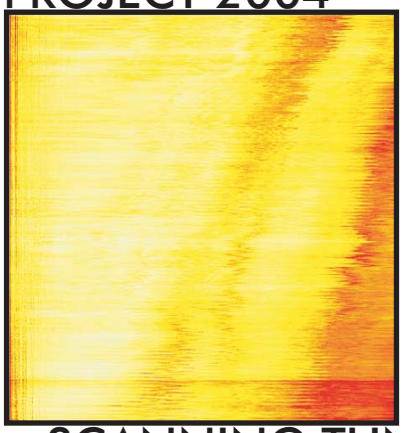
PHYSICS 312 PROJECT 2004



SCANNING TUNNELING MICROSCOPE

Jeffrey Klein
Erin Rooney
Kyle Slack
Michelle Peterson

Derin Sherman, advisor

A Scanning Tunneling Microscope (STM) is a simple and cost effective tool to see structures smaller than the wavelength of light, all the way down to the size of an atom. The following details our efforts to build our own STM.

HISTORY

Before the invention of the Scanning Tunneling Microscope (STM), there was an invention called a Topographiner. Russell Young (1) and his colleagues created the Topographiner (2) for the National Bureau of Standards back in 1971. This microscope used piezo drivers, much like the STM does, to control the tip that swept the surface of their specimen. A z-direction piezo controlled the height of the tip above the sample, always keeping the tip at a constant voltage compared to the specimen. An electron multiplier detected the tunneling current from the specimen to the tip.

Fig. 3: Binnig and Rohrer


Fig 1: Young

In 1981, Heinrich Rohrer and Gerd Karl Binnig, employees of IBM, took the Topographiner to a new level with their invention, the Scanning Tunneling Microscope (3). This new microscope also used the theory of tunneling electrons to scan a specimen and output an image of it at the atomic level (4). These were the first great images of surfaces at the atomic level. Rohrer and Binnig won the Nobel Prize in Physics for their STM in 1986. The Nobel committee said that the STM project opened up "entirely new fields…for the study of the structure of matter."

