An Approach to Processing Noisy 3D Trace Data from Nuclear Particle Collisions

Oliver Song

Advisors: Dr. C.W. Christoe, HTHS; Dr. Knies, NRL

$E=mc^2$submission

i. Personal Section

My interest in the subject began at the end of my junior year. I had previously been heavily involved in computer tech and "entry level" activities like blogging and RSS feed reading, and I knew that my future was with electrical engineering and computer science. At the time I didn't know any computer science. I had also had very unfulfilling experiences with research before. I knew I wanted to conduct some research in senior year, so I was looking around, asking my counselors and teachers for help. One of my closest teachers, Dr. Christoe, had a friend working at the Naval Research Lab who needed help with some computer science work and was willing to take on a student-intern. I figured it would be a perfect two-birds-one-stone situation to learn programming and do a high-level computer science project.

Of course this was a lot more difficult than it sounds, efficient as is it is. Dr. Knies, the particle physicist I was working with, worked in the department pertaining to Cold Fusion. I went down to Maryland to talk to him face to face. He needed help in some key places in his "CR-39" detection project, and we managed to isolate a very specific need, the analysis and cataloguing of massive matrices of data. I jumped on the project. The project, although I didn't understand it at the time, was heavily involved in image analysis. I spent a day considering how to even go about the thing, considering what a good entry level programming level was. After considering python, java, C, and many others, I settled on MATLAB because of its built in image analysis package. My dad knew a bit of MATLAB, so I was lucky he could answer my questions throughout learning the language.

Throughout the rest of the summer and into the school year, I worked at home learning MATLAB and chipping away at the project. It turned out to be a massive, multistage (10 stages in total) problem-solving labyrinth, so there were many times I was incredibly frustrated and

exhausted. Even if I knew the thought process and the function types I wanted to use, I didn't know which ones were specific to MATLAB, and I didn't know how to find them. On top of that, syntax errors plagued the project like a virus. On top of learning the MATLAB language, I had to learn new mathematics to tackle the problems as well. The data given was a massive 3-D matrix, so I had to learn advanced Calculus concepts such as gradients and spatial convolution. Additionally, I had to find noise reduction filters for the data, and in two cases create my own methods for noise reduction.

For me, this was as far off the beaten path as I could have gotten. With insufficient skill in computer science and mathematics going into this project on top of the requirement to develop new methods of image analysis, it was a daunting task. I've come out with a strong knowledge of computer science and advanced calculus, but not without some serious stumbles along the way. For example, after having studied the MATLAB software for quite some time, I found myself at a serious block in how to immerse myself in the actual problem solving of the situation. Analyzing massive 3D data sets for minute details is not the same as those beginner tutorials you find online, after all. I found myself at a loss as for inspiration and direction. When I went to my advisor Dr. Christoe, although he couldn't help with the computer science itself, he sat me down and looked me in the eye and told me, "You can't quit now. Get it done." Those words really stuck with me, blunt as they were, and I really crunched it out.

Besides the sheer material I flew through in the project, I think I learned a lot about greater things and earned a severe insight on science and mathematics. The project showed me the power and flexibility of Computer Science, and the sheer applicability of Mathematics. Later in the year in Multivariable Calculus I learned the content I had speed-studied before—

directional derivatives and convolutions and whatnot—and although we sped through it just as well, I saw on a much grander scale of what Math was capable of.

If I could leave future students undertaking science and mathematics projects some advice, I would tell them to persevere. When you're closest to giving up, you're probably actually closest to the breakthrough. Don't be afraid to ask for help, because sometimes people know exactly what you need to know. Follow what you truly want to do, and research turns out not to be so difficult after all; in fact, try to make research as relevant to your life as possible to keep things interesting. Never give up on your goals! Don't cut corners, because the "scenic route" is worth it in the end. Finally, get sleep. Health is more important than getting ahead a tiny increment.

ii. Research Section

Cold Fusion has been an active research field in the quest for next-generation energy. In Andrei Lipson's CR-39 experiments, oscillating deuterium atoms or other particles were accelerated (collective acceleration effect) through an electric field and collided with each other to undergo fusion. Another procedure conducted by Roussetski involved the bombardment of TiD_2 with a Deuteron beam.

In all these scenarios of fusion research [1][7][9], a significant bottleneck is the detection of reactant molecules. The application of CR-39 plastic track detectors in cold fusion experiments is vital to detecting and identifying different particles and background/foreground separation.The current method of gathering data from CR-39 tracks is to use an electron microscope to dissect each individual crater in the x-y and z planes. There has been no way to analyze large amounts of CR-39 data in a reasonable time frame.

In this research, we study 3D trace data from nuclear particle impacts upon CR-39 detectors to identify craters made by particles. We utilize a new process, confocal microscopy, to gather numerical trace data from the polycarbonate. We propose and apply new approaches for detecting and computing several main characteristics, such as depth and incident angle, of the impact of the particle. Our approach and related code serves as a tool for automatically classifying the craters and matching them to known collision types and corresponding particles, therefore enabling the efficient and accurate processing of large quantities of CR-39 data.

