It can be observed on the original image (Figure 8) that the bottom of the picture is darker than the top. This may cause problem when the thresholding will be done to detect the objects. It is necessary to correct the background illumination.
First the image is divided into small squares (32 by 32 pixels) and the program computes the minimum value (the blackest pixel) in each square. Then it creates an image with the size of the original one with these values. Finally, this background representation image is subtracted to the original image. Figure 9 shows the corrected picture.

Contrast correction:

Because there has been a subtraction, the value of the intensity has lowered, thus the image is blacker. It is necessary to adjust the contrast. This is done with the `imadjust` Matlab function. It rescales the image intensity redistributing the pixel intensity values between the minimum and the maximum intensity value. The result can be observed in Figure 10.

![Figure 8: Original image.](image1)
![Figure 9: Image with background illumination corrected.](image2)
![Figure 10: Image with contrast adjusted.](image3)

Segmentation

This step is done by the Matlab function `graythresh`, it detects the correct threshold level by the Otsu method [ref 8]. Then the segmentation is done with the `im2bw` function. Figure 11 shows the segmented image.

The program has correctly detected the objects from the background. However, nothing guarantees that for every picture, the level chosen by the function will be correct. That is why the segmented picture will be shown at the end to the user so that he can check that the binary image displays the particles. A possible improvement of this step would be an option that lets the user try different levels of threshold until obtaining a satisfactory result.

An important step is to remove the objects that touch the border. Indeed, they are not complete objects and can reduce the accuracy of the measurements of object characteristics. This is done by the Matlab function `imclearborder`. The result can be observed in Figure 12.

Attribute Extraction

First, the program detects and labels each object in the picture. Then with the `imfeature` function, the following parameters are extracted and kept in an array:

- Area.
- Equivalent diameter: computes the diameter of a disk that has the same area.
- Centroid: keeps the centre of mass of the object.
- Image: keeps the image of each object individually.
Note that all these characteristics are measured in number of pixels (or square pixels). The scale has to be put into the program to have the results in nanometres.

Because there is no automatic way to obtain the perimeter of an object, it is necessary to use an indirect way. From each individual object (obtained with “image”) a black pixel is added to the borders of the box. Then the `bwperim` function transforms the image keeping only the edge of the object (if no extra pixels had been added, the function would be unable to detect the object in the box). The perimeter is obtained by the `imfeature` function computing the area (with the option of a connectivity of 8 because in this case the pixels that share corners are considered from the same object).

The mean particle’s equivalent diameter and its normal deviation can be calculated with Matlab. It also computes the $f$ form factor described in the direct method section (expression 2).

**Finishing**

The program displays the results of the computing: average particle size and area and their normal deviations, the time taken for the computing and the relevant images (original image and segmented image).

If the user wishes to do further analysis or obtain the histogram from another program, he can use a file containing the computed characteristics saved by the program. The file is automatically named after the original image name and contains for each object: the area, the equivalent diameter and the $f$ form factor.

**2.2. Results**

This program works well with the example and other pictures where the objects to be measured are separated. However, it is difficult to analyse the BF TEM pictures such as the mf9422 (Figure 2) for two reasons.

Although it is a good quality TEM image, the contrast shown is very complex for the particle detection. First, as can be observed in Figure 13, there is little difference between the background and the light particles’ intensity. Then, there is a big difference of intensity from different particles. Because some of them are diffracting the electrons from the TEM beam, the transmitted beam is weaker and they appear blacker. This difference also turns the detection more difficult.
Nevertheless, the most important problem lies in the fact that there is a lot of particle superposition. When the program does the segmentation step, it is impossible to separate the particles. It finds objects that are several agglomerated particles. Therefore the characteristic analysis obtained with this program is irrelevant. The result of the automatic segmentation of the described algorithm is shown in Figure 14. It can be observed that the detected objects do not correspond to the particles.

Other ideas were used to try and better the critical segmentation step. One was to do double threshold segmentation; another was to use the watershed method [ref 9]. But the results were not satisfying.

2.3. Conclusion

An algorithm was developed under Matlab, it permits to extract the desired characteristics of objects. However, my BF TEM pictures showing the nanoparticles proved to be too complex to be analysed by this program. That is why the program was left at an incomplete state.

Nevertheless, elaborating this first program has been my first experience with Matlab and image processing. This has been a useful and formative opportunity and left me more prepared for the following programs.

3. Conclusion

Two approaches for the size and form distribution of the particles were tempted. The first direct method using a specialised software led to the desired characteristics. The average particle size was found to be 25.4 nm. The form factor analysis showed that the particles are roughly spherical. However it is tedious to obtain accurate and statistical values. The second method, involving image processing, did not bring results due to the complexity of the BF images. However, it proved to be very useful and formative for the techniques that were going to be developed.
PART III: LATTICE CHARACTERISTICS FROM HIGH RESOLUTION ELECTRON MICROSCOPY ANALYSIS

Another important part of the project consisted in analysing the crystal structure of the TiO₂ particles. For this, the HREM images were studied. It was also realised at the CBPF.

From a direct image analysis method, the HREM images showed nanoparticles of size from 8 to 28 nm in diameter. It can also be observed that they are almost defect free single crystalline particles which is an expected result for such small particle size. Finally, a Matlab image processing algorithm is presented as a new technique for computing statistical values for the atomic lattice characteristics.

1. Direct Method

1.1. Introduction

The first technique used to extract information about the TiO₂ nanoparticle’s lattice properties was direct. It consisted in measuring plane separation distances and angles “by hand”, using measurement tools with Adobe Photoshop.8

The measurements were done on the mf9415 micrograph (Figure 15) because it was the picture with the best quality. Only on this one the atoms were identifiable in two directions. The other pictures mostly showed rows in one direction, thus offering less information.

1.2. Results

The values of the atomic raw separation (d) in the two directions and angle between them are in Table 3. They result from an average made with several measurements, nevertheless they cannot be considered as statistical. That is why it is not possible to say if the difference found in the parameters in the two directions is relevant. The accuracy is at minimum of the spatial resolution (0.03 nm).

<table>
<thead>
<tr>
<th>Direction 1</th>
<th>Direction 2</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>d = 0.3509 nm</td>
<td>d = 0.3549 nm</td>
<td>97.7°</td>
</tr>
</tbody>
</table>

Table 3: Result of the direct measurement method.

It seems that the distances measured on the image could correspond to the first parameter of the anatase system (a = 3.7820 Å from Table 2 values of the refined Rietveld parameter analysis). However, the values are 7% smaller. Moreover, the measured angle do not correspond to a simple plane (such as (0,0,1) for example).
Several dislocations were found.

It was expected that the particles were defect free because of their small size, however the central particle of Figure 15 presented dislocations. The arrows indicate the positions of identified dislocations. Figure 16 is a detail of the mf9415 picture where the atomic rows have been coloured in red to make the dislocation plane appear.
1.3. Conclusion

The particles were found to be almost defect free single crystalline particles. Nevertheless, obtaining a good estimation of the lattice’s characteristics with this direct method in not obvious. It is also a tedious work to obtain statistical values of the plane separation distances. There was not enough time to explore the methods of complex atomic plane determination, that is why the atomic plane showed was not identified. More details about the attempts of lattice plane identification will be given in Part III.

2. Automation of the Analysis of HREM Images

In this final part, a Matlab program called Lattice Parameter Analyser (LPA) is described. Its aim is to give distances between first neighbours from a high resolution electron micrograph. The LPA is also characterised using simulated lattices generated by another elaborated program: Lattice Generator (LG). Finally, the LPA is used for the analysis of the HREM images from our TiO$_2$ sample.

2.1. Lattice Generator

To be able to test the Lattice Parameter Analyser, a program was elaborated. The Lattice Generator (LG) creates images with perfect lattices, thus simulating the pictures that can be obtained with the TEM in HREM conditions. Indeed, it is necessary to check that the LPA programs gives correct results with images for which the lattice parameters are perfectly known.

2.1.1. Program Input

The LG program reads a configuration file (text file) containing information about the lattice the user wants to create and then generates it. The information contained are:

- The image size in pixel
- The lattice parameters a and b in pixels
- An option if an hexagonal lattice is desired
- An angle value to rotate the lattice
2.1.2. Program Output

The LG program creates the desired lattice and saves it as a tif format picture. The picture’s name is automatically given according to the parameters (a, b and angle values).

2.1.3. Program Description

First, the LG reads and converts the parameters contained in the configuration file. It then creates a black picture of the specified size. It puts a white dot at a regular interval given by the parameters specified by the user. It creates “atoms” from the dot applying a dilatation [ref 9], then the resulting image is blurred so that it looks like a real atomic resolution picture. The image is enlarged before being rotated of the specified angle value. Finally the rotated image is cropped to the desired image size. If the user set the hexagonal option, a special function is called to generate the lattice.

2.1.4. Results

This program functions correctly. Figure 17 and Figure 18 are examples of generated lattices.