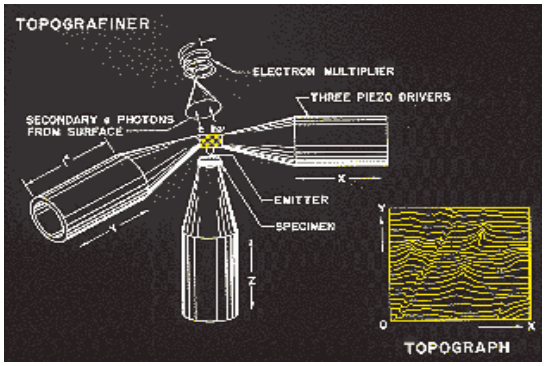


Fig. 2: The Topographiner

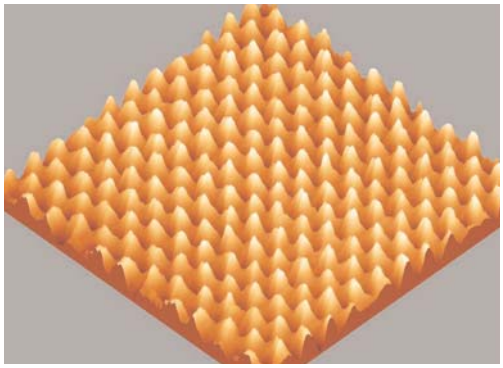


Fig 4: Atomic scan

USES

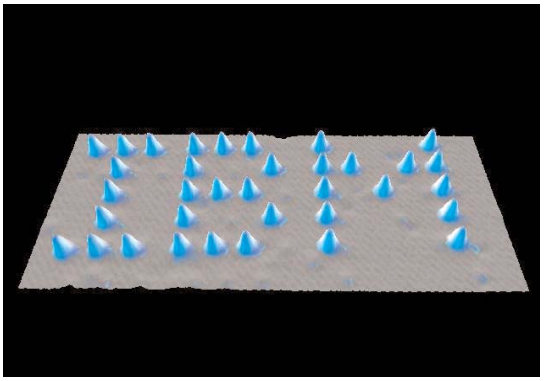


Fig. 5: IBM logo made with atoms

Scanning tunneling microscopes are used to obtain atomic-scale images of conductive surfaces. They are used to provide a three-dimensional profile of the surface, which shows roughness, defects, determines size, and confirms the existence of molecules on the surface of the substance. As the Nobel committee foresaw, though, STM's have been put to use in many different ways beyond just looking at the atomic surfaces of specimens. People have been able to manipulate the arrangements of individual atoms on surfaces, much as IBM did with there logo. They used an STM to arrange Xenon atoms on a Nickel surface to spell out IBM (5). Other uses for this technology include: applications in Nanotechnology, semiconductor physics, microelectronics, and even chemistry. A modified version of an STM can be used to look at DNA.

TUNNELING

An STM works off a concept known as tunneling. Tunneling occurs when an electron moves some distance into an insulated material. If the electron reaches a conductive material before it stops moving, the current, and therefore voltage will resume, albeit at a smaller value. If we measure this potential change, we can determine how far the electron traveled through insulated material. Using this information, we can potentially produce detailed images of conductive surfaces. Our microscope uses a circuit that keeps the distance the electrons travel constant. This requires the tip to be moved in accordance with any height differences in the surface. We do this by applying a small voltage change to our piezoelectric disc. By measuring this voltage change, we can tell by how much the tip has moved.

OUR STM

The plans for our STM came from a web site article called "Simple STM Project". It can be made in a surprisingly simple and cost effective manner. In consists of three portions, in essence: a mechanical componet that holds the sample and the tip, computer software that tells the microscope where to scan and reads, displays, and stores the result, and a electronic circuit that communicates between the two.

Upon receiving instructions from the user, the computer program gives instructions to the

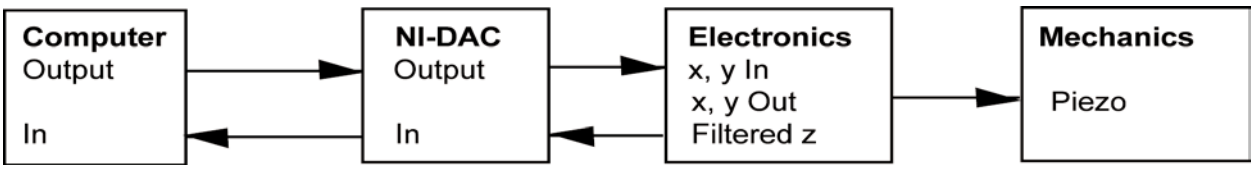


Fig. 6: An overview

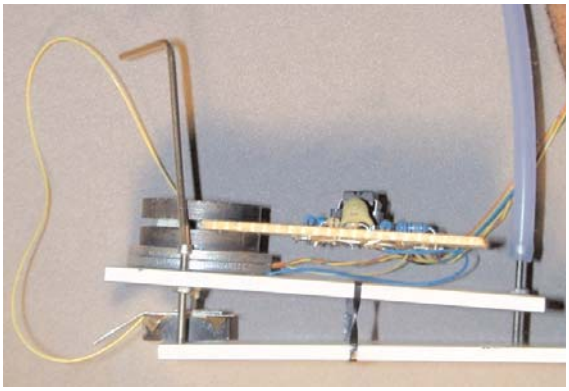
electronics to scan a rectangle, via an interface card, point by point. The electronics move the tip to each point, and monitor the tunneling current through it (or monitor the amount of voltage required to keep it at a constant height). With each point it scans, the software reads that voltage, which represents the height, back in from the electronics, and creates a plot of the data.

Thus, the total amout of significant pieces in the project are a piezo buzzer to control the tip, a tip etched out of copper (or ideally platinum/iridium), a holder for the tip and the sample, and an

interface card for the computer that can write and read voltages.

MECHANICS of an STM

The actual microscope part of an STM is physically made up of very few components. There are two stages, one that holds the sample and the other that holds the all important tip. There is also the tip and the piezoelectrics that control the movement of the tip (7 and 8).