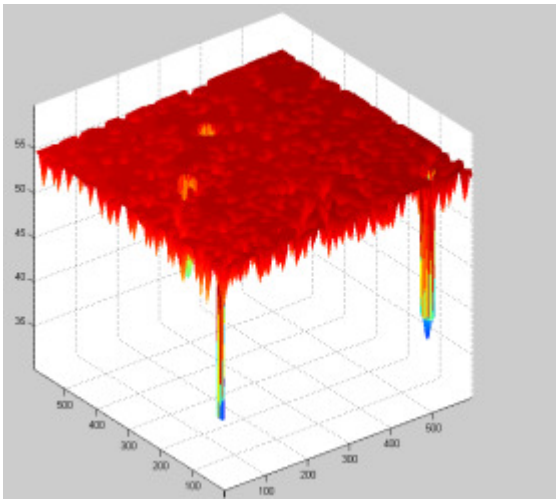


Figure 1(A)—3D plot of height data

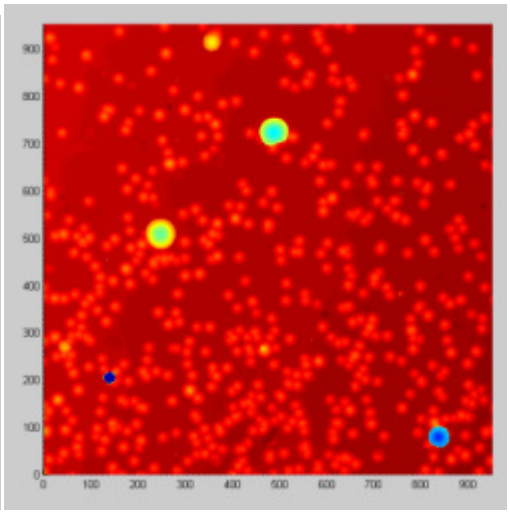


Figure 1(B)—Top view of height data

The first step in our research is to identify the craters. Since the data is very noisy, we need to filter out measurement noise and abnormal crater data (random bumps and holes in the substrate that couldn't be particle collisions). In this step, we first find the gradient of the data set using the Sobel operator, a noise resistant convolution kernel that generates gradient estimates.

$$Dy = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathrm{Z} \qquad \text{and} \qquad Dx = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathrm{Z}$$

where * here denotes the 2-dimensional convolution operation:

$$c(n_1, n_2) = \sum_{k_1 = -\infty}^{\infty} \sum_{k_2 = -\infty}^{\infty} a(k_1, k_2)\, b(n_1 - k_1, n_2 - k_2)$$

The *x*-coordinate is here defined as increasing in the "right" direction, and the *y*-coordinate is defined as increasing in the "down" direction. At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using: $R = \sqrt{Dx^2 + Dy^2}$

Using this information, we can also calculate the gradient's direction:Theta = arctan(Dy/Dx).