![Figure 17: Example of created image: square lattice of 512x512 pixels with a parameter of 20 pixels and an angle of 10 degrees.](image1)

![Figure 18: Example of created image: hexagonal lattice of 512x512 pixels with a parameter of 50 pixels and no rotation.](image2)

2.1.5. Conclusion

A program was developed to serve as a generator of test images to check the LPA program. The user does not need to understand Matlab coding or how it functions to use it. What is simply required is to fill the desired parameters in the configuration file and to run the main program. The generated image will be automatically created and saved. The user can use the ReadMe file to have a brief description of LG, the information about the required software to run it and indications to know how to use it.

The LG complete code can be found in Appendix A. The LG program can be downloaded with the ReadMe file and a default configuration file at http://www.cbpf.br/cat/lpdsi/lpa.
2.2. Lattice Parameter Analyser

The LPA computes the distances between all atoms in a defined neighbourhood area through a Radial Distribution Function (RDF). The program outputs a histogram of such distances, it will feature peaks corresponding to the first neighbour’s distances. It is a powerful tool since the information extracted is statistical.

2.2.1. RDF Method

The RDF is a method to obtain an average notion of a structure [ref 14]. The RDF addresses the question: “given that I have one atom at some position, how many atoms can I expect to find at a distance r away from it?”. An example of an application of the method is the determination of patterns in the formation of magnetic bubbles in ferrofluids [ref 10].

2.2.2. Input

The inputs are a picture and, as with the LG program, a configuration file. It specifies relevant parameters to be set in order to obtain a correct analysis. The analysed image must be a good quality atomic resolution picture where the atoms are separated. The configuration file must specify the following parameters:

- The image name (with the extension).
- The image scale: it is the spatial resolution, in other words, the distance value of one pixel.
- The neighbourhood value: the distance in nanometres for which the distance analysis (RDF) will be done.
- FlagPrint: if set to 0, the program will run in silence, if set to 1, it will show the running status and intermediary results (graphs, data…)

It also contains additional parameters that can be modified to obtain an optimization of the analysis:

- The RDF Threshold: parameter used in the detection of the peaks in the histogram of the distances.
- The RDF Filter Order: order of the filter that smoothes the histogram to help doing a Gaussian fit of the curve.

However, the analysis works well in general with the default values.

2.2.3. Output

The LPA shows the results in the Matlab window and saves relevant information in different forms:

- The segmented image
- The list of all the distances computed by LPA
- The histogram graph
- The peak positions and standard deviations

The output files allows the user not only to check that the analysed image was correct and to have the distances’ list to do an independent analysis but also to have all the results saved in the form of a complete graph and a text file.
Figure 19: Flow chart of the Lattice Parameter Analyser.
2.2.4. Program Description

The program is divided into steps which correspond to a function calling of the main program. The flow chart of the LPA can be found in Figure 19 and the complete code of the program’s functions can be found in Appendix B.

Reading the Configuration file: ReadConfigFile.m

This function is the part of the program that reads the parameters in the configuration file and converts the text strings into correct units.

Processing the Image: ProcessImage.m

The image processing step (Table 4) is basically the same as the pre processing and segmentation steps in the size distribution analysis program (see section 2.1 of Part II).

<table>
<thead>
<tr>
<th>Adjusting background illumination</th>
<th>( I_{\text{bkgd}} = \text{GaussFilter}(I_{\text{init}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( I_2 = I_{\text{init}} - I_{\text{bkgd}} )</td>
</tr>
<tr>
<td>Adjusting the contrast</td>
<td>( I_3 = \text{RedistributeIntensity}(I_2) )</td>
</tr>
<tr>
<td>Segmentation</td>
<td>( \text{Level} = \text{GrayThresh}(I_3) )</td>
</tr>
<tr>
<td></td>
<td>( I_4 = \text{Thresholding}(I_3, \text{Level}) )</td>
</tr>
<tr>
<td>Removing dots/noise</td>
<td>( I_5 = \text{MorphologicalOperation}(I_4) )</td>
</tr>
<tr>
<td>Removing edge objects</td>
<td>( I_{\text{final}} = \text{ImgClearBorder}(I_5) )</td>
</tr>
</tbody>
</table>

Table 4: Steps of the ProcessImage sub program.

Comments:

Adjusting illumination:

The method used here is different from the first program because it had a drawback: it was limited to images with a size in length and width divisible by 32. To solve this problem the image is filtered by Gaussian type of filter to obtain a background approximation [ref 12]. Then this calculated background is subtracted from the original image.

Morphological operation:

A basic morphologic operation is done on the segmented image. An opening [ref 9] is done to remove small dots that may appear after the segmentation that do not correspond to atoms on the original picture. An amelioration of the program could be to transform this morphological operation as a parameter in the configuration file that the user could modify to optimize the analysis.

Computing the Distances: DistMeasurements.m

This part of the program (Table 5) first detects and labels each object of the processed image. It then extracts with the \text{imfeature} function the centre of mass of each atom. Finally, it computes the distances between atoms (RDF method). It uses a double loop algorithm: for each atom \( i \) in the picture, it computes for each atom \( j \) within the neighbourhood distance, the distance between the centres of mass of atoms \( i \) and \( j \). All the distances calculated are kept in an array.
Labelling object  | Objects = Label(I_{final})
Extracting centre of mass | Centroid = ImFeature(Objects)
Distance computation  
| For i=1 to nb of all objects
| For j=1 to nb of objects in neighbourhood
| Compute distance between object i and j
| End
| End

Table 5: Steps of the DistMeasurements sub program.

Comments:

In a first version of the LPA, there was not the neighbourhood concept for the distance computation. It calculated all the distances between all the atoms in the image. This computation took a lot of time (about 30 minutes for 1000 atoms) and it was necessary to use the Linux Cluster of the CBPF to run the program. The final version including the neighbourhood distance is much faster (about a minute for 1000 atoms) and it is relevant for our study because the objective was to extract the distances between the first neighbours in the atomic lattice.

Finding the Peaks: FindMaxGauss.m

First, this sub program (Table 6) creates a histogram from the array of all the distances, then it smooths the signal with a convolution technique and normalizes it. The critical step is finding the peaks in the histogram. A special Matlab function `imextendmax` detects the peaks and their range in the histogram. A Gaussian fitting is accomplished for each detected peak. The position of the maximum of the curve and its standard deviation (width of the Gaussian curve) are calculated for each peak. The result is shown in a scaled graph (the distances are converted to nanometres with the scale given in the configuration file).

| Histogram | Histogram = Hist(Distance, 500) |
| Smooth signal | Signal = Convolution(Histogram, Filter) |
| Find peaks of the signal | Peaks = Imextendmax(Signal) |
| Gaussian fitting | GaussPeaks = Gaussfit(Peaks) |
| Extract information | NeighDistance = Position(Max(GaussPeaks)) |
| | Deviation = std(GaussPeaks) |
| Showing graph and results | Plot(Histogram, GaussPeaks, NeighDists) |

Table 6: Steps of the sub program FindMaxGauss.

Comments:

Histogram creation:

The number of bins is fixed to 500 and Matlab estimates automatically the bin width. An optimization for the number of bins was tempting. An evaluation of the number of bins as a function of the neighbourhood distance and the scale was searched. However, there is no formula that works for all cases [ref 11] that is why the value of 500 was left as a default setting.

Convolution for signal smoothing:

The convolution is done between an array containing the histogram’s counts and an array containing ones [ref 12]. The length of the second array determines the force of the smoothing. This length is a parameter that can be changed in the configuration file (RDF Filter Order), it is set to 10 as default. For example, it can be increased when the histogram is very rough; the effect will be to facilitate the peak detection and the Gaussian fitting.
Peak detection:

There is a parameter of the configuration file that the program takes into account when calling the `imextendedmax` function. The RDF Threshold is a sort of thresholding: it is the difference in height that will determine if two close peaks are separated or not. Its value is set to 0.05. If the user is not satisfied with the peak detection, he can try to raise the RDFTh value if he wants to obtain a better peak separation.

Gaussian fitting:

A trick for a correct Gaussian fitting is to enlarge the range given by the `imextendedmax` function. A good value of 2 bins on each side was found by testing various values and images.

**Saving the Outputs: SavingData.m**

This function creates a directory after the original image’s name. It will contain the 4 output files, also named automatically.

Peak position: The position and standard deviation of the peaks are saved in text file.

Neighbour’s distances: All the distances computed in the DistMeasurement function are saved in a data file Therefore, the user has all the information needed if he wishes to plot the histogram with another program or run other analysis.

Segmented image: The processed image showing the detected objects is saved in a jpeg format. The user can then check if the image that was analysed for the distance computation was correct. Indeed, there may be cases of HREM image where the atoms are not well separated and the computed distances will not correspond to the first neighbour’s distance since the objects did not correspond to the atomic lattice. In addition to that, on one of the atoms, this image will show coloured circles that correspond to the first neighbour’s distances. It is useful to visualise those distance on the atomic lattice.