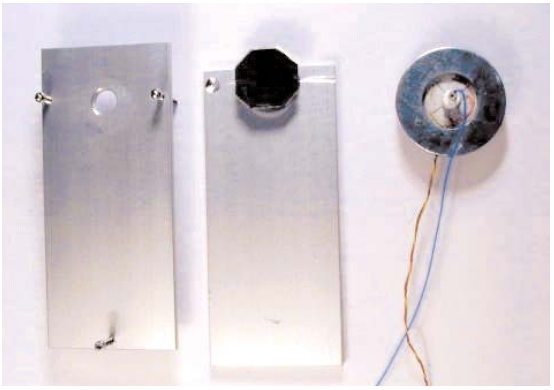


Fig. 7 and 8: A microscope like ours. The tip can be seen on the far right.

Sample Stage:

The sample stage consists of a plate of steel, a magnet, and a copper shim. The steel plate has grooves filed in to assist in the coarse adjustments made to the height of the tip. The magnet is used to equally distribute the bias voltage sent to the sample through the copper shim (the sample sits on top of the shim).

Scanner Holder Stage:

The tip holder stage includes three screws, two used for coarse height adjustments and one for fine height adjustments. It also has the piezo disk that controls the movement of the tip, and the tip itself.

Fig. 9 and 10: Piezo motion

A piezoelectric is a device that can be found in many everyday objects. Ours came from a telephone. A piezo is used because when a voltage is applied to it, it will warp and bend accordingly. When an alternating current is applied to the piezo, it will expand and contract rapidly, creating a buzz that you can hear (9 and 10). We took our piezo and divided it into four quadrants: x+, y+, x-, and y-. With these four separately controlled quadrants, we are able to move the tip (which is connected to the piezo) anywhere in the x-y plane. The piezo also has a z-component separate from the x-y components on the back of the disk that controls the height of the tip. Attached

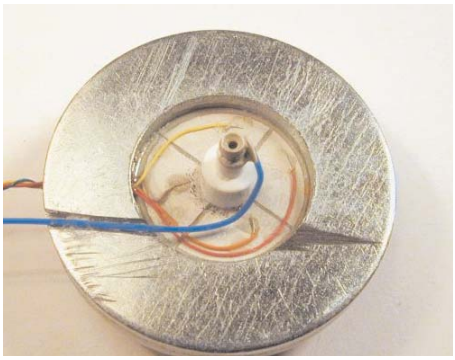


Fig. 11: Four-quadrant disk

to the middle of our piezo is the actual tip holder, a small IC socket. The tip can be inserted in the socket, where it will stick while we are working but is still easily changeable (11).

The tip is the most important part of the mechanics involved in an STM. The tip is the part that will actually move across (scan) the specimen's surface and allow the electrons to tunnel between the specimen and the STM (12). We need the tip to be atomically sharp in order to get quality images. The reason behind this is fairly simple. In order for the tunneling to work properly, the electrons would like this super sharp tip to travel to. If there is more than one tip at the end, tunneling may occur between more than just the one point creating ghost images making it hard to read the surface.

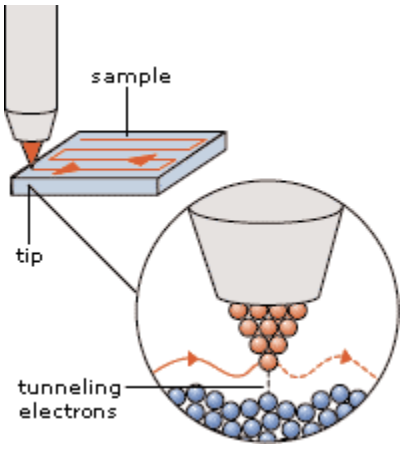
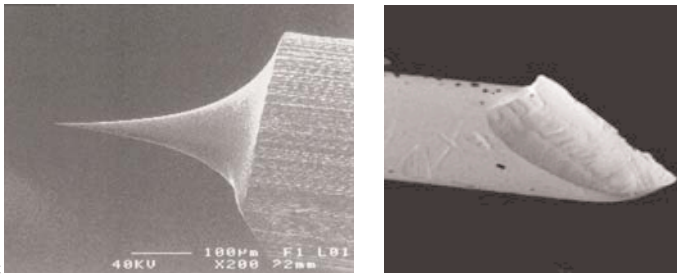