€

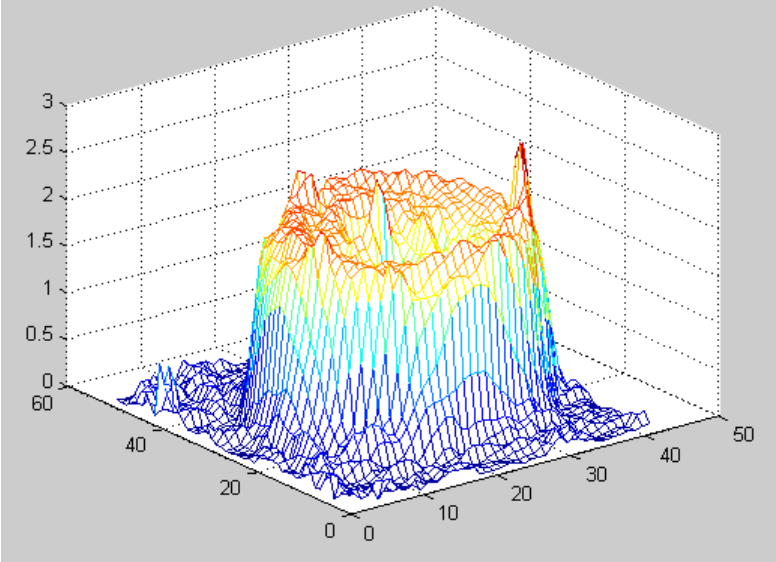


Figure 2(A)—Isolated crater gradient

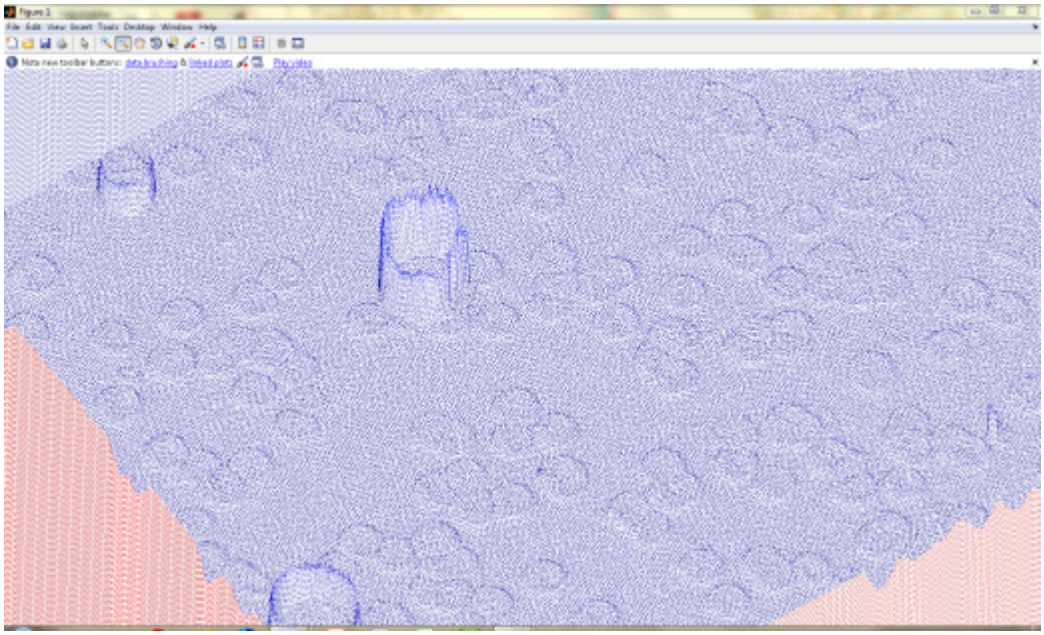


Figure 2(B)—Total gradient results from the Sobel Method

We can clearly see all the craters in the gradient map. However, to further identify the crater, we need to transform this data set to a binary set.

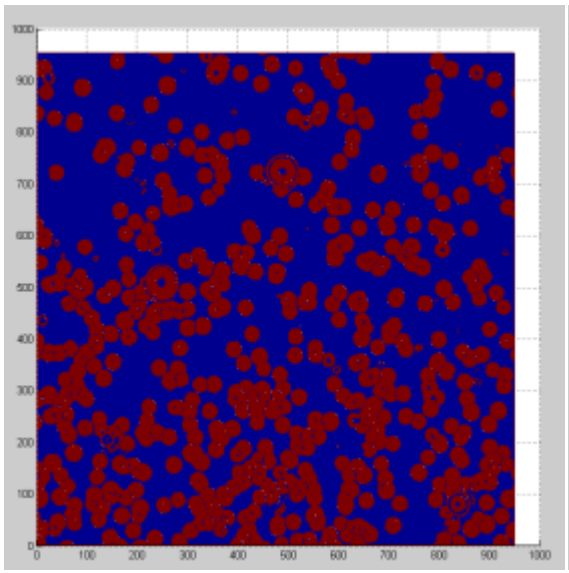


Figure 3(A)—Binary Gradient Matrix

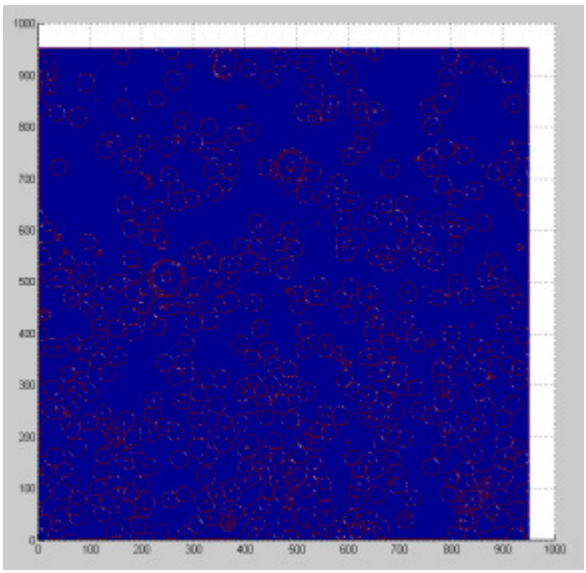


Figure 3(B)—Edge of Gradient Matrix

Then, we cut off other noises by discarding those points with gradient values less than a certain threshold. In our test, we used 0.5 as the threshold based on our examination of the gradient matrix R. This algorithm generates a binary data set that represents the locations of the circular craters. We then further process the data set using the edge() detection method to extract all the circles corresponding to each crater. We use these circles later to identify the center and diameter of the impact.

(1) Before the mean and standard deviation of data matrix Z can be computed, the matrix Z must be converted into vectors as functions mean and std are for vectors.

```
X=Z(:)
mean1 = mean(X)
std1 = std(X)
```

(2) All data points outside of a certain number of standard deviations are removed to cut off crater data. For this algorithm 1 standard deviation was used.

```
B=X(abs(X-mean1)<std1);
```

(3) After this we find the mean and standard deviation of B to identify our substrate height.

```
meanB=mean(B)
```

stdB = std(B)

It is assumed that the substrate level is within 1 standard deviation of the true mean—this can be adjusted later with more data to compare to.

Numerical Results:

The mean of vector v=z(:) was calculated to be 54.7070.

The standard deviation of vector v=z(:) was calculated to be 1.4481.

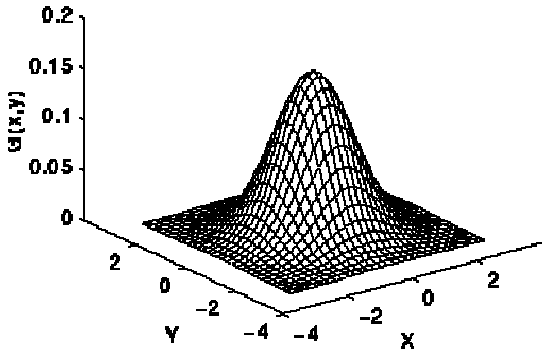
After the exclusion algorithm to cut points beyond a standard deviation out, the calculated mean of remaining data points was 55.0381. *Thus the substrate base height was found to be 55.0381.*

The second step is to identify key characteristics of each crater. Again, since the data is noisy, special considerations must be taken to remove the abnormal points. We study the Gaussian filter for noise reduction for this purpose. However, the Gaussian is a low pass filter that is effective against high frequency random noise but not effective against low frequency system noise like crater imperfection.

