High-speed atomic force microscopy on soft matter

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

by Arnaud Benard

To my parents…

Acknowledgements

Lausanne, 12 Mars 2011 D. K.

Preface

A preface is not mandatory. It would typically be written by some other person (eg your thesis director).

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Lausanne, 12 Mars 2011 T. D.

Abstract

Contents

List of Figures

List of Tables

1 Introduction

In 1986, a group of scientist from IBM research developed the first Atomic Force Microscope.[3] The original idea is to measure forces between a sharp tip and a sample. This force bends the cantilever according to Hooke's law. The deflection is measured using a laser reflected from the top of the cantilever into a photodetector(array of photodiodes). The application of AFMs ranges from measuring elastic properties of biological samples to imaging of graphite[12] [4] Using a XY-scanner to move the sample on the horizontal plane allows to acquire data on different areas. Also, an AFM has a feedback system to avoid the tip to crash on the surface. The PID loop maintains a constant tip-sample force.

The most conventional way to scan is with a raster pattern. This technique steers the tip of the AFM on specific points of a grid. Past AFM research has been focused on improving position controllers. Indeed, the AFM precision depends from the quality of the closed loop feedback on XY. These improvements, however, didn't solve fundamental problems with raster scanning. Most of the data is thrown away (trace and retrace) and the actual position on the XY plane is inaccurate - and directly correlated with the efficiency of the position controller.

Because of its limited bandwidth, fast raster scans generate distortions in the image.[16] Spiral pattern allows to generate high-quality images at higher scan frequencies than the raster one. [10]. Moreover, this method reduces the number of sample acquired. [6]

With raster scanning, the data is evenly distributed in space. Generating an image is trivial. Spiral scan, however, needs new techniques to render images. Fortunately, image processing algorithms like inpainting [13] or Delaunay triangulation have been developed to generate images from sparse data.[6].

The bandwidth on the z-axis control loop is limited by the dynamics of the z scanner.[7] We can achieve higher frequencies by using a small piezoelectric ceramic.[15]

In this thesis, we will see how we can use non-raster scan pattern to improve the bandwidth on the XY-plane. Also, we have developed image processing algorithms to render non-gridded data. We use the popular Open Graphics Library (OpenGL) for the rendering of the X, Y, Z data and investigate ways to use the Graphics Processing Unit (GPU) to improve the computing time. Finally, we will investigate new ways to go beyond the limitations on the z-axis with tilt correction and dual actuators feedback on z. If an image is tilted, you can use first-order plane fitting to flatten the image. A more efficient way to to dynamically compensate for the tilt of the sample. Also we will see how to improve the bandwidth on the z-axis by introducing a small piezoelectric ceramics.

2 Visualization of non-gridded data

First, we discuss using sensor data instead of theoretical one and its implications in non-raster scanning. Then, we review how to render images from sparse data with Delaunay triangulation and inpainting.

Current AFMs run in closed loop: position controllers on the XY axis are needed to steer the tip at the right position. Instead of using this method, we will work in open loop and register the data of the position sensors. One of the advantage of using sensor data is that we don't need accurate position controllers: it has no impact on our data. Instead of position the tip on an exact position, we're embracing its inaccuracy. The precision of our system is limited by the sensors and not the feedback on XY. Our current setup (Asylum Research MFP3D) uses inductive sensors.

Using this technique frees us from raster scan but as we don't acquire specific points on a grid, we need image processing algorithms to render missing parts of our scan. The easiest way is to have a linear gradient between the closest points. We will see in the next section how to find these points.

2.1 Image rendering techniques

We will see two ways to render images from sparse data: inpainting and OpenGL.

2.1.1 Inpainting algorithms

Reconstructing missing parts of images was first developed for restoring photographs and paintings or remove undesirable data like text and publicity. The art of restoration was performed manually. Nowadays, tools like Photoshop or Gimp are widely used in the media. It can also be used to produce special effects [13].

This process is called inpainting. The principle behind it is to fill a patch with its surroundings.

Mathematicians have developed wide range of algorithms to solve that kind of problems. We will investigate a special case of partial differential equations (PDE): heat equations. The heat equation is a PDE that represents the distribution of heat in a region over time.

$$\frac{\partial u}{\partial t} - \alpha \nabla^2 u = 0 \tag{2.1}$$

α is the thermal diffusivity - that is interpreted as a "thermal inertia modulus" - and u is the temperature over space and time (i.e. $u(x, y, z, t)$). A high thermal diffusivity implies that the heat moves rapidly.

The algorithm has been implemented on Matlab by the Paul Ashby Group at LBNL which developed inpainting for AFM in collaboration with Prof. A. Bertozzi of UCLA. This algorithm will spread out the information of each point on missing parts of our grid.