Peak graph: A complete nanometre scaled and normalized graph is saved in a jpeg format. It contains:

- The bar histogram
- The Gaussian curve fitting each peak
- The peak position in nanometres on top of each one

**2.2.5. Conclusion**

To accomplish this complete LPA program, a lot of optimization was done. Several methods were tried before obtaining the final version. It also contains various parameters that can be modified if the user wishes to reach the best results.

The distances evaluated by this program are obtained automatically and are useful for the determination of the atomic plane of HREM images. It is no longer necessary to measure “by hand” the lattice characteristics. Moreover, it is a powerful tool because its results are statistical. The user just needs to interpret the result to finally know which atomic plane is shown on the original HREM picture.

The LPA program can be downloaded with the ReadMe file for instructions and a default configuration file at [http://www.cbpf.br/cat/lpdsi/lpa](http://www.cbpf.br/cat/lpdsi/lpa).
2.3. LPA Results with Simulated Lattices

An important step is to check that the elaborated program gives correct results before using it to analyse real HREM photos. The Lattice Generator program was used because, as the lattice characteristics are set, a simple comparison of the values of the first neighbours’ distances can tell the accuracy of the LPA program.

2.3.1. Square Lattices

First, the simplest lattice was tested. Square lattices with different lattice parameters and angles of rotation were made with the LG program. The scale was set to 1 pixel = 0.03 nm.

Figure 20 shows a representation of a square lattice and the values of the first neighbours’ distances in function of the lattice parameter a.

![Representation of a square lattice and the values of the first neighbours’ distance in function of the lattice parameter a.](image)

Figure 20: Representation of a square lattice and the values of the first neighbours’ distance in function of the lattice parameter a.

![Example of peak graph that is created and saved by the LPA program.](image)

Figure 21: Example of peak graph that is created and saved by the LPA program.
Figure 21 shows an example of the output graph of the LPA program for a square lattice of 20 pixels created by the LG (similar to Figure 17). The bar histogram values are in green and in red are the Gaussian curves that fit the histogram’s peaks. The positions written correspond to the first neighbours’ distances computed by LPA. The standard deviations are not showed on the graph because it would over load it. They are given in the Matlab window and in the text file saved automatically.

The comparison of the theoretical values and the results given by the LPA are featured in Table 7. What can be observed is that the program always gives values very close to the theoretical values. In addition to that, the theoretical values are always included in the error range according to the standard deviations. This is proved by the value of \( \chi^2 \) [ref 11]. Its formula is given in expression (3) where \( N \) is the number of peaks taken into account, \( X^\text{th}_{ij} \) is the theoretical value of the peak position, \( X^\text{mes}_{ij} \) is the measured peak position and \( \sigma \) is the standard deviation of the peak. When the value of \( \chi^2 \) is inferior to 1, the measured value is inside the error bar given by the standard deviation.

\[
\chi^2 = \frac{1}{N} \sum_{i} \left( \frac{X^\text{mes}_{ij} - X^\text{th}_{ij}}{\sigma^2_{ij}} \right)^2
\]  

<table>
<thead>
<tr>
<th>Neighbour</th>
<th>Theory</th>
<th>Angle = 0</th>
<th>Angle= 20</th>
<th>Theory</th>
<th>Angle = 0</th>
<th>Angle= 20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.6000</td>
<td>0.5994 +/-0.0060</td>
<td>0.5997 +/-0.0073</td>
<td>0.9000</td>
<td>0.9077 +/-0.0078</td>
<td>0.9014 +/-0.0118</td>
</tr>
<tr>
<td>2</td>
<td>0.8485</td>
<td>0.8482 +/-0.0057</td>
<td>0.8485 +/-0.0078</td>
<td>1.2728</td>
<td>1.2712 +/-0.0137</td>
<td>1.2703 +/-0.0158</td>
</tr>
<tr>
<td>3</td>
<td>1.2000</td>
<td>1.1996 +/-0.0056</td>
<td>1.2004 +/-0.0069</td>
<td>1.800</td>
<td>1.7966 +/-0.0137</td>
<td>1.7976 +/-0.0152</td>
</tr>
<tr>
<td>4</td>
<td>1.3416</td>
<td>1.3418 +/-0.0071</td>
<td>1.3411 +/-0.0090</td>
<td>2.0125</td>
<td>2.0121 +/-0.0137</td>
<td>2.0097 +/-0.0154</td>
</tr>
<tr>
<td>( \chi^2 )</td>
<td>0.0047</td>
<td>0.0020</td>
<td>0.2627</td>
<td>0.0243</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Neighbour</th>
<th>Theory</th>
<th>Angle = 0</th>
<th>Angle= 20</th>
<th>Theory</th>
<th>Angle = 0</th>
<th>Angle= 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.2</td>
<td>1.2094 +/-0.0095</td>
<td>1.2023 +/-0.0139</td>
<td>0.4500</td>
<td>0.4510 +/-0.0037</td>
<td>0.4516 +/-0.0048</td>
</tr>
<tr>
<td>2</td>
<td>1.6971</td>
<td>1.6935 +/-0.0168</td>
<td>1.6941 +/-0.0178</td>
<td>0.6364</td>
<td>0.6360 +/-0.0038</td>
<td>0.6351 +/-0.0068</td>
</tr>
<tr>
<td>3</td>
<td>2.4000</td>
<td>2.3977 +/-0.0168</td>
<td>2.3976 +/-0.0168</td>
<td>0.9000</td>
<td>0.9002 +/-0.0037</td>
<td>0.8994 +/-0.0099</td>
</tr>
<tr>
<td>4</td>
<td>2.6833</td>
<td>2.6780 +/-0.0189</td>
<td>2.6802 +/-0.0190</td>
<td>1.0062</td>
<td>1.0051 +/-0.0043</td>
<td>1.0063 +/-0.0047</td>
</tr>
<tr>
<td>( \chi^2 )</td>
<td>0.2806</td>
<td>0.0257</td>
<td>0.0381</td>
<td>0.0379</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Table comparing the theoretical values of the first neighbours’ distance and the values given by the Lattice Parameter Analyser.
The intensity of the peak cannot be taken into account for an extra analysis. Indeed, the convolution done to smooth the curve to facilitate the Gaussian fitting, multiplies the counts value.

The LPA program managed to find the correct neighbours’ distances of all the square lattices tested and gives results with a considerable accuracy.

### 2.3.2. Rectangular Lattices

Some rectangular lattices were tested too. Figure 22 shows an example of the resulting graph in the case of a rectangular lattice (a=15 pixels and b=30 pixels as can be seen in the title). What can be observed is that the Gaussian fitting is correct for the peak position but over estimates the peak intensity. Indeed, because the analysed lattice is almost perfect (created by the LG program), the histogram’s peaks are very sharp, turning the Gaussian fitting difficult. However, this does not induce any error on the result since what is important are the peak positions that correspond to the first neighbour’s distances of the lattice. The scale was set to 1 pixel = 0.03 nm.

Figure 23 shows a part of the image that the LPA program saves. It shows the segmented image resulting from the image processing of the original picture (rectangular lattice with a=15 pixels, b=30 pixels and an angle of 0). It can be observed that the detected objects correspond to the simulated atoms. The circles represent the distance computed by the program, they are alternatively green and red. It is a very practical way for the user to check the veracity of the program’s results.

![Figure 23: An example of the peak graph in the case of a rectangular lattice.](image)

Figure 24 shows a representation of a rectangular lattice and the values of the first neighbours’ distances in function of the lattice parameters a and b.
The comparison of the theoretical values and the results given by the LPA in this case are featured in Table 8. The first 5 neighbour’s distances were analysed. The comments are the same as in the square lattices’ case. The program always gives values very close to the theoretical values and the exact values are always within the error range (value of $\chi^2$ inferior to 1).

<table>
<thead>
<tr>
<th>Neighbour</th>
<th>Distance in nm</th>
<th>Neighbour</th>
<th>Distance in nm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$a = 20$ pixels = 0.6 nm</td>
<td></td>
<td>$a = 15$ pixels = 0.45 nm</td>
</tr>
<tr>
<td></td>
<td>$b = 30$ pixels = 0.9 nm</td>
<td></td>
<td>$b = 30$ pixels = 0.90 nm</td>
</tr>
<tr>
<td>Neighbour</td>
<td>Theory Angle = 10</td>
<td>Neighbour</td>
<td>Theory Angle = 0</td>
</tr>
<tr>
<td>1</td>
<td>0.6 0.5986 +/- 0.0075</td>
<td>1</td>
<td>0.45 0.4515 +/- 0.0048</td>
</tr>
<tr>
<td>2</td>
<td>0.9 0.8992 +/- 0.0070</td>
<td>2</td>
<td>0.9 0.9003 +/- 0.0050</td>
</tr>
<tr>
<td>3</td>
<td>1.0820 1.0805 +/- 0.0091</td>
<td>3</td>
<td>1.0062 1.0049 +/- 0.0050</td>
</tr>
<tr>
<td>4</td>
<td>1.2 1.1994 +/- 0.0066</td>
<td>4</td>
<td>1.2728 1.2733 +/- 0.0050</td>
</tr>
<tr>
<td>5</td>
<td>1.5 1.4989 +/- 0.0077</td>
<td>5</td>
<td>1.35 1.3497 +/- 0.0050</td>
</tr>
<tr>
<td>$\chi^2$</td>
<td>0.0208</td>
<td>$\chi^2$</td>
<td>0.0365</td>
</tr>
</tbody>
</table>

Table 8: Table comparing the theoretical values of the first neighbours’ distance and the values given by the Lattice Parameter Analyser.
2.3.3. Hexagonal Lattice

Hexagonal lattices created with LG were also tested. A representation of a hexagonal lattice and the values of the distances in function of the lattice parameter \( a \) are given in Figure 25.