Gaussian filtering uses an isotropic form:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Visualized as:



$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

To remove both random noise and system noise, we propose a new threshold cut off method. This method proves to be very effective in removing both random and system noise.

```
cutoffratio=0.6; zc=zeros(sz);
fori=1:sz(1)
        for j=1:sz(2)
                if z(i,j)>meanB-stdB*cutoffratiozc(i,j)=meanB;
                elsezc(i,j)=z(i,j);end
        end
end
```

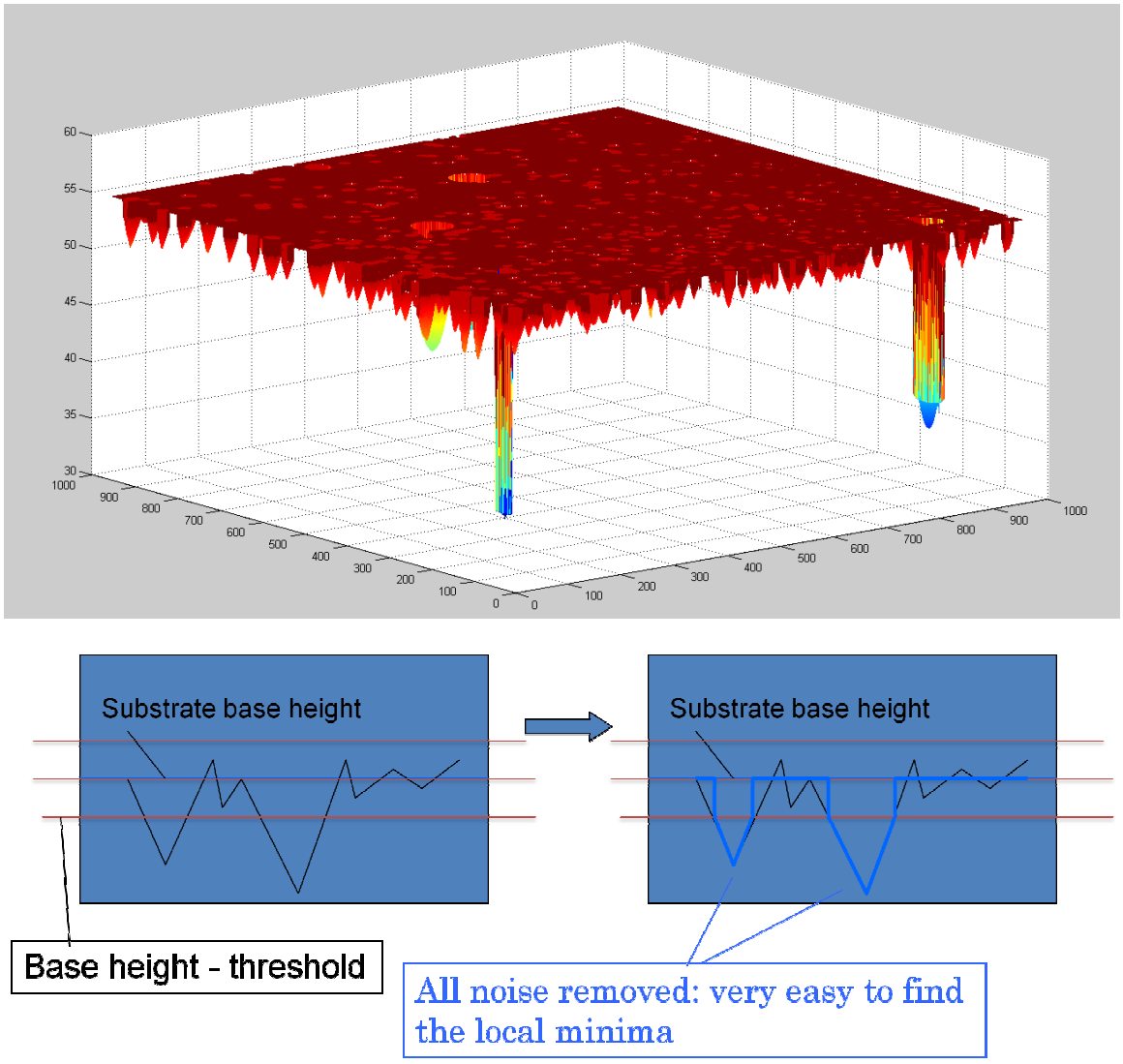


Figure 5—Custom filter "cutoff" method

Based on the noise free cut-off data set, we then find the regional minima using the Matlab function imregionalmin(). We successfully found local minima and their related heights and depths using this method. The final data table is omitted due to page limitations. Figure 6 illustrates the relative depth of each center point found in the center point matrix.