Fig. 12: Tunneling electrons

Creating atomically sharp tips that last is a difficult process. One can use tungsten, copper, platinum/iridium, or other conductive metals (preferably very stiff metals that won't be ruined easily). There are benefits and drawbacks to every kind of metal. Tungsten is very hard, but oxidizes quickly in air making the tip unusable. Platinum/Iridium wire does well in air, but is very expensive (we used plain platinum, which is softer than platinum/iridium, but easier to obtain). Copper seems to be a good starting place for tip construction since it is easy to obtain and can be used in air for much longer than tungsten, but it is not the ideal metal either.

There are two ways to make atomically sharp tips. What metal you use will help to make this decision. You can either cut or chemically etch a tip (13 and 14). With cutting, you slowly pull at the wire with a pair of wire cutters at a very sharp angle. If done correctly, the metal will eventually break and leave you with an atomically sharp point. The problem with cutting a tip is that it is a very "messy" way to do it. You end up with more than one point at the end of the tip and you just hope one is longer than the rest. While this method is quick, it does not make as reliable or as reproducible a tip as does etching. Etching a tip initially takes longer than cutting, but gives you higher quality, more reproducible tips. Etching a tip means that you take your tip piece (called an electrode) and place it in a chemical bath (chemical used is dependent on what type of metal is used in creating the tip) along with a counter electrode (also dependant on what type of tip you want to etch). You then apply a voltage to the electrode and run the counter electrode to ground. The current running through the chemical bath creates a reaction that will eat away, or etch, the metal tip electrode. This etching takes place at where the chemical bath's meniscus touches the electrode. As the tip gets thinner, the meniscus has less to hold on to and lowers. Eventually, the metal gets extremely thin. At this point, the part of the tip below the surface of the bath breaks off, leaving an atomically sharp point (15).


Figures 13 and 14: Etched and cut tips

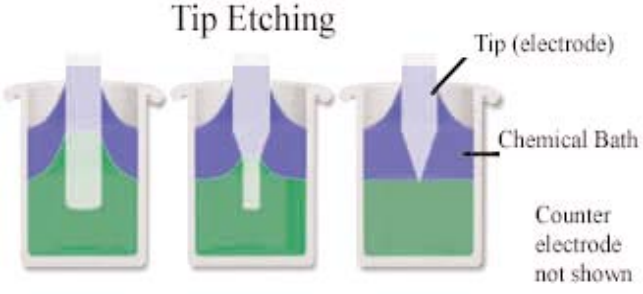


Fig. 15: Tip Etching

While etching, you want to stop the voltage as soon as the tip breaks. Letting the voltage run longer than this results in a dull tip. In order to prevent this, a circuit can be built to shut the voltage to the tip off as soon as it breaks. When the tip breaks, a large resistance is introduced to the circuit. A good circuit will read when this occurs and shut off the voltage running to the tip immediately. We built two circuits, one for use with a DC voltage, the other for an AC voltage. Which one to use depends on the type of tip being made. Copper tips respond well to DC, while platinum tips need the extra oomph that AC can provide. Both circuits have a switch to push to start the voltage running and the etching going and a way to sense when the tip breaks. We made many quality copper tips using the DC circuit. The AC circuit works properly, but we have not yet found the correct counter electrode and chemical bath to etch our platinum tips.

Future Developments:

If we had more time, we would like to explore a few more avenues in our STM. We would like to be able to find the correct counter electrode and chemical bath to create platinum tips. We would like to use a ceramic magnet instead of the metal magnet used in the sample stage. More high-tech STM's use a computer controller to finely adjust the tip's height to a proper level. It would also be nice to make the tip controls out of transducers, much like the Japanese team did for their STM. Transducers are more accurate than the piezo is, which would probably result in better scans.