![Hexagonal lattice diagram](image)

\[ b = a\sqrt{3} \]
\[ c = 2a \]

Figure 25: Representation of a hexagonal lattice and the first neighbours’ distance in function of the lattice parameter \( a \).

The comparison for the first 3 distances given by the LPA and the theoretical values are given in Table 9. The comments are identical: the LPA successfully finds the correct value with a good accuracy. However on the case of the simulated lattice of 50 pixels parameter, the \( \chi^2 \) is found slightly superior to 1. This is probably due to the fact that the histogram has extremely thin peaks, turning the Gaussian fitting is more delicate.

<table>
<thead>
<tr>
<th>Neighbour</th>
<th>Theory</th>
<th>Angle = 10</th>
<th>Theory</th>
<th>Angle = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9</td>
<td>0.8996 +/- 0.0051</td>
<td>1.5</td>
<td>1.4955 +/- 0.0093</td>
</tr>
<tr>
<td>2</td>
<td>1.5588</td>
<td>1.5595 +/- 0.0046</td>
<td>2.5981</td>
<td>2.5869 +/- 0.0104</td>
</tr>
<tr>
<td>3</td>
<td>1.8</td>
<td>1.8000 +/- 0.0051</td>
<td>3.0</td>
<td>2.9880 +/- 0.0092</td>
</tr>
<tr>
<td>( \chi^2 )</td>
<td>0.0098</td>
<td></td>
<td>1.0317</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: Comparison for some hexagonal lattices.

2.3.4. Verification with Real Lattices

An old HREM picture of TiO\(_2\) was analysed with the LPA program. The analysed picture can be seen in Figure 26. However the scale for this picture was not found, it was set to 1 pixel = 0.03 nm.

![Real lattice image](image)

Figure 27 shows a detail of the saved image showing the distance computed by LPA in the form of coloured circles and Figure 28 shows the resulting graph. The first observation is that LPA correctly detected the distances. The peaks are wider than the ones found in the simulated images. This is due to the fact that a real, therefore not perfect, atomic lattice was analysed. But the Gaussian fitting is close to the histogram values.

Because the scale was not found, it was impossible to check the values of the distance given by the program. However, the ratios of the values respect a square lattice:
- d2/d1 = 0.8297/0.5871 = 1.4132 ≈ √2
- d3/d1 = 2.0015 = 2
- d4/d1 = 2.2410 = √5

This result also proves that LPA correctly found the first neighbour’s distances.

Figure 26: Atomic resolution HREM of a rutile TiO₂ (0,0,1) plane.

Figure 27: Detail of the saved image showing the neighbour’s distances found by LPA.

Figure 28: Graph resulting from the LPA computation for the rutile picture.


2.3.5. Conclusion

With all the generated lattices, the Lattice Parameter Analyser managed to detect the first neighbours’ distances in the histogram. The values given are correct and the exact values are almost always included in the error range given by the standard deviation of the Gaussian curves that best fit the peaks (in only one extreme case it was not included, nevertheless the value was very close). The deviations found are very small due to the fact that the analysed lattices are almost perfect. Indeed, the peaks are very sharp because the atoms are positioned perfectly. It was also tested with a real HREM picture and it proved to give correct distances.

Now that the program has been checked and that it was concluded that it functioned correctly, it can be used to find atomic lattice characteristics of our TiO$_2$ HREM images.

2.4. LPA Results with the TiO$_2$ HREM Images

From the set of HREM images obtained with the TiO$_2$ powder sample, only one showed a sufficient quality to be analysed by the Lattice Parameter Analyser. It is the central particle of the mf9415 image (Figure 15) that was analysed (Figure 29).

![Image](image1.png)

**Figure 29:** Image that was analysed by the LPA. It is a part of the mf9415 picture (Figure 15).

![Image](image2.png)

**Figure 30:** Detail of the segmented image. The circles corresponding to the distances found by the LPA are red and green.

2.4.1. Comments

The scale was measured on the complete image, one pixel measures 0.03086 nm. 1247 atoms were taken into account in the distance analysis (RDF). The analysis was done for a distance of 1.5 nm (value of the Neighbourhood distance), RDF Threshold is set to 0.05 and RDF Filter Order is set to 10. The segmented image is relevant since the selected objects correspond to the atoms of the original image, as can be observed in Figure 30.
2.4.2. Results

Figure 31 shows the graph produced by the program with the first neighbours’ distance found and Table 10 gives the distances found and their standard deviation.

<table>
<thead>
<tr>
<th>Peak number</th>
<th>Distance (in nm)</th>
<th>Standard Deviation (in nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3536</td>
<td>0.0198</td>
</tr>
<tr>
<td>2</td>
<td>0.4671</td>
<td>0.0244</td>
</tr>
<tr>
<td>3</td>
<td>0.5395</td>
<td>0.0289</td>
</tr>
<tr>
<td>4</td>
<td>0.7313</td>
<td>0.0433</td>
</tr>
<tr>
<td>5</td>
<td>0.8369</td>
<td>0.0345</td>
</tr>
<tr>
<td>6</td>
<td>0.9314</td>
<td>0.0244</td>
</tr>
<tr>
<td>7</td>
<td>1.0712</td>
<td>0.0228</td>
</tr>
<tr>
<td>8</td>
<td>1.1873</td>
<td>0.0292</td>
</tr>
<tr>
<td>9</td>
<td>1.4118</td>
<td>0.0265</td>
</tr>
<tr>
<td>10</td>
<td>1.4831</td>
<td>0.0157</td>
</tr>
</tbody>
</table>

Table 10: Peak positions and standard deviation.

As expected, the peaks are wider than with the simulated lattices. However it can be observed that the Gaussian curves from the fitting respect roughly the histogram’s shape. It is important that the fitting respects closely the top of the peaks because it means that the distances given as the position of the maximum of the Gaussian curves correspond to the first neighbours’ distances.

The value of the first neighbour’s distance was found to be 0.3536 nm. It is close to the value found with the direct measuring method. But the LPA proved that the difference found with the first method does not exist. Indeed, there is only one peak at 0.3536 nm.

At first, the graph seems to show too many peaks, not that many were expected. However, the veracity of the result can be checked with the circles on the segmented image.
(Figure 30). It can be observed that each circle truly corresponds to a relevant distance between neighbours. For example the first green circle passes through the 4 first neighbouring atoms.

Something aroused from this analysis that was not expected: the program detected that the second neighbour’s distance in one direction is different in the other direction. Indeed, the second distance is 0.4671 nm and the third is 0.5395 nm (Figure 31), there are 2 peaks on the graph. Figure 30 shows the difference in the form of the first red circle and the second green circle. In addition, the 2 distances are represented in the bottom left corner: it can be seen that d2 is slightly smaller than d3. The argument that this can be observed on the image because it is a coincidence is false. The results given by the program are statistical because it made the RDF calculation for all 1247 atoms in the picture. The program gave information about this lattice that was not observed initially.

Also, it can be noticed that the 4th peak corresponds to the double of the 1st. The 6th corresponds to the double of the 2nd and the 7th to the double of the 3rd.

2.4.3. Discussion

To determine the atomic plane of the image, a demo version of the Crystal Studio program has been used (http://www.crystalsoftcorp.com/index.html). It helped visualize the projections of the complex TiO2 crystallographic structures.

The geometry is the same as the (2,1,1) plane of anatase phase but the distances are not respected. In this case, the first neighbour’s distance is 0.8464 nm [ref 2].

The atomic lattice also has the same geometry as the (1,1,1) plane of the brookite structure [ref 13]. This structure was analysed with LPA with a HREM image taken from this reference (Figure 32), the results are shown in Table 11 and the graph can be found in Figure 33. For this analysis, 635 atoms were taken into account, the RDFTh was set to 0.075 (because the histogram is noisier, it was necessary to lower this threshold) and RDF Filter Order was set to 15.