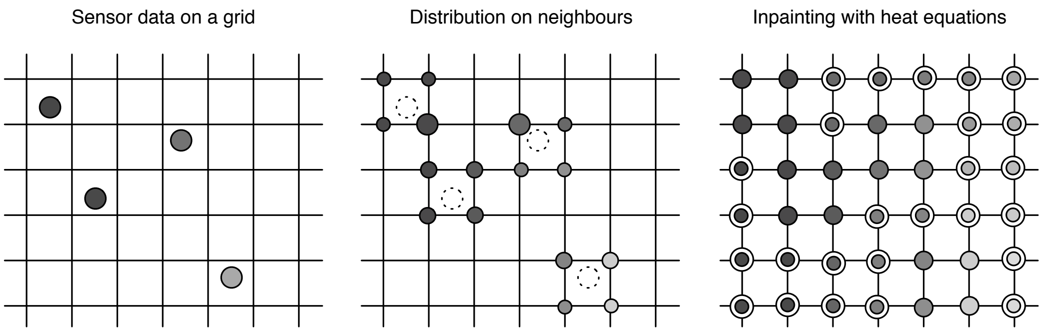

Figure 2.1: The first image shows that we don't have enough informations to fill out all the pixels. We need to project the data on the neighbours. Finally, the heat equation inpainting will spread the weighted data on the missing spots.

Literature shows that heat equations are powerful to fill out these patches of missing data, but it smooths data on sharp edges(high frequency data) [2] . One of the effect is that edges are blurred out by the algorithm.

2.1.2 OpenGL

In this section, we see how to render images with OpenGL(Open Graphics Library). It is an API (Application Programming Interface) developed by Silicon Graphics to hide the complexities of interfacing with different 3D accelerators and mainly used for 3D modeling in video games and simulations. OpenGL leverages the fact that GPUs are designed to render triangle. It optimizes the rendering by using the repetition of a geometric shape (tessellation). The more complex the shape the harder it will be for the GPU to process it. If we already pre-process the data into triangles, we minimize processing costs. [1]

The flow of our program is described by the Fig 2.2. First, we take the X, Y, Z data and generate a list of triplets of points (triangles). Then, we render this list with OpenGL. We will see two methods to render to list of triplets: immediate mode and Vertex Buffer Objects.

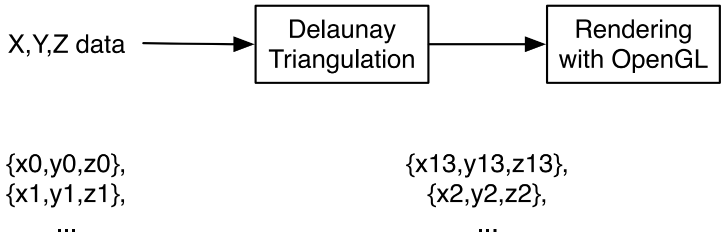

Figure 2.2: The

Triangulation with Delaunay

The first problems we face how to generate triangles from sparse data. Indeed, OpenGL can only render triangles from a triplet of points. If we render the X, Y, Z data without preprocessing, OpenGL will generate triplets from the closest index in the list. We need efficient algorithms like Delaunay triangulation to take into account the X, Y position of the point and not the index.

The algorithm minimizes the angles of each triangle. The triangulation is successful if no vertex (i.e. 3-dimensional point) is inside a triangle.

Jonathan Shewchuk [14] has developed a library, triangle.c, to compute Delaunay triangulations and other meshes. The Figure 2.3 shows the effect of the program on a spiral scan. We will discuss later about spiral scanning. The input data has 20'000 points and the program generates 38'784 triangles.

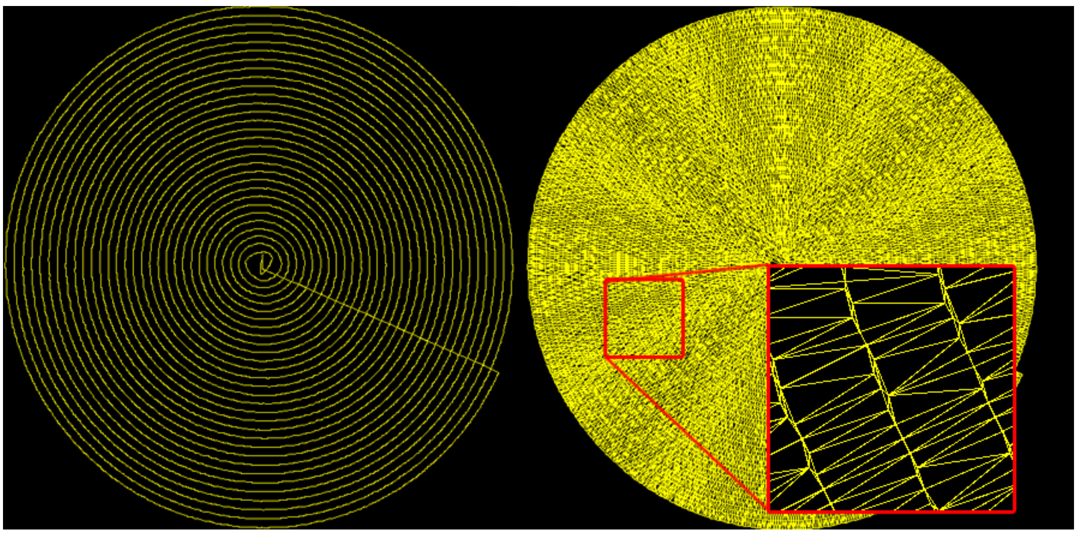

Figure 2.3: Before and after triangulation. This image is rendered by using the triangle.c library.