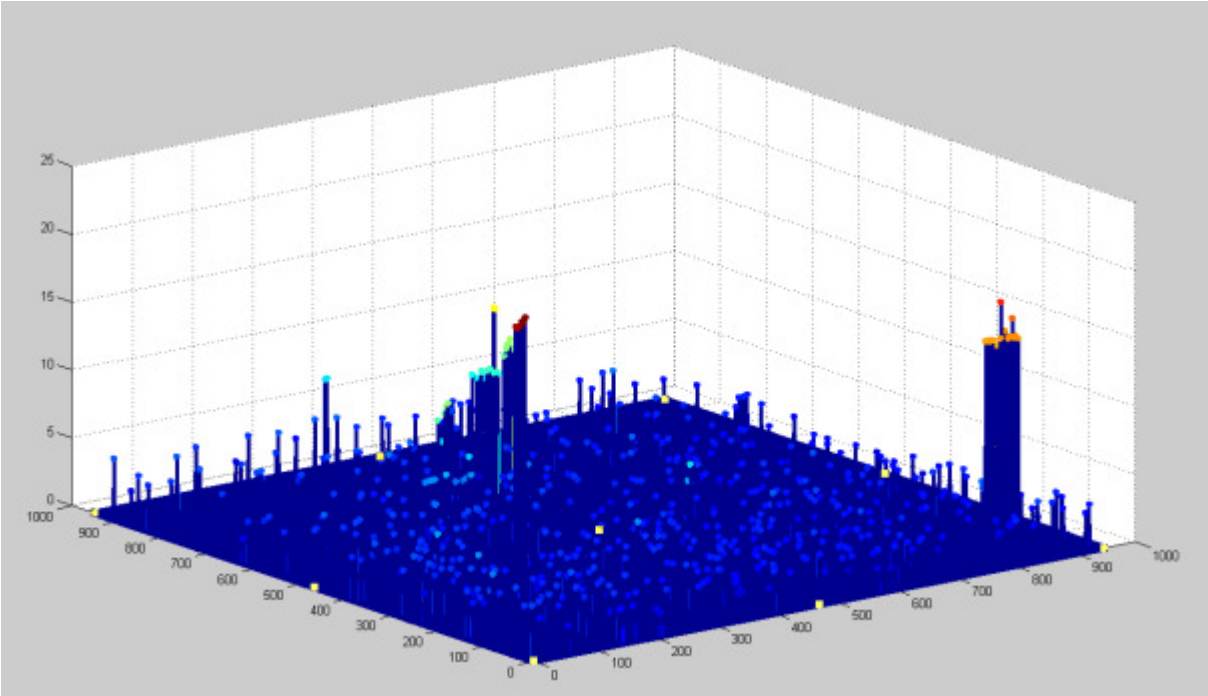


Figure 6—3D Plot of depths in corresponding crater loci

Next, the Hough Transform is used to identify the center of the surface impact circle. However the regular Hough Transform method cannot detect hundreds of imperfect circles. We propose and have implemented a "divide and conquer" Hough Transform method. This method loops through all impact points via relative minima, each time only applying Hough Transform to a small surrounding area. Our results show that *all* circles within the local region were detected by this method. Furthermore, we used an accumulator method similar to Hough Transform to detect the radius of the circle and compute the cone angle of each crater.