![Figure 32: Part of the HREM brookite image from ref 13.](image)

<table>
<thead>
<tr>
<th>Peak number</th>
<th>Distance (in nm)</th>
<th>Standard Deviation (in nm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3138</td>
<td>0.0164</td>
</tr>
<tr>
<td>2</td>
<td>0.3981</td>
<td>0.0275</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>0.6303</td>
<td>0.0308</td>
</tr>
<tr>
<td>5</td>
<td>0.7515</td>
<td>0.0659</td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>0.9357</td>
<td>0.0237</td>
</tr>
<tr>
<td>8</td>
<td>1.0246</td>
<td>0.0329</td>
</tr>
<tr>
<td>9</td>
<td>1.2413</td>
<td>0.0341</td>
</tr>
</tbody>
</table>

Table 11: Results for the LPA analysis of the HREM from ref 13 with the same peak indexing as Figure 31.

The histogram shows great similarities with the one from our sample. Some of the peaks are not detected by the program but the missing ones can be guessed (e.g. the 3rd peak would be at approximately 0.4 nm). This is probably due to the fact that fewer atoms are detected because the image is smaller and of a lower quality. The values are slightly lower (e.g. 0.3138 nm for the first peak instead of 0.3536 nm in our case), however, the ratios between the distances are
respected. This histogram seems to be a noisier version of ours, with the values shifted 10% to the left.

Nevertheless, it does not mean that the analysed image of our sample shows a brookite particle. Indeed, similar atomic plane geometries can be found in different structures (e.g. (1,1,1) plane of a body centred cubic structure has a hexagonal symmetry). Moreover, the X-ray analysis did not detect the brookite phase.

Therefore, more investigation has to be done to analyse the complex anatase and rutile structures. Also, a diffraction pattern was obtained for the mf9415 image but it shows several diffraction patterns because several particles are diffracting and it has not been studied yet.

![Figure 33: Resulting graph from the Figure 32 image.](image)

2.4.4. Other Idea

Another useful information would be the value of the angle between the direction 1 and 2. Indeed, angle between lattice directions is an important characteristic and would help the atomic plane identification in HREM images.

We began to elaborate a method also using a Matlab algorithm in this objective. Nevertheless, there was no time to accomplish a complete program.

It consists in rotating the segmented image with a step of 1 degree. At each rotation, the projections on the X and Y axis are computed. From each projection, the standard deviation is calculated. In the end the standard deviations are plotted for the X and Y axis in function of the angle. Those plots present peaks that correspond to the direction of the atomic lattice where the alignment of the atoms is maximum. With the technique of peak detection presented in the LPA program, the angles could be determined.

This technique would bring extra information about the lattice. The work could be continued to integrate this tool in the LPA program.
2.4.5. Conclusion

Despite the precious information given by the Lattice Parameter Analyser, the atomic plane on our HREM image was not identified. However, the LPA proved to give precious information in a totally automatic way. It only takes a few seconds to obtain precise lattice distances between first neighbours from a good quality HREM image. This program can be a powerful tool for scientists that analyse lattice structures using high resolution TEM micrographs.

3. Conclusion

Obtaining accurate values for lattice parameters from direct HREM image analysis is not straightforward. However, this step is necessary to make basic observations and collect information about crystal defects such as dislocations.

A program called Lattice Parameter Analyser (LPA) was developed to have a different approach to the measurements of lattice parameters of the HREM pictures of the analysed TiO₂ particles. Its results were analysed with simulated lattice images. It also permitted to discover characteristics of the HREM image that were not detected in the direct analysis method. In the end, what was developed is a complete program that measures lattice characteristics from HREM images that can be used by other persons. No knowledge of Matlab coding is required to use LPA, however, the final interpretation of the results has to be done carefully by the scientist.
CONCLUSION OF THE INTERNSHIP

Summary of the work accomplished and main results:

First, the physical experiments and characterisation techniques have been presented. The analysed TiO\(_2\) powder is composed of a majority of 82% of anatase particles and the remaining ones are of rutile structure. Sufficient TEM images were obtained for the particle’s size and shape analysis. Nanostructured TiO\(_2\) pellets were obtained from a cold pressing and sintering of the nanopowder. A sample preparation for TEM observation of the grain boundaries of the pellets has been almost completed.

Then, the particle’s characterisation from the Bright Field images has been described. The average particle size is 25.4 nm and their shape is roughly spherical. However obtaining accurate and statistical values is not obvious.

Finally, lattice characteristics from high resolution electron microscopy have been obtained from a direct image analysis as well as from image processing techniques. The nanoparticles of the powder are single crystalline and almost defect free. A Matlab program has been developed to extract accurate and statistical values of the distance between first neighbours. The Lattice Parameter Analyser has been proved to be an efficient tool.

Work to continue:

A continuation of this work would be to complete the sample preparation and to start the grain boundary study using the TEM. Further analysis of the complex crystallographic structures of TiO\(_2\) has to be done to complete the identification of the atomic planes of the HREM images of the particles. Also, what could be explored to complete the LPA program is the proposed rotation method.

Produced scientific material:

From the project’s results and the new characterisation techniques based on image processing algorithm, a poster has been prepared for the 3\(^{rd}\) Brazilian Material Research Society conference in Foz do Iguaçu on the 10\(^{th}\) of October 2004 (symposium D: Nano-Scale Structural Characterizations of Materials, http://www.sbpmat.org.br/3meeting/). Since at this date I will be following the Nanomaterials Master’s courses at Imperial College London, Professor Solórzano will present my work and results.

Also, a Technical Note has been elaborated for the PDSI Laboratory of the CBPF. This could be the base for the prolongation of my work of developing atomic lattice characterisation from HREM images of crystalline material. It contains the description of the Lattice Generator and Lattice Parameter Analyser, as well as the characterisation of the LPA. The algorithm code is also available.

In addition to that, an internet page (http://www.cbpf.br/cat/lpdsi/lpa) has been done where basic information about the LPA can be found. The Matlab program can be downloaded together with all the information needed to use it and with an example of a simulated lattice image that can be analysed by the LPA. This report is available there in a pdf format. Also, the poster for the 3\(^{rd}\) Brazilian Material Research Society conference can be viewed from this page.
Experience acquired:

This internship has been a unique opportunity to acquire experience in a prestigious South American university and in a renowned Brazilian research centre for physics. My involvement in the project has resulted in a considerable background for scientific research:

- Experimental techniques for the elaboration of nanostructured materials
- Handling of up to date equipments for the TEM sample preparation
- Characterisation methods
- Image processing techniques and tools
- Programming under Matlab
- Use of relevant softwares for poster and internet page elaboration

What is more, I have consolidated my knowledge and experience about the scientific processes of reaching results, especially concerning careful characteristics’ collecting and the importance of always being as critical as possible with the results.

This particularly formative project has been relevant for my professional goals and for the continuation of my studies as a Research Master student (MRes in Nanomaterials). Those internship’s accomplishments will be certainly extremely valuable as going on with my studies aiming at performing fruitful research in applied physics.
REFERENCES


BIBLIOGRAPHY


APPENDIX A: LATTICE GENERATOR

ReadMe

README FOR THE LATTICE GENERATOR (LG)
---------------------------------------------

This file contains basic information on how to run and test the Lattice Generator program.

PRINCIPLE OF LG
---------------------------------------------

The LG Matlab program generates simulated images of atomic lattices such as can be obtained with the Transmission Electron Microscope.

INSTRUCTIONS
---------------------------------------------

This Matlab program runs with the Matlab Version 6.5 Release 13. The user’s Matlab version must feature the Image Processing Toolbox as well as the Signal Processing Toolbox.

Simply run with Matlab the "Main_LatticeGenerator.m" program.

It will read the input file "LGConf.txt" which contains set up parameters and will generate an image named after the parameters.

INPUT FILE
---------------------------------------------

- FlagPrint: if set to 1, the program status and the final image will appear on the Matlab window.
- ImgSize: size in pixel of the created image
- a: first lattice parameter in number of pixels
- b: second lattice parameter in number of pixels
- Hexagonal: an hexagonal lattice can be created with the a as parameter.
  - if you wish square or rectangular write 0
  - if you wish hexagonal write 1
- Angle: the final image can be rotated if wished

The file "LGConf.txt" is set as followed:

FlagPrint : 1
Image size  : 512
Lattice Parameter a : a = 20 pixels
Lattice Parameter b : b = 20 pixels
Hexagonal  : 0
Angle      : 10 degrees

These parameters can be modified if desired.

OUTPUT
---------------------------------------------

The LG program will generate an image in tif format of the following
name: Lattice(a=20 b=20 angle=10).tif.

If the parameters of the "LGConf.txt" configuration file are modified, the name will change according to the values chosen.

If the hexagonal option is chosen (Hexagonal=1), the name will be: LatticeHex(a=20 angle=10).tif.