With this method we can connect the points on a 2D plane. In the figure 2.4, we use the z-axis data to compute the height of each of our point and render 3D models of our scans. To add colors to our data, OpenGL will create a linear gradient between each of the data points.

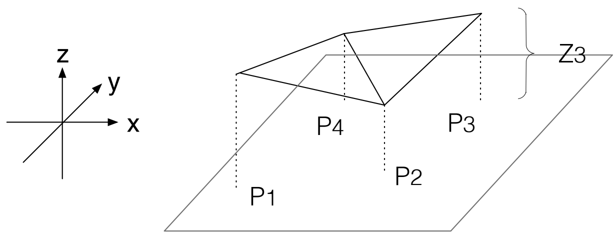

Figure 2.4: Delaunay triangulation: From 2D to 3D

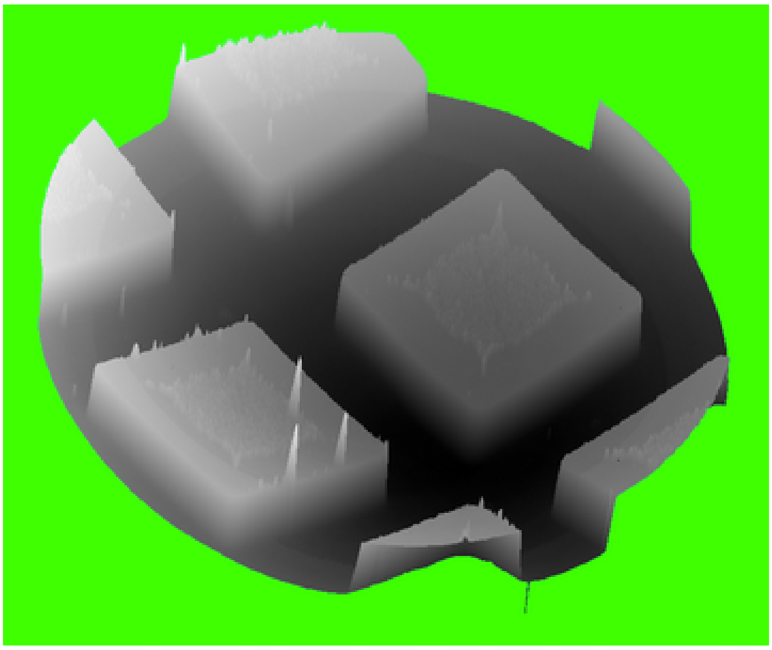

Figure 2.5: 3d rending of 1.2M points

Immediate mode vs Vertex Buffer Objects

There are two ways to render our surface with OpenGL: the immediate mode and vertex buffer objects.

The immediate mode is the simplest implementation of OpenGL. Indeed, we render every frame. If we rotate the our 3D model, we'll have to regenerate the latter. The power of the immediate mode is its simple implementation (no initialization and extra code). Moreover, it is easier to debug. For a small number of vertices (< 10'000) the immediate mode is appropriate. [8] states that the immediate mode is more convenient and less overhead than other implementations (Vertex Buffer Objects).

The display function is called when GLUT(OpenGL Utility Toolkit) determines that the windows needs to be redisplayed. Action like rotation, translation or resizing of the model will trigger the display event. Each time the display function will be called, the program will upload the vertices to the GPU.

If we try to display a significant number of triangles (> 10'000 vertices), the CPU will be the bottleneck. The GPU doesn't start rendering data before the last callback ($glEnd()$). Thus, the CPU is spoon-feeding the GPU by transferring the data triangle by triangle. Moreover, the number of API calls is proportional to the number of triangles. I.e. if you have 10 triangles you will make (10*(2+3+3)) 80 API calls [11]. In conclusion, if you want to render less than 10'000 vertices, code a quick implementation and are not planning on making a lot of changes in your rendering, the immediate mode is the way to go.

One of the problem we have encountered with the immediate mode is the transfer from the system memory to the GPU. We've seen there is a bottle neck in the transfer. With 10'000 points we can only have 3 frames per seconds. It means that our computer takes 300 ms for the whole process.

Instead of transferring the data from the memory to the GPU, the GPU could read the memory of the program. Buffer objects have been created to allow the GPU to have access to the memory. The process of reading the memory from the GPU is called Direct Memory Access (DMA). A buffer object is a contiguous untyped memory which the CPU and the GPU have access to.

We can't just upload our data into the memory without any structure. We need to map the data and make it readable for the GPU. We store our data in a vertex array object.