ELECTRONICS of an STM

The electronics component is made up of a few different stages (see Figure 16, next page). The first is the power supply. This runs off of two 9-volt batteries and produces a voltage from negative to positive nine volts. The next produces the sample bias voltage. It provides the sample with potential so that the electrons will flow across the gap between the sample and the tip. Another part of the circuit is two smaller identical circuits meant to control the movement of the tip. They apply equal and opposite voltages to diametrically opposed quadrants of the piezoelectric disc, warping it and thus causing the tip to move. In this manner, the overall voltage is kept at zero, allowing for movement in the vertical Z-direction. The final part of the circuit does just that. It tries to keep the tunneling current constant and sends a signal to the piezoelectric disc, warping it in the Z-direction. Measuring this signal, we can tell how far we are moving the disc vertically.

SOFTWARE of an STM

The purpose of the software used in this project is twofold: first, to communicate with a hardware controller card that can read and write voltages to transmit commands to the electronic controller circuit (and thus, to the microscope itself), and second, to read back data from the circuit and display it on the screen in a meaningful way. From the point of view of the computer, the rest of the microscope appears to be a mathematical function; that is, the computer gives it an (x, y) value, and a z height value is returned. It is then up to the software to manipulate, visualize, and store this data.

Display:

Before continuing, it is worth explaining a couple of computer science concepts. An API (Application Programming Interface) is a set of tools used by a programmer to bypass lower forms of programming. An API provides abstraction; that is, it allows a programmer to ignore the nuts and bolts of how the computer actually works and focus on what he is doing. Just as Microsoft has