MORE INFO ABOUT LG

More detailed explanations about the LPA program can be found in the webpage:
http://www.cbpf.br/cat/lpdsi/lpa

AUTHORS

Marcelo Portes de Albuquerque
Contact: marcelo@cbpf.br

Carole Jonville
Contact: carole@cbpf.br
    jonville@dcmm.puc-rio.br
    carole.jonville@enspg.inpg.fr
# Program     : Lattice Generator
# Written by  : Carole Jonville & Marcelo Portes
# email for   : carole@cbpf.br
#               jonville@dcmm.puc-rio.br
#               carole.jonville@enspg.inpg.fr
#               marcelo@cbpf.br
# Copyright(©): (c) CBPF - CAT
# Version     : 4.0814
# Objective   : Creates images simulating HREM images of atomic lattice
# Project     : PFE ENSPG - PUC Rio - CBPF

---

# FlagPrint=1 shows the results on the screen
FlagPrint=1

# ImgSize: size in pixel of the created image
ImgSize=512

# a: first lattice parameter in number of pixels
a=20

# b: second lattice parameter in number of pixels
b=20

# Hexagonal: an hexagonal lattice can be created with
# the a as parameter.
# if you wish square or rectangular write 0
# if you wish hexagonal write 1
Hexagonal=0

# Angle: the final image can be rotated if wished
Angle=10
Main_LatticeGenerator

clear all; close all; clc;
warning off;
% -------------------------------
% Cabecalho
fprintf('
Lattice Generator - LaGen Version 4.0814');
fprintf('
Centro Brasileiro de Pesquisas Fisicas - CBPF Brazil');
fprintf('
Ecole Nationale Superieure de Physique de Grenoble - ENSPG France');
fprintf('
Pontificia Universidade Catolica do Rio de Janeiro - PUC Rio Brazil\n');
fprintf('
(C) MCT - All Rights Reserved - 2004');
%---------------------------------
%Read the configuration file

[FlagPrint ImgSize a b Hexagonal Angle] = LGReadConfigFile('LGConf.txt');

FlagPrint=str2num(FlagPrint);
if (FlagPrint==1)
    fprintf('
-------------------------------------------------');
    fprintf('
Input parameters...
	Image size            : %s', ImgSize);
    fprintf('
	FlagPrint             : %d', FlagPrint);
    fprintf('
	Lattice Parameter a   : a = %s pixels', a);
    fprintf('
	Lattice Parameter b   : b = %s pixels', b);
    fprintf('
	Hexagonal             : %s', Hexagonal);
    fprintf('
	Angle                 : %s degrees', Angle);
    fprintf('
');
    fprintf('
Hit any key to continue...
-------------------------------------------------');
else fprintf('
Silent mode
');
end

ImgSize=str2num(ImgSize);
a=str2num(a);
b=str2num(b);
Hexagonal=str2num(Hexagonal);
Angle=str2num(Angle);
DeltaEnlarge=200;

%---------------------------------
%If Hexagonal call special function

if Hexagonal==1
    [ImgNoir Rayon SizeX SizeY ImgName] = LGHex(FlagPrint, ImgSize, a, Angle);
end
if Hexagonal==0
    [ImgNoir Rayon SizeX SizeY ImgName] = LGSquare(FlagPrint, ImgSize, a, b, Angle);
end

%---------------------------------
%Dilating atoms
if (FlagPrint==1) fprintf('
Dilating atoms ...'); end
se = strel('disk',Rayon);
ImgCircle = imdilate(ImgNoir,se);

% Enlarging image
if (FlagPrint==1) fprintf('
Enlarging image ...'); end
ImgGrande=zeros(SizeY+DeltaEnlarge,SizeX+DeltaEnlarge);
ImgGrande(DeltaEnlarge/2:(DeltaEnlarge/2+SizeY-1),DeltaEnlarge/2:(DeltaEnlarge/2+SizeX-1))=ImgCircle;

% Rotating image
if (FlagPrint==1) fprintf('
Rotating image ...'); end
ImgRotTemp = imrotate(ImgGrande,-Angle,'bilinear');

% Bluring the image
if (FlagPrint==1) fprintf('
Computing gaussian filter...'); end
H = fspecial('gaussian', 12, 4);
ImgCircleBlur = imfilter(ImgRotTemp, H , 'replicate');

% Croping image
if (FlagPrint==1) fprintf('
Croping image ...'); end
[ESizeY ESizeX]=size(ImgCircleBlur);
ImgFinal=ImgCircleBlur(ESizeY/2-SizeY/3:ESizeY/2+SizeY/3-1,ESizeX/2-SizeX/3:ESizeX/2+SizeX/3-1);

% Showing image
if (FlagPrint==1) fprintf('
Showing images ...');
image(ImgFinal); hold on; colormap(gray); axis off; hold off;
end

% Saving the image
ImgSave = uint8(ImgFinal);
max = max(max(ImgSave));
max = double(max)/256;
min = min(min(ImgSave));
min = double(min)/256;
ImgSave = imadjust(ImgSave, [min max], [0 1]);
imwrite(ImgSave,ImgName,'tif');

fprintf('
End!');
function [FlagPrint, ImgSize, a, b, Hexagonal, Angle] = LGReadConfigFile(ConfigFileName)

fid=fopen(ConfigFileName);
while 1
    tline = fgetl(fid);
    if (~ischar(tline)), break, end
    idx = findstr(tline, '=');
    if(strcmp(tline(1:idx-1),'FlagPrint'))
        FlagPrint=tline(idx+1:end);
    else
        if(strcmp(tline(1:idx-1),'ImgSize'))
            ImgSize=tline(idx+1:end);
        else
            if(strcmp(tline(1:idx-1),'a'))
                a=tline(idx+1:end);
            else
                if(strcmp(tline(1:idx-1),'b'))
                    b=tline(idx+1:end);
                else
                    if(strcmp(tline(1:idx-1),'Hexagonal'))
                        Hexagonal=tline(idx+1:end);
                    else
                        if(strcmp(tline(1:idx-1),'Angle'))
                            Angle=tline(idx+1:end);
                        else
                            end
                        end
                    end
                end
            end
        end
    end
end
fclose(fid);
function [ImgNoir, Rayon, SizeX, SizeY, ImgName] = LGSquare(FlagPrint, ImgSize, a, b, Angle)

% Fixed Global Variables
if (FlagPrint==1) fprintf('Global variables...'); end
Rayon=round( (a + b)/2/6);
ImgName = sprintf('Lattice(a=%d b=%d angle=%d).tif', a, b, Angle);

% Putting black points
if (FlagPrint==1) fprintf('Creating image...'); end
ImgSize=ImgSize*1.5;
SizeX=ImgSize; SizeY=ImgSize;
ImgNoir=zeros(SizeX,SizeY);

StepX=a;
StepY=b;
cont=0;
for y=1:StepY:SizeY
    for x=1:StepX:SizeX
        ImgNoir(y,x)=255;
    end
    cont=cont+1;
end
function [ImgNoir, Rayon, SizeX, SizeY, ImgName] = LGHex(FlagPrint, ImgSize, a, Angle)
% --------------------------------------------------------------------
% Fixed Global Variables
if (FlagPrint==1) fprintf('Global variables...'); end
Rayon=round(a/6);
ImgName = sprintf('LatticeHex(a=%d angle=%d).tif', a, Angle);

% --------------------------------------------------------------------
if (FlagPrint==1) fprintf('Creating image...'); end
ImgSize=ImgSize*1.5;
SizeX=ImgSize; SizeY=ImgSize;
ImgNoir=zeros(SizeX,SizeY);
StepX=a;
StepY=round(a*sqrt(3)/2);
cont=0;
for y=1:StepY:SizeY
   for x=1:StepX:SizeX
      if(mod(cont,2)==0)
         idx=round(x+StepX/2);
         if(idx>SizeX) continue; end
         ImgNoir(y,idx)=255;
      else
         ImgNoir(y,x)=255;
      end
      cont=cont+1;
   end
end
APPENDIX B: LATTICE PARAMETER ANALYSER

ReadMe

README FOR THE LATTICE PARAMETER ANALYSER (LPA)
---------------------------------------------------------

This file contains basic information on how to run and test the Lattice Parameter Analyser program.

PRINCIPLE OF LPA
-------------------------------

The LPA Matlab program analyses High Resolution Electron Microscopy (HREM) pictures obtained with Transmission Electron Microscope (TEM). Such images show an atomic plane of the material observed. This program aims at extracting information on the lattice of atoms.

The LPA analyses the distance between atoms and computes the distance between the first neighbours of the lattice. Therefore, the user obtains automatically precious information about the atomic lattice.

INSTRUCTIONS
-------------------------------

This Matlab program runs with the Matlab Version 6.5 Release 13. The user’s Matlab version must feature the Image Processing Toolbox, the Curve Fitting Toolbox and the Signal Processing Toolbox.

Simply run with Matlab the "Main_LatticeParamAnalyser.m" program.