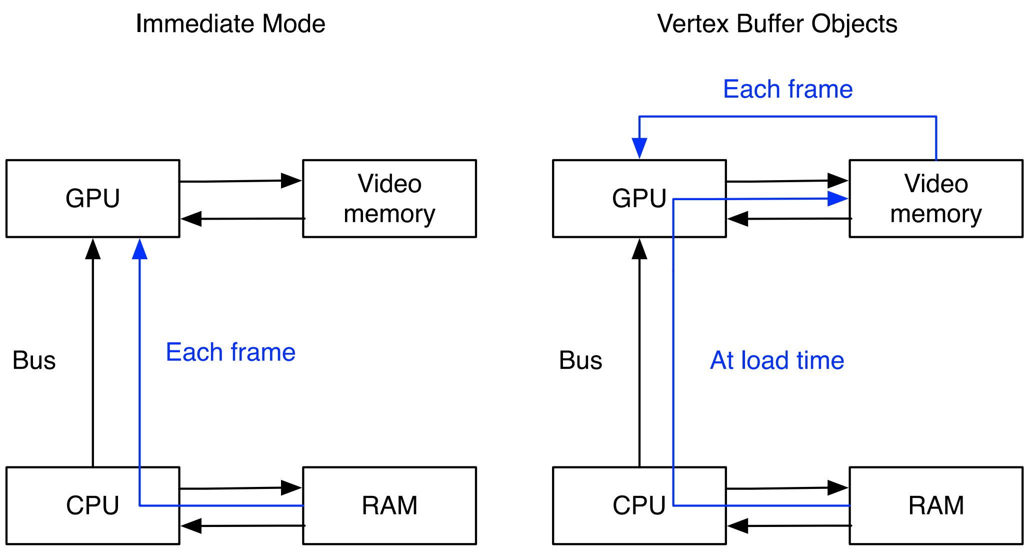

Figure 2.6: Workflow of the immediate mode and the vertex buffer objects

After having allocated and created this chunk of data, we need to map it to make it readable to the GPU. OpenGL has API calls for that application.

The advantage of this implementation is that you directly pull your data to a shared memory between the CPU and the GPU. Your CPU will spend less cycles making API calls thus improving the performances of the program. The power of the VBOs is that you just need to upload your data and your display function will just bind the VBO. Our performances have improved from 3FPS to 130FPS for 100'000 data points. Having a higher FPS count makes the animations smoother.

Table 2.1 shows the non-linearity of our implementation. We see that Delaunay triangulation doesn't scale well for 1'000'000 points. In AFM scans we will rarely sample 1'000'000 datapoints. The limits of our AFM is 100'000 kHz. If we take 10 seconds scans at the limit rate, we observe that the computation time is still way below the scanning time.

Table 2.1: Rendering results[ms]

Nb of points	Delaunay	VBO	Total time
1000	2.9	23.9	26.8
10000	8.1	27	35.1
100000	66.9	181	247.9
1000000	640.7	267	907.7

3 Techniques for fast z feedback

Implementing model based controller or high frequency actuators improves the AFM feedback loop[15]. The drawback with these actuators is the decrease in the positioning range.

We improve the bandwidth and the scan range of our device. Past research has introduced an external piezoelectrical actuator on top of the cantilever. With this scheme, we can combine the advantages of a high bandwidth from the piezo-electrical actuator and the long range of the tip.

3.1 Double PID

The principal feature of an AFM is its probing system. The feedback control system is designed to adjust the motion of the tip on the z-axis. It will adjust the tip-to-sample distance.

The figure 3.1 shows the most classic feedback loop in an AFM. The reference of the controller is the force setpoint. The output of the controller is the topography of our surface. Also, the error signal is the sum between the reference and the deflection of the cantilever.

The problem with this system is that it can't pick up abrupt changes in the topography.[7] The bandwidth of the cantilever is not large enough. This leads to scanning artefacts like parachuting and /ADD OTHER. We could use a cantilever with a higher bandwidth, but the amplitude would be too small. We decided to take to the best of both world and design a dual-actuator system. We use a standard cantilever to scan the surface and we will add a small piezoelectrical ceramic on top of the X-Y scanner. It enables the AFM to have a larger bandwidth. Thus, it picks up high spatial frequency topography.

We design an other feedback system in order to make the X,Y,Z scanner and the piezoelectrical ceramic work in pair.

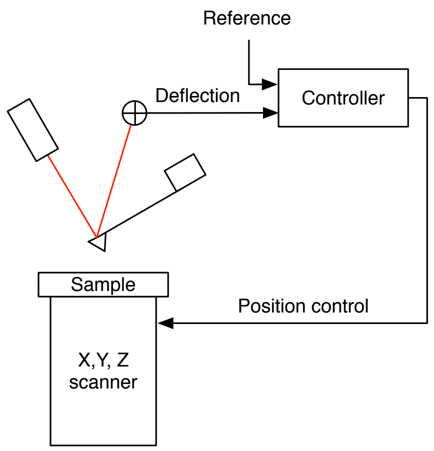

Figure 3.1: Normal AFM setup

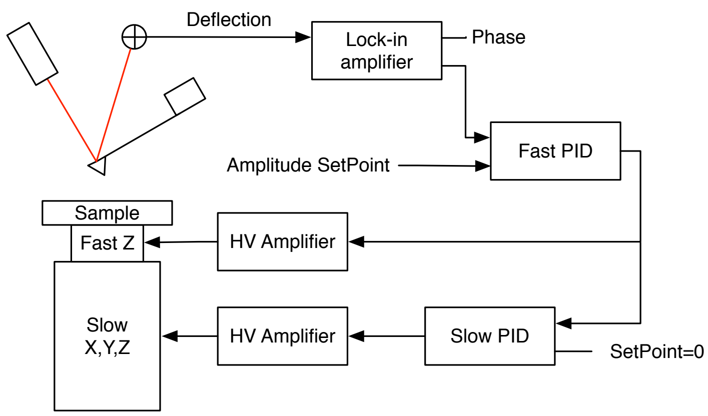

Figure 3.2: Double PID

We need a high-voltage amplifier to operate the small piezoelectric device.