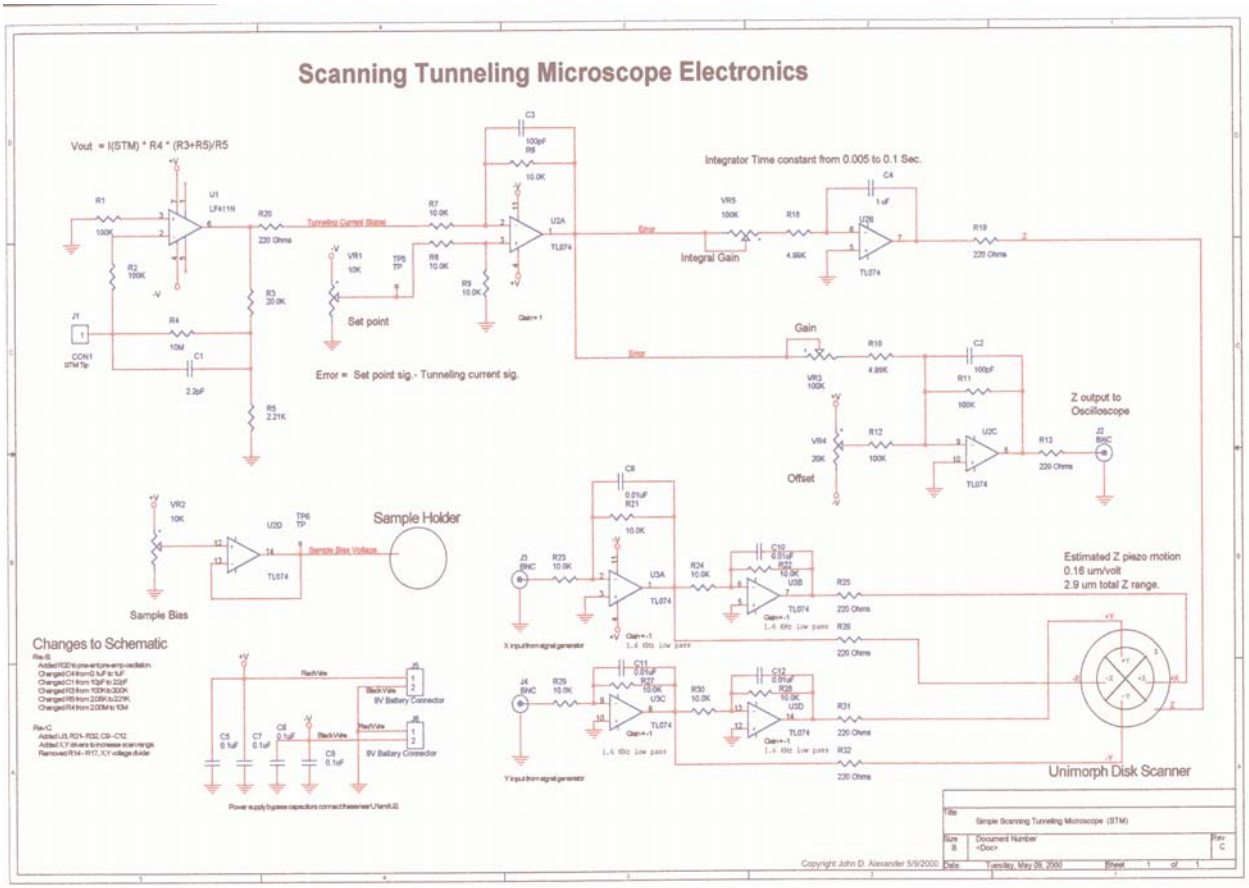


Fig. 16: STM electronic circuit

an API to help people program Windows applications, APIs exist to draw on the screen two or three-dimensional images.

It is also worth making a distinction about several methods of drawing. Ultimately, of course, everything you draw on a computer will end up on the flat monitor as a two-dimensional image. However, there are several options on how to accomplish this. The computer can simply draw a two-dimensional, flat image, that is coded by color to represent different heights. The computer can render a 3d-looking graph as two dimensions, then display that. Or, the computer's video card can keep track of all of the points in three dimensions, then make the conversion itself right before the image is displayed (this "true 3d" mode is the basis of most modern video games).

Choosing the proper APIs was a surprisingly large part of the programming portion of the project. I was initially seduced by the display possibilities of a "true 3d" API such as OpenGL (commonly known for being used in the first true 3d video game, Quake, and still in common use today). By rendering an image in three dimensions, it could easily rotate and scale it in real time. However, OpenGL proved too slow with our equipment.

APIs exist purely for graphing, and will create a two dimensional picture of a three dimensional graph simply by being handed the data. Unfortunately, they are largely not free, and if they are, exist for the wrong platforms, wrong languages, or simply have not been properly maintained for years. Trying without success to make one of these APIs work led to several days of frustration and wasted time.

The visualization was eventually done with the basic Windows / Microsoft 2d drawing API known as GDI+. I was able to use it to efficiently draw colored 2d height maps on the screen. With some more effort, it could be used to draw 3d graphs.

Data Collection and Storage:
Upon receiving a request to do a scan, the software transmits the points to a rectangle to the electronics, one at a time, each time waiting to receive the height back. These data points are stored in a 3d (essentially 2d with a couple of extra cells for additional data, should it be needed in the future) array in memory. The program uses this data to write a standard Windows bitmap, just a long string of colors (which are chosen by the heights), and then uses GDI+ to display this bitmap. The program also has its own proprietary format that it can read and write, to save and open files.

GUI, Features, and Future Improvement:

The STM software is made up of one window, used to display scans, and dialog boxes for the user to communicate requests to the software. Keyboard commands are used to move and zoom the image. Dialog boxes include open, save as, and export; as well as scan. This has proven a simple and effective interface.

In general, the program could certainly be made more robust and user friendly. A 3d display mode is the most obvious improvement that could be made.

FINAL RESULTS

While each individual part of the project came together with a fair amount of ease, actually getting everything to work in conjuction and work properly proved to be quite difficult. After