It will read the input file "LPAConfig.txt", analyse the "Lattice(a=20 b=20 angle=10).tif" image and the results will be found in the directory "Lattice(a=20 b=20 angle=10) Results" created by the program and in the Matlab window.

INPUT FILE
-------------------------------

The input of this program is the text file "LPAConfig.txt" containing the following information:

- **ImgName:** Name of the image to be processed.
- **FlagPrint:** if 0 the program will run in silence. if 1 the program will show the program status and the results on the screen.
- **Scale:** Scale of the picture analysed so that the program reveals results in nanometers. The scale is the distance that one pixel of the image represents (spacial resolution).
- **Neighbourhood:** Distance in nanometers within which the program will compute the distance between atoms.

Additional parameters only to use in case of program optimization:
- RDFThreshold: value used in the function FindMax (that finds the position of the peaks in the histogram of the distances). It is a type of threshold used to separate peaks that are close to each other.
- Filter Order: order of the filter that smoothes the Radial Distribution Function (histogram of the distances)

The file "LPACfg.txt" is set as followed:

- ImgName = Lattice(a=20 b=20 angle=10).tif
- FlapgPrint = 1
- Scale = 0.03
- Neighbourhood = 1,5 nm
- RDFThreshold = 0.05
- Filter Order = 10

These parameters can be modified if desired (eg if the user wants to test another image). The images cannot be colored images.

OUTPUT FILE

The LPA program will create a file named after the analysed picture. After the LPA ran, it will contain:

- Lattice(a=20 b=20 angle=10)_dist.dat : data file containing all the distances computed. If the user wishes, the data can be studied to draw histograms.
- Lattice(a=20 b=20 angle=10)_peakpos.dat : data file containing the position in nanometers of the first neighbours found by the LPA.
- Lattice(a=20 b=20 angle=10)_peak.jpg : graph of the peaks of the histogram, showing the distance of the first neighbours.
- Lattice(a=20 b=20 angle=10)_segmented.jpg : image resulting from the image processing. The user can check that the binary image used for the distance analysis is correct.

IMAGE "Lattice(a=20 b=20 angle=10)"

The "Lattice(a=20 b=20 angle=10).tif" image simulates a typical square lattice of atoms. It has been generated by another Matlab program developed by the same authors (Lattice Generator LG).

The measurement of the first neighbour distance is 20 pixels, so with the arbitrary scale of 0.03 nm, it represents about 0.60 nanometers.

MORE INFO ABOUT LPA

--------------------------------------------------------------------

--------------------------------------------------------------------
More detailed explanations about the LPA and LG programs can be found in the webpage:
http://www.cbpf.br/cat/lpdsi/lpa

AUTHORS

Marcelo Portes de Albuquerque
Contact: marcelo@cbpf.br

Carole Jonville
Contact: carole@cbpf.br
    jonville@dcmm.puc-rio.br
    carole.jonville@enspg.inpg.fr
Configuration File

---
# Program     : Lattice Parameter Analyser
# Written by  : Carole Jonville & Marcelo Portes
# email for   : carole@cbpf.br
#               jonville@dcmm.puc-rio.br
#               carole.jonville@enspg.inpg.fr
#               marcelo@cbpf.br
# Copyright(©): (c) CBPF - CAT
# Version      : 4.0816
# Objective    : Extract information on the atomic lattice from HREM image
# Project      : PFE ENSPG - PUC Rio - CBPF
---

# ImgName: name of the image to be analysed
ImgName=Lattice(a=20 b=20 angle=10).tif

# FlagPrint=1 shows the results on the screen
# set to 0 if want the program to run in silence
FlagPrint=1

# Scale: distance value of 1 pixel in nanometers (spatial resolution)
Scale=0.03

# Neighbourhood: distance in nanometer in the neighbourhood for Radial Distribution
# Function (the distance analysis)
Neighbourhood=1.5

# Additional parameters only to use in case of program optimization:
---

# RDFThreshold: value used in the function FindMax (that finds the position of the peaks in the histogram of the distances). It is a type of threshold used to separate peaks that are close to each other.
RDFTh=0.05

# Filter Order: order of the filter that smoothes the Radial Distribution Function (histogram of the distances)
RDFFilterOrder=10
Main_LatticeParameterAnalyser

clear all; close all; clc;
warning off;
% --------------------------------------------------------------------
---
% Cabecalho
fprintf('
Lattice Parameter Analyser - LPA Version 4.0816');
fprintf('
Centro Brasileiro de Pesquisas Fisicas - CBPF Brazil');
fprintf('
Ecole Nationale Superieure de Physique de Grenoble - ENSPG
France');
fprintf('
Pontificia Universidade Catolica do Rio de Janeiro - PUC
Rio Brazil\n')
%---------------------------------------------------------------------
---
%Read the configuration file
[FlagPrint ImgName Neighbourhood Scale RDFTh RDFFilterOrder] =
ReadConfigFile('LPAConfig.txt');

FlagPrint=str2num(FlagPrint);
if (FlagPrint==1)
    fprintf('
-------------------------------------------------------
');
    fprintf('
Input parameters...
	ImgName          : %s', ImgName);
    fprintf('
	FlagPrint        : %d', FlagPrint);
    fprintf('
	Scale            : 1 pixel = %s nm', Scale);
    fprintf('
	Neighbourhood    : %s nm', Neighbourhood);
    fprintf('
	RDF Threshold    : %s', RDFTh);
    fprintf('
	RDF FilterOrder  : %s', RDFFilterOrder);
    fprintf('
');
    fprintf('
Hit any key to continue...
-------------------------------------------------------
');
    pause;
else fprintf('
Silent mode');
end

Scale=str2num(Scale);
Voisinage=str2num(Neighbourhood);
RDFTh=str2num(RDFTh);
RDFFilterOrder=str2num(RDFFilterOrder);
Voisinage=Voisinage/Scale;
%---------------------------------------------------------------------
---
%Processes the image
[ImgSEB, ImgBW] = ProcessImage(ImgName, FlagPrint);

%---------------------------------------------------------------------
---
%Computes distance
[Dist, cont, CoordObjAtCenter] = DistMeasurement(ImgSEB, FlagPrint,
Voisinage);
%---------------------------------------------------------------
% Finds the peaks on the histogram

[sinal2, Intensity, PeakPos, Deviation, xout, DeltaBin] = 
FindMaxGauss(Dist, Scale, FlagPrint, RDFTh, RDFFilterOrder);

% -----------------------------------------------
% Saves data and images

SavingData(FlagPrint, ImgName, ImgSEB, Dist, xout, Scale, sinal2, 
Intensity, PeakPos, Deviation, cont, DeltaBin, CoordObjAtCenter);

% -----------------------------------------------
% End

fprintf('
End !!')
ReadConfigFile

function [FlagPrint, ImgName, Neighbourhood, Scale, RDFTh, RDFFilterOrder] = ReadConfigFile(ConfigFileName)

fid=fopen(ConfigFileName);
while 1
    tline = fgetl(fid);
    if (~ischar(tline)), break, end
    idx = findstr(tline,'=');
    if(strcmp(tline(1:idx-1),'FlagPrint'))
        FlagPrint=tline(idx+1:end);
    else
        if(strcmp(tline(1:idx-1),'ImgName'))
            ImgName=tline(idx+1:end);
        else
            if(strcmp(tline(1:idx-1),'Neighbourhood'))
                Neighbourhood=tline(idx+1:end);
            else
                if(strcmp(tline(1:idx-1),'Scale'))
                    Scale=tline(idx+1:end);
                else
                    if(strcmp(tline(1:idx-1),'RDFTh'))
                        RDFTh=tline(idx+1:end);
                    else
                        if(strcmp(tline(1:idx-1),'RDFFilterOrder'))
                            RDFFilterOrder=tline(idx+1:end);
                        end
                    end
                end
            end
        end
    end
end
fclose(fid);
function [ImgSEB, ImgBW] = ProcessImage(ImgName, FlagPrint)

% Reading the image.
if (FlagPrint==1) fprintf('Reading the image...'); end
ImgOriginal=imread(ImgName);%ImgOriginal is RGB thus 3D
if (isrgb(ImgOriginal)==1) ImgOriginal=ImgOriginal(:,:,1); end;% If
image is RGB makes it grey scale.