3.2 Tilt compensation

In theory, the mounted sample should be flat: parallel to the XY-Scanner. If the probe/sample angle is not perpendicular, we observe a tilt on the surface. This tilt is problematic when it becomes larger than the features. There are multiple ways to compensate for this. The most common technique is to use post-processing to adjust the image. Flattening algorithms or first-order planefitting restore the image and put the data on the same level. This technique works if the range of the tip is large enough. We have decided to take another approach and to dynamically compensate for the tilt. Before performing our scan, we will do a first scan to compute the tilt of the sample by considering our tilt as a 3D plane. Then, we'll generate a tilt correction signal that will be added to z scan output.

We use a circle pattern to scan the surface of the sample. The radius of the circle is equal to the scan size. It gives us informations about the general topography of the surface. Then, we compute the plane equation of the surface by applying a fit in Igor Pro. This fit will generate a plane that models the tilted surface. The input of this fit are the theoretical X-Y output waves of the circle pattern: a sinus and a cosines. The data on the z axis is the height.

$$z = a_1 x + a_2 y + a_3 \tag{3.1}$$

The coefficients by minimizing the values of Chi-Square (error function). Then, we generate the waves to send to the controller the following way.

$$wavetosend = a_1 wavex + a_2 wavey \tag{3.2}$$

Wavex and wavey are the output of our scan pattern. With this method, we can do a tilt correction with any scan pattern.

Then, we send the previously computed wave to the controller.

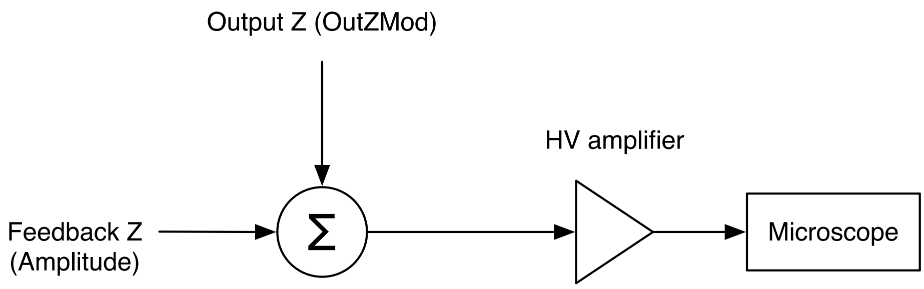

Figure 3.3: Asylum

4 Results

4.1 Calcite experiment

In this experiment we've observed the dissolution of calcite ($CaCO_3$) with water. We have decided to choose this reaction because the chemical reaction leading to the dissolution is simple. Moreover, it is abundant and geologically important material.[5] [9]

/ ADD SCHEMA of the setup

We have used our double PID feedback system and /InsertTipType. Calcite dissolution is an interesting process to observe with a high speed AFM. The whole process lasts X min which is observable with our setup. With a commercial AFM, that takes a few minutes for a raster scan, the dissolution process is hard to monitor (only a few pictures).

$$CaCO_3(calcite) + H_20 \rightarrow Ca^{2+} + HCO_3^- + OH^- \tag{4.1}$$

We have mounted our calcite sample on top of the fast piezo-electrical ceramics. Also, we have stuck a plastic cover underneath the piezo for the water. THe calcite will dissolve itself with a losange pattern. It is linked to its original geometry.

We have imaged the same sample with different parameters: scan size, number of spirals and scanning time.

The Figures 4.1 to 4.6 show the evolution of the calcite. Every scan took 10 seconds over a surface of 10 μm. The scan pattern is an archimidean spiral with 50 loops. We observe the evolution of the front wave over time. After a few minutes, the water will saturate the calcite and the reaction will stop. You can redo the experiment by removing the water and adding new one afterwards.