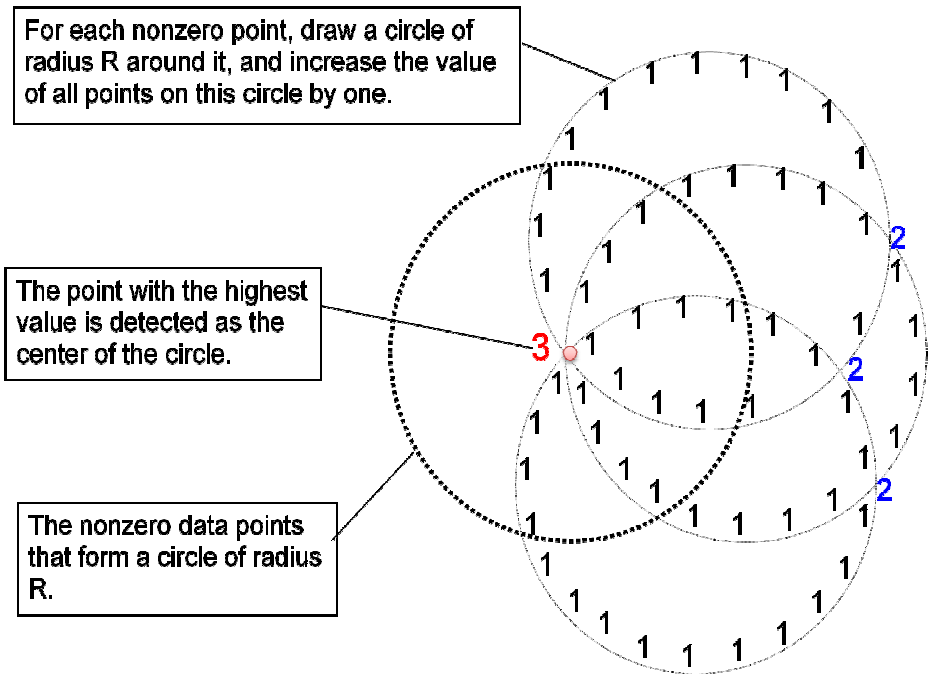


Figure 7—Depication of the Hough Transform algorithm

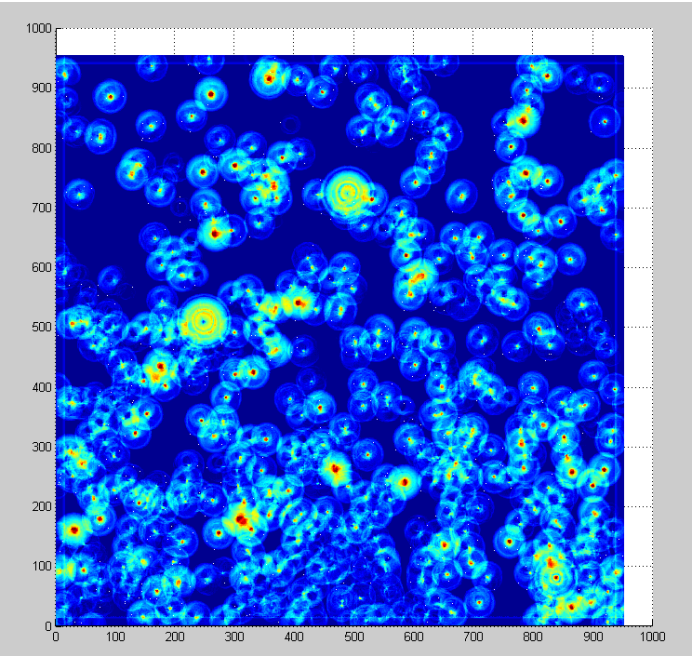


Figure 8—Result of the Hough Transform applied to the whole data set

It is also important to note that we convolve a 17 x 17 “Mexican Hat” matrix with each accumulator to concentrate the highest brightness in the center of gravity. Since the circle we are

detecting is not a perfect one, the algorithm may find multiple points in the resultant matrix. Thus, we further filter and process these points to find the point with maximal strength—this point is considered as the center of the circle.

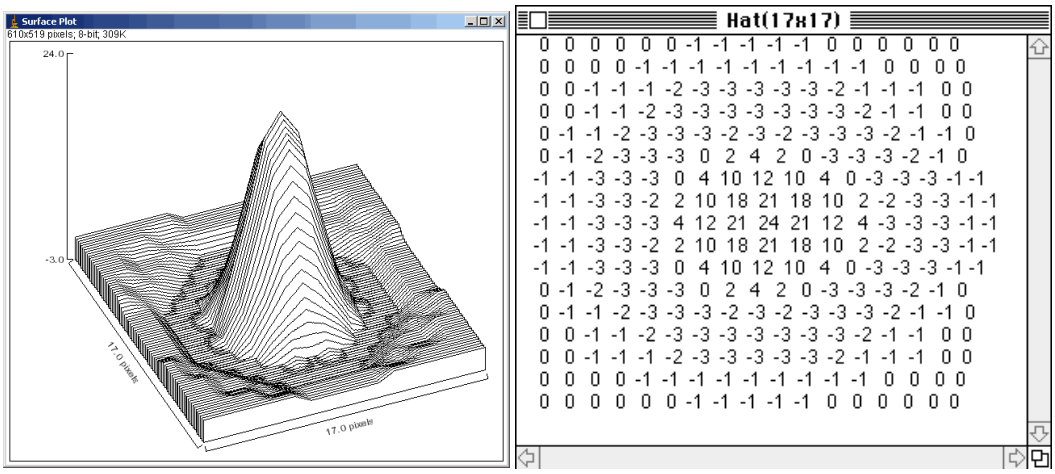


Figure 9—Mexican Hat kernal

Figure 10 shows our method for calculating the incident angle. "h" is the depth of the local minimum, "θ" is the incident angle, and "d" is the distance between the center of impact detected by Houghcircle and the local minimum projected onto the surface.

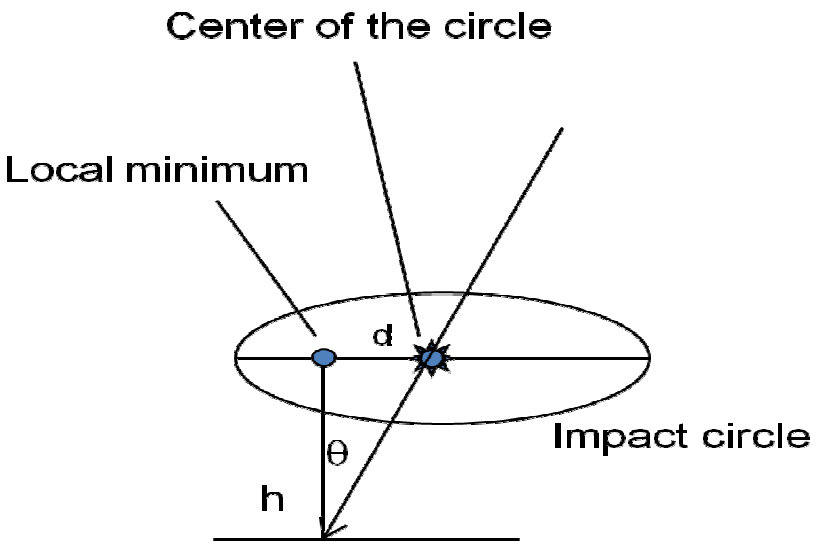


Figure 10—Diagram for trigonometrically calculating incident angle theta.

Our diameter algorithm utilizes a custom version of the Hough Transform. Based on the binary edge matrix B2, the algorithm sweeps through all nonzero points. For each point (x, y), it computes d = $\sqrt{(x-a)^2+(y-b)^2}$; then increases the accumulator acc(i)if | d–r(i) | < threshold.We scale the accumulator by dividing each value by its respective r. We then find the radius by identifying the r(k) corresponding to the maximum acc(k). This method is shown in Fig. 11.