% Corrects the background.
if (FlagPrint==1) fprintf('Computing background...'); end
tic;
H = fspecial('gaussian', 100, 30);
ImgBlurGaus = imfilter(ImgOriginal, H, 'replicate');
t1=toc;
if (FlagPrint==1) fprintf(' (it took %4.2f s !)', t1); end
if (FlagPrint==1) fprintf('Removing background ...'); end
ImgCorrected = double(ImgOriginal)*.5-double(ImgBlurGaus)*.5;
ImgCorrected = uint8(ImgCorrected);

% Adjusts the contrast.
if (FlagPrint==1) fprintf('Adjusting the contrast...'); end
ImgAjusted = imadjust(ImgCorrected, stretchlim(ImgCorrected), [0 1]);

% Creating a binary image.
if (FlagPrint==1) fprintf('Creating a binary image...'); end
level = graythresh(ImgAjusted);
ImgBW = im2bw(ImgAjusted, level); % Makes ImgAjusted binary using a
threshold value of level

%Removing small irrelevant objects
ImgSeparated=bwmorph(ImgBW, 'open', 3);

%Removing image edge particles.
if (FlagPrint==1) fprintf('Removing image edge particles...'); end
ImgSEB=imclearborder(ImgSeparated,4);
Circle

function H=circle(center,radius,NOP,style)
%---------------------------------------------------------------------
---% H=CIRCLE(CENTER,RADIUS,NOP,STYLE)
% This routine draws a circle with center defined as
% a vector CENTER, radius as a scaler RADIS. NOP is
% the number of points on the circle. As to STYLE,
% use it the same way as you use the routine PLOT.
% Since the handle of the object is returned, you
% use routine SET to get the best result.
% Usage Examples,
% circle([1,3],3,1000,':');
% circle([2,4],2,1000,'--');
% Zhenhai Wang <zhenhai@ieee.org>
% Version 1.00
% December, 2002
%---------------------------------------------------------------------
---

if (nargin <3),
    error('Please see help for INPUT DATA.');
elseif (nargin==3)
    style='r-';
end;
THETA=linspace(0,2*pi,NOP);
RHO=ones(1,NOP)*radius;
[X,Y] = pol2cart(THETA,RHO);
X=X+center(1);
Y=Y+center(2);
H=plot(X,Y,style);
axis square;
DistMeasurement

function [Dist, cont, CoordObjAtCenter] = DistMeasurement(ImgSEB,
FlagPrint, Voisinage)

%---------------------------------------------------------------------
---
% Labeling objects.
if (FlagPrint==1) fprintf('Labeling objects...'); end
[ImgLabeled,numObjects] = bwlabel(ImgSEB,4);% Label components.
if (FlagPrint==1) fprintf('(%d)!',numObjects); end

% --------------------------------------------------------------------
---
% Tracking features.
if (FlagPrint==1) fprintf('Tracking features...'); end
Objects=imfeature(ImgLabeled, 'Centroid'); %Computes only mass center.
[SizeY SizeX] = size(ImgLabeled);

%---------------------------------------------------------------------
---
%Computing distances between centroid of all objects.
numCalc=numObjects;

if (FlagPrint==1) fprintf('Computing distances between objects...'); end
cont=1;
tic;
DistCenter=inf;
for i=1:numCalc-1
    DistCenterTmp=sqrt( ( Objects(i).Centroid(1) - SizeX/2 )^2 + ( Objects(i).Centroid(2) - SizeY/2 )^2 );
    if (DistCenterTmp<=DistCenter)
        DistCenter=DistCenterTmp;
        CoordObjAtCenter = [Objects(i).Centroid(1),
Objects(i).Centroid(2)];
    end
    for j=i+1:numCalc
        DistTmp=sqrt( ( Objects(i).Centroid(1) - Objects(j).Centroid(1) )^2 + ( Objects(i).Centroid(2) - Objects(j).Centroid(2) )^2 );
        if (DistTmp<=Voisinage)
            Dist(cont)=DistTmp;
            cont=cont+1;
        end
    end
end
t2=toc;
if (FlagPrint==1) fprintf(' (it took %4.2f s !)', t2); end

54
FindMaxGauss

function [sinal2, Intensity, PeakPos, Deviation, xout, DeltaBin] = FindMax(Dist, Scale, FlagPrint, RDFTh, RDFFilterOrder)

if (FlagPrint==1) fprintf('Finding peaks on histogram...'); end

nbins = 500;
[HistCont, xout] = hist(Dist,nbins);

if (FlagPrint==1) fprintf('Smoothing signal ...'); end
h=ones(1,RDFFilterOrder);
sinal2 = convn(HistCont,h,'same');
sinal2 = sinal2 ./ max(sinal2);
sinal3 = imextendedmax(sinal2,RDFTh); % Find maximas, threshold to determine

% search maximas range and peaks
FlagOnTop=0; count=0;
for i=1:length(sinal3)
    if(sinal3(i)==1 & FlagOnTop==0)
        FlagOnTop=1;
        LeftRange=i;
    else
        if(sinal3(i)==0 & FlagOnTop==1)
            FlagOnTop=0;
            RightRange=i;
        end
        %Enlarge Range to best fit gaussians
        LeftRange=LeftRange-2;
        if LeftRange<1 LeftRange=1; end
        RightRange=RightRange+2;
        if RightRange>length(sinal3) RightRange=length(sinal3);
    end
end

%Fit gaussians
ftype = fittype('gauss1');
Xdata = xout(LeftRange:RightRange);
Xdata = Xdata.';
Ydata = sinal2(LeftRange:RightRange);
Ydata = Ydata.';
gfit = fit(Xdata, Ydata, ftype);

count = count + 1;
Intensity(count) = gfit.a1;
PeakPos(count) = gfit.b1*Scale;
Deviation(count) = gfit.c1/sqrt(2)*Scale;
end
end

if (FlagPrint==1) fprintf('Showing Results ...'); end
DeltaBin=xout(2)-xout(1);
if (FlagPrint==1)
    for ind=1:count
        fprintf('
	Peak position(\%d) = (%4.4f \text{ nm} +/-%4.4f \text{ nm})', ind, PeakPos(ind), Deviation(ind));
    end
end
```matlab
function SavingData(FlagPrint, ImgName, ImgSEB, Dist, xout, Scale, sinal2, Intensity, PeakPos, Deviation, cont, DeltaBin, CoordObjAtCenter)

if (FlagPrint==1) fprintf('\nSaving data and images...'); end

if (FlagPrint==1) Option='on'; else Option='off'; end

rep = sprintf('%s Results',ImgName(1:end-4)); mkdir(rep);

%---------------------------------------------
% Saving Segmented Image

BWname = sprintf('.//%s//%s_segmented.jpg',rep,ImgName(1:end-4));

h=figure('Visible',Option,'Color',[1,1,1]); imshow(ImgSEB);
hold on; colormap(gray); axis square;

[trash count] = size(PeakPos);
for ind=1:count
    if(mod(ind,2)==0) ColorStr=sprintf('r-');
    else ColorStr=sprintf('g-');
    end
    H=circle(CoordObjAtCenter,PeakPos(ind)/Scale,5000,ColorStr);
end
hold off;
title('Segmented image'); saveas(gcf,BWname,'jpg');

%---------------------------------------------
% Saving distances

DataOutFileName = sprintf('.//%s//%s_dist.dat',rep,ImgName(1:end-4));

fid = fopen(DataOutFileName, 'w');

for i=1:cont-1
    fprintf(fid, '
%f',Dist(i)*Scale);
end

%---------------------------------------------
% Saving the peaks graph

t=(1:1:500);
DeltaX=xout(2)-xout(1);
t2=(DeltaX*t+xout(1)-DeltaX)*Scale;
h=figure('Visible',Option,'Color',[1,1,1]);
bar(t2,sinal2,'g'); set(gca,'Layer','top'); hold on;
tt=(0:0.00001:1);
hirest2=(tt*xout(end))*Scale;
```
[trash count] = size(PeakPos);
for ind=1:count
    EstSinal2=Intensity(ind)*exp(-((hirest2-
    PeakPos(ind))./Devi(ation(ind)*sqrt(2))).^2);
    plot(hirest2,EstSinal2,'r-');
    TextStr=sprintf('\leftarrow %4.4f',PeakPos(ind));
    text(PeakPos(ind), Intensity(ind), TextStr,
    'HorizontalAlignment','left')
end
grid on; hold off;
title(['First Neighbours Position of ', ImgName]);
xlabel('Distance in nm');
ylabel('Counts (relative scale)');
max=((xout(2)-xout(1))*500+xout(1))*Scale;
axis([0 max 0 2000]);
axis 'auto y';
NamePeak=sprintf('.'.$/s//s_peak.jpg', rep, ImgName(1:end-4));
saveas(gcf,NamePeak, 'jpg');
fclose(fid);

%---------------------------------------------------------------
---
%Saving peak information
DataOutPeakFile = sprintf('.//%s//%s_peakpos.txt', rep,
    ImgName(1:end-4));
fid = fopen(DataOutPeakFile, 'w');
for ind=1:count
    fprintf(fid, '
Peak number %d at %4.4f nm +/- %4.4f nm
', ind,
    PeakPos(ind), Deviation(ind));
end
fclose(fid);

%---------------------------------------------------------------
---
%End
if (FlagPrint==1)
    fprintf('
Segmented image                       : %s', BWname);
    fprintf('
Graph with first neighbours´ distances: %s',NamePeak);
    fprintf('
All distances within the neihgbourhood: %s',DataOutFileName);
    fprintf('
First neighbours´ distances           : %s',
    DataOutPeakFile);
end