It shows the wide range of experiments possible in the future. Inde

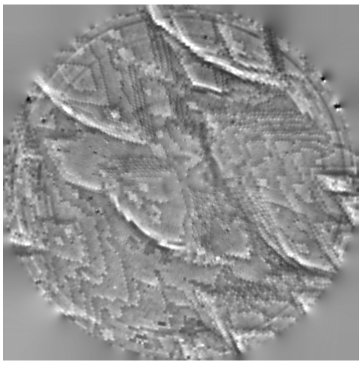

Figure 4.1

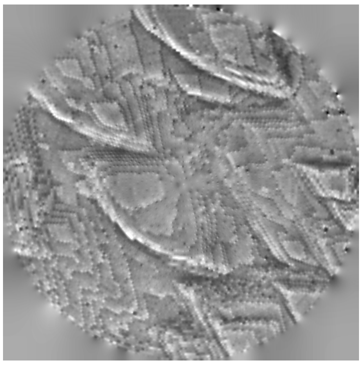

Figure 4.2: A really Awesome Image

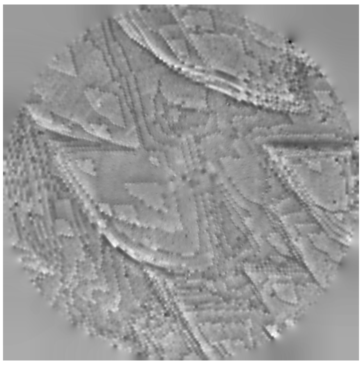

Figure 4.3: A really Awesome Image

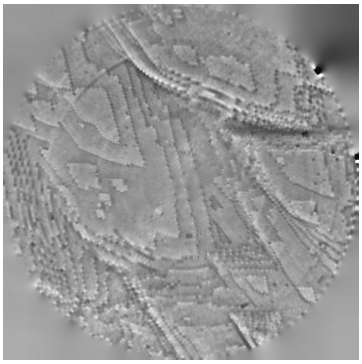

Figure 4.4: A really Awesome Image

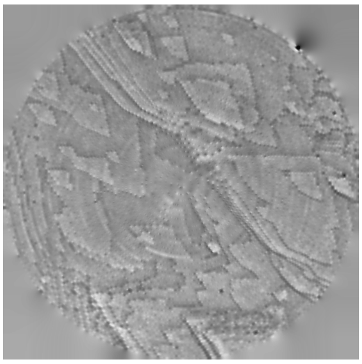

Figure 4.5: A really Awesome Image

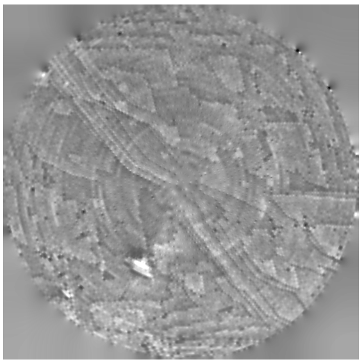

Figure 4.6: A really Awesome Image

4.2 Tilt correction

Wavex and wavey are the x,y components of our scan pattern. The scan pattern for an Archimedean spiral is defined by the equation 4.2.

$$x(t) = \alpha\sqrt{t}cos(\beta\sqrt{t}) \qquad y(t) = \alpha\sqrt{t}sin(\beta\sqrt{t}) \tag{4.2}$$

We took a calibration sample to test the efficiency of our method. The size or our scan is 30 μm. We see on the figure 4.7 that the surface has a tilt. The small spikes on the curve represent the pyramids on the sample.

Then we compute our fitting algorithm on the x,y and z data.

Table 4.1: Planefit coefficients

a_1	a_2	a_3
-0.14615	-0.031882	29.537

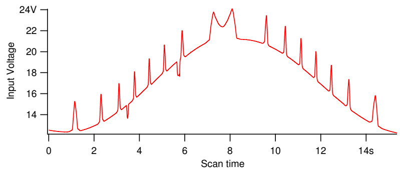

Figure 4.7: Height of the tilt correction

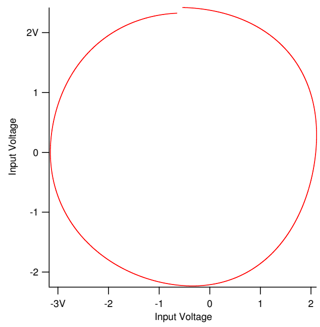

Figure 4.8: Path on the XY plane

The size of the scan is still 30 μm and the spiral has 80 loops. The scan pattern we are going to send to the controller is generated with the previously computed planefit coefficients.

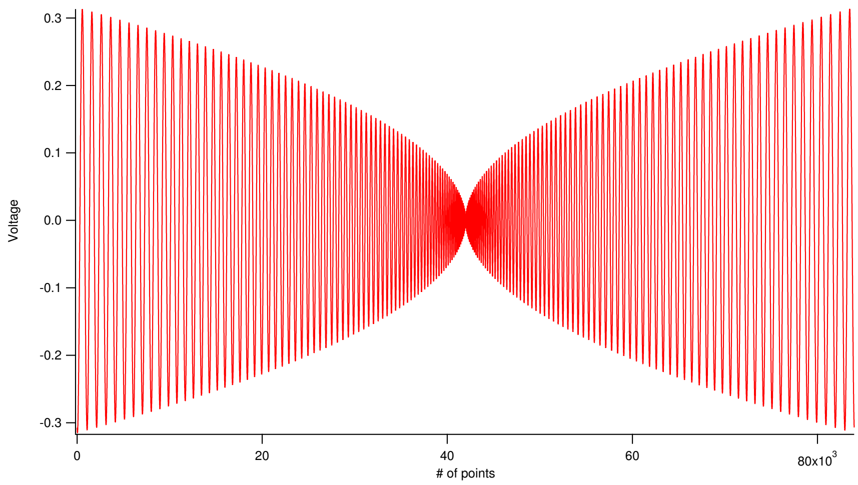

Figure 4.9: Input of the tilt compensation

We have calibrated the small piezoelectric ceramic and found that a step of 5V is equal to 90 nm.