After finding the radius, we find cone angle through a simple trigonometric calculation:

θ= ATAN(radius/depth)

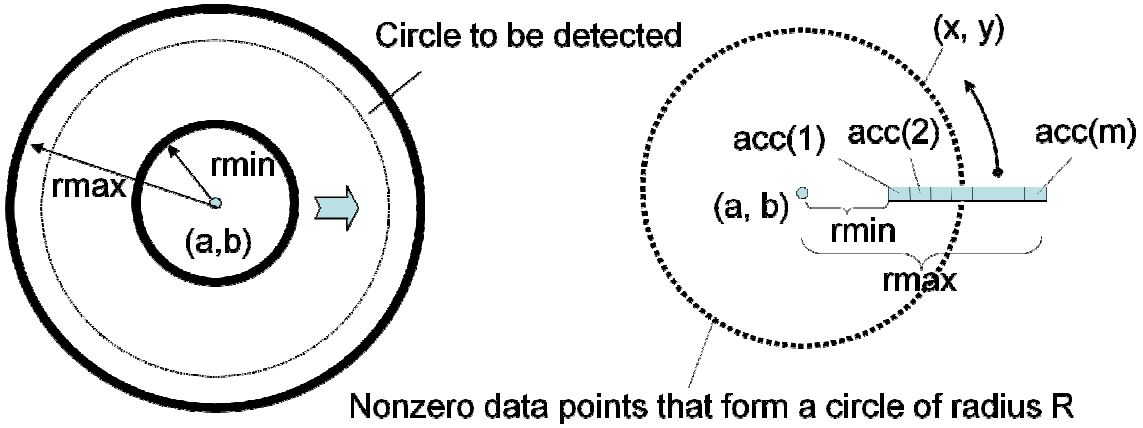


Figure 11(A) and 11(B)—Diameter method

After computing the main characteristics of the craters, we plotted the results to analyze the distribution of depth, cone angle, diameter, and incident angle.

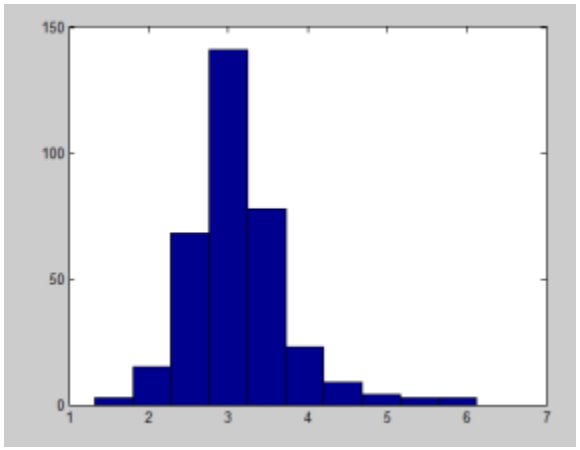


Figure 12(A)—Depth

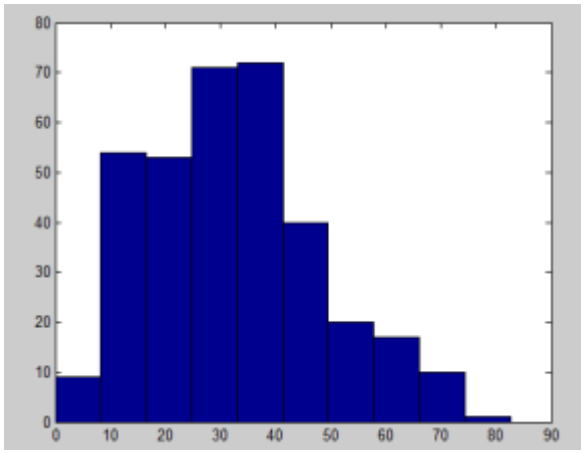


Figure 12(B)—Incident Angle

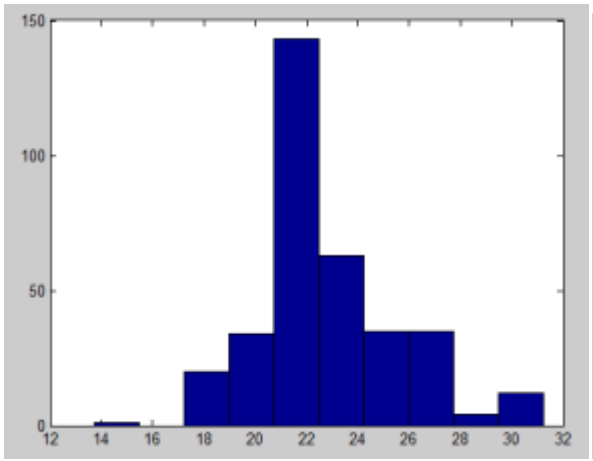


Figure 12(C)—Diameter

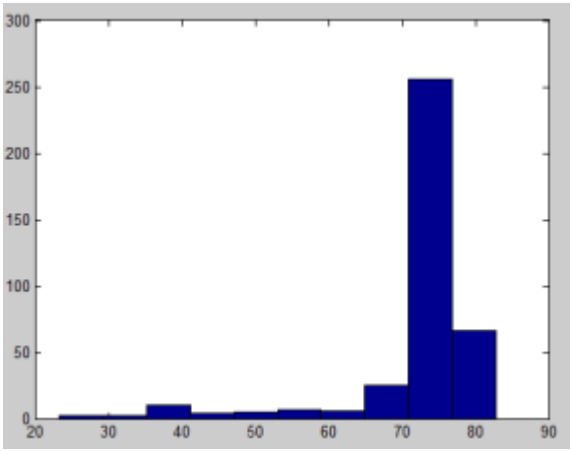


Figure 12(D)—Cone Angle

In conclusion, we used the 3D data from the observed nuclear particle collisions to identify and locate craters. The data set contained both random and system noise and one of our main goals was to filter out this noise. The Sobel method for calculating the gradient proved to be effective, and the binary matrix extracted from the gradient data matched the location of the craters in the original data set. Our new "cut-off" method was able to remove both random and system noise in the original data set. This made it possible to estimate regional minima and depths, thus finding the impact points of the particles.