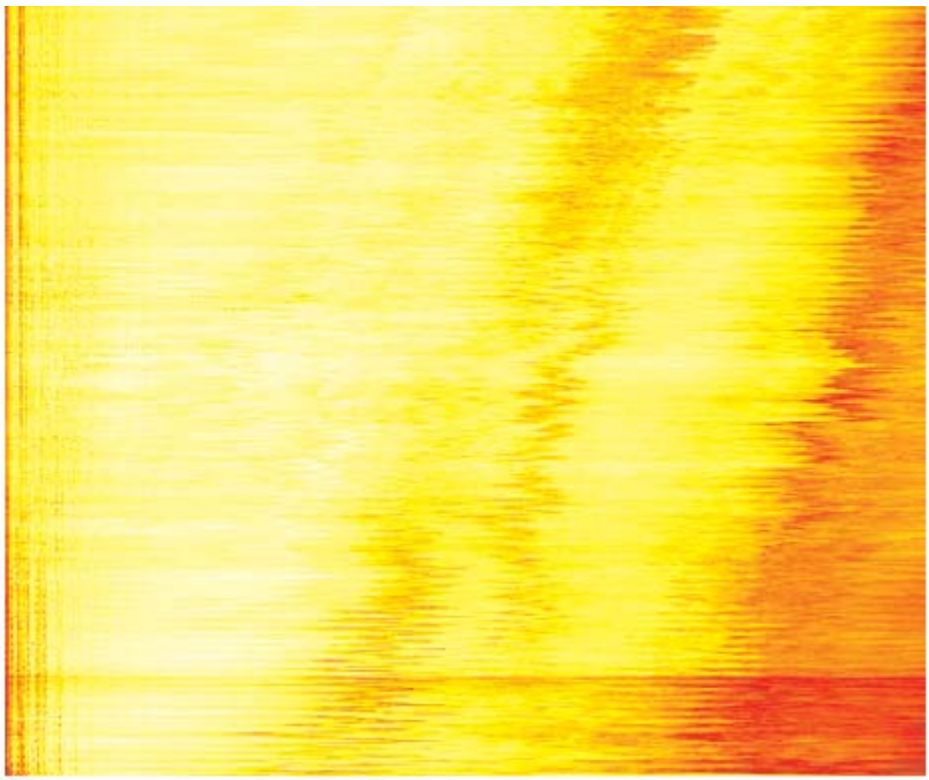


Fig. 17: Ridges in graphite

doing some debugging, and very carefully fine tuning the distance of the tip, we managed to obtain a few sub-micron pictures of graphite, the best of which is shown in Figure 17. A rough pattern, mostly some ridges, (along with noise and lines from the tip bouncing) can be observed. We did not have time to use the proper materials to achieve single atom resolution.

WORKS CITED

Background of STM's page,
http://www.nobel.se/physics/educational/microscopes/scanning/

Chapman, Stephen J. MATLAB Programming for Engineers. Pacific Grove, CA: Brooks/Cole, 2001.

Deitel, H. M. and Deitel, P. J. C++ How to Program (4th Edition). Upper Saddle River, NJ: Prentice Hall, 2003

Hawkins, Kevin and Astle, Dave. OpenGL Programming. Roseville, CA: Premier Press, 2001.

IBM's page on STM's,
http://domino.research.ibm.com/Comm/bios.nsf/7de2da47216ee36985256ad5005
7ce42/9343fa24fb21b85085256ae100554f3d?OpenDocument

Foley, van Dam, Feiner, Hughes. Computer Graphics, Principles and Practice (2nd Edition). Addison-Wesley, 1990

Information about the Topographiner page,
http://physics.nist.gov/GenInt/STM/topograf.html

Libioulle, L., Houbion, Y., and Gilles, J.-M. "Very sharp platinum tips for scanning tunneling microscopy". *Review of Scientific Instruments.* Vol 66, January 1995.

MacLeod, Jennifer M. "A beetle-type scanning tunneling microscope for studies of nano-structured surfaces". Queen's University, Kingston, Ontario Canada, November, 2001

MSDN: Visual C++, GDI+, Windows API. http://msdn.microsoft.com

NI-DAQ Function Reference Manual for PC Compatibles. Austin, TX: National Insturments 1993.

NI-DAQ Lab-PC+ User Manual. Austin, TX: National Insturments 1993.

Neilson, Linn. Windows API and OpenGL.

Sherman, Derin. In class discussions, comments, and all around help.

STM project page, http://www.geocities.com/spm_stm/Project.html

theForger's Win32 API Tutorial v 2.0, http://www.winprog.org/tutorial/index.html

Wildenberg, Andrew. OpenGL and advice.