The tilt compensation takes a load off the small fast piezoelectrical ceramics. The Figure 4.12 shows the efficiency of our method. Indeed, the fast piezo was previously saturating. The

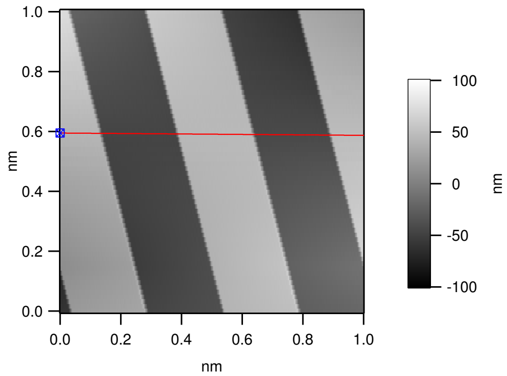

Figure 4.10: Height of the calibration

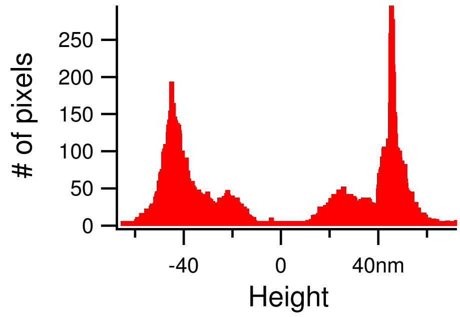

Figure 4.11: Histogram of the calibration

piezos was trying to reach features that are larger than his range. If we use the tilt correction, we see that our piezo has no problem reaching those features.

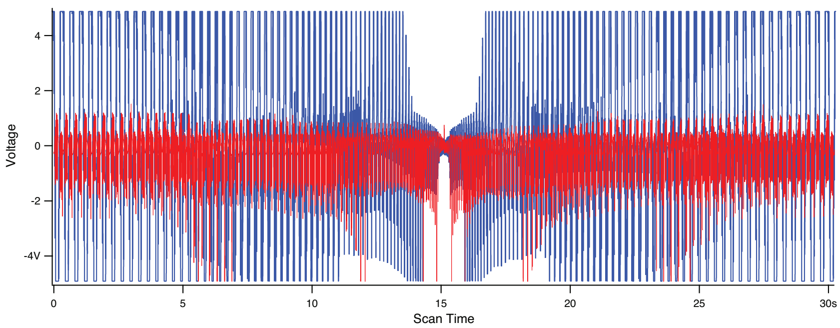

Figure 4.12: Output of the fast piezo

We can see the effect of the tilt correction on the following figure.

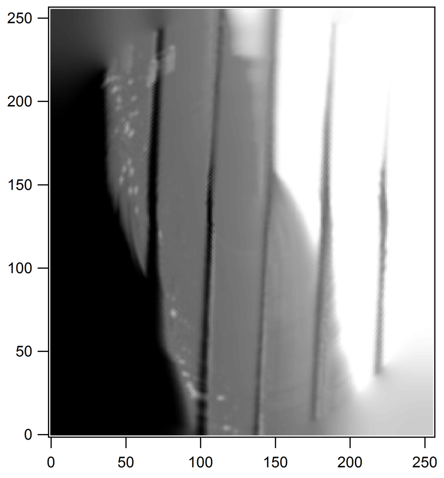

Figure 4.13: Before the tilt correction

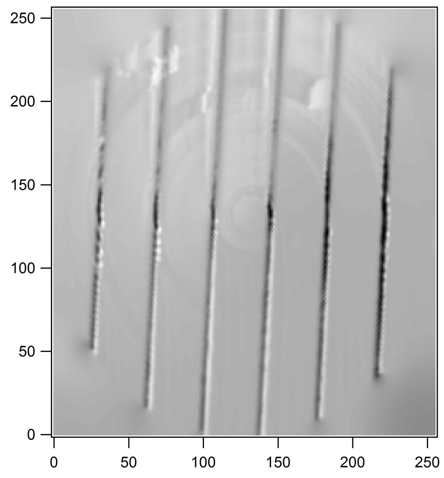

Figure 4.14: After the tilt correction

A Programming

A.1 Structure of the Spiral Scan program

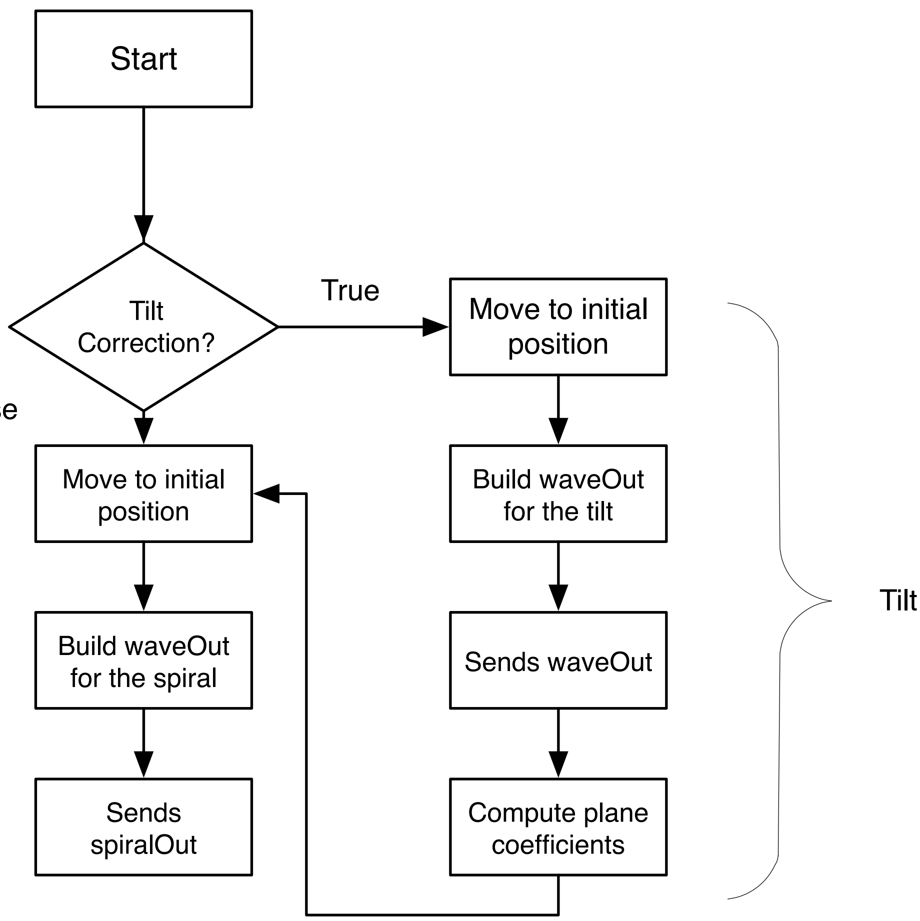

Figure A.1: Flowchart of the spiral scanning program

A.2 Other figures

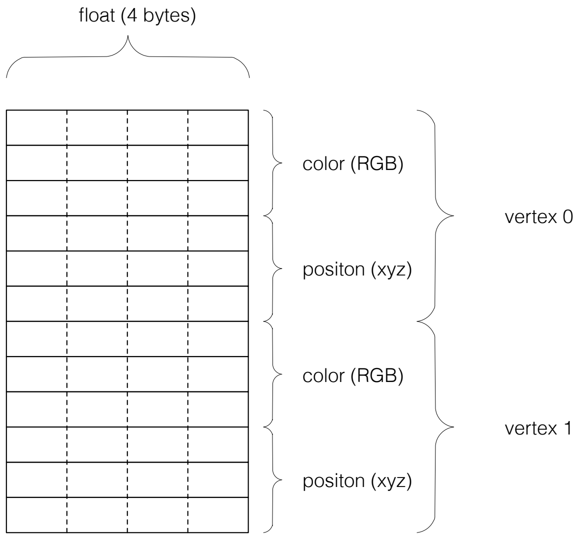

Figure A.2: Memory for the VAO

Bibliography

[1] Abobe. Tips for optimizing gpu rendering performance. 2013.

[2] Gilles Aubert and Pierre Kornprobst. *Mathematical problems in image processing: partial differential equations and the calculus of variations*, volume 147. Springer, 2006.

[3] Gerd Binnig, Calvin F Quate, and Ch Gerber. Atomic force microscope. *Physical review letters*, 56(9):930–933, 1986.

[4] PK Hansma, VB Elings, O Marti, CE Bracker, et al. Scanning tunneling microscopy and atomic force microscopy: application to biology and technology. *Science (New York, NY)*, 242(4876):209, 1988.

[5] PE Hillner, AJ Gratz, S Manne, and PK Hansma. Atomic-scale imaging of calcite growth and dissolution in real time. *Geology*, 20(4):359–362, 1992.

[6] Peng Huang and S.B. Andersson. Generating images from non-raster data in afm. In *American Control Conference (ACC), 2011*, pages 2246 –2251, 29 2011-july 1 2011.

[7] Younkoo Jeong, G. R. Jayanth, and Chia-Hsiang Menq. Control of tip-to-sample distance in atomic force microscopy: A dual-actuator tip-motion control scheme. *Review of Scientific Instruments*, 78(9):093706, 2007.

[8] Mark J. Kilgard. Modern opengl usage: Using vertex buffer objects well. Technical report, 2008.

[9] Yong Liang, Donald R Baer, James M McCoy, James E Amonette, and John P Lafemina. Dissolution kinetics at the calcite-water interface. *Geochimica et Cosmochimica Acta*, 60(23):4883–4887, 1996.

[10] IA Mahmood and SO Reza Moheimani. Fast spiral-scan atomic force microscopy. *Nanotechnology*, 20(36):365503, 2009.

[11] OpenGL, 2012.

[12] Manfred Radmacher. Measuring the elastic properties of biological samples with the afm. *Engineering in Medicine and Biology Magazine, IEEE*, 16(2):47–57, 1997.

[13] Manuel M Oliveira Brian Bowen Richard and McKenna Yu-Sung Chang. Fast digital image inpainting. In *Appeared in the Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain*, 2001.

[14] Jonathan Richard Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[15] T Sulchek, SC Minne, JD Adams, DA Fletcher, A Atalar, CF Quate, and DM Adderton. Dual integrated actuators for extended range high speed atomic force microscopy. *Applied physics letters*, 75(11):1637–1639, 1999.

[16] YK Yong, SOR Moheimani, and IR Petersen. High-speed cycloid-scan atomic force microscopy. *Nanotechnology*, 21(36):365503, 2010.