Our "divide and conquer" Hough Transform Method was highly effective in detecting hundreds of circles and determining the centers of the impact circles. Our radius detection

method was also effective in finding the radiuses of the circles. Based on those parameters, we then calculated the incident angle and cone angle by trigonometry. Our approach has proved to be quite effective, and suitable for automatically processing large amounts of data.

The overall fitting of the detected points is depicted in Figure 12(G). The algorithm successfully detected all 524 craters in the sample space. It can be seen that there are still some false detections made by the algorithm. For future research, we will continue to optimize the algorithm to improve the detection accuracy. Furthermore, we need to study the method for automatically removing outliers and fixing imperfect or badly shaped impact points.

Our methodology is represented in graphical format in Figure 13. A typical data point output from the method is depicted in Figure 14.

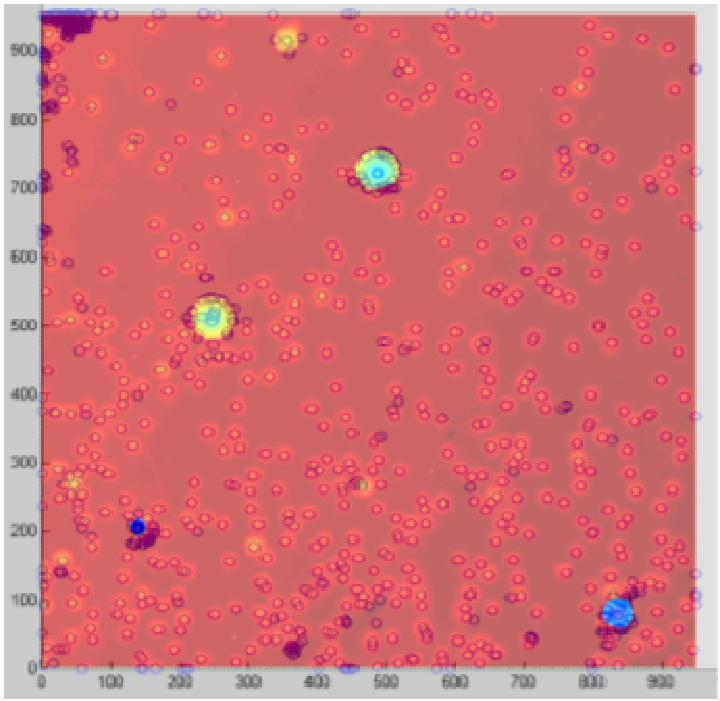


Figure 12(G)—Overlay of the Detected Impact Points on the Original Data

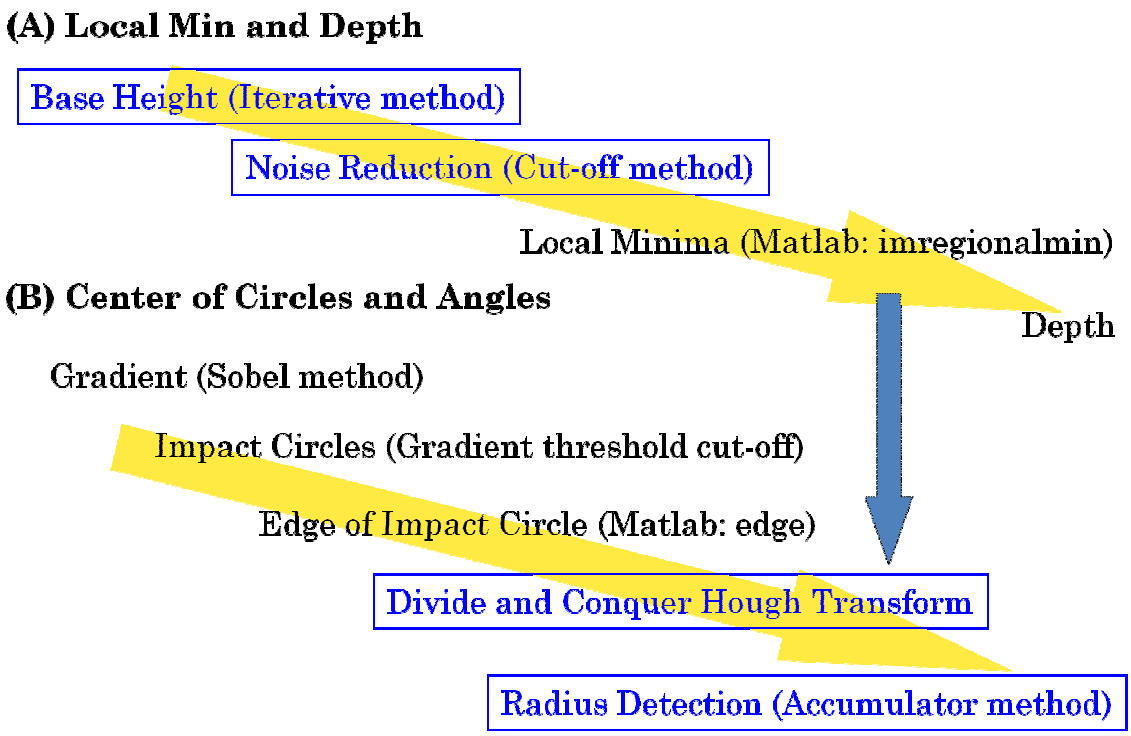


Figure 13 – Graphical overview of our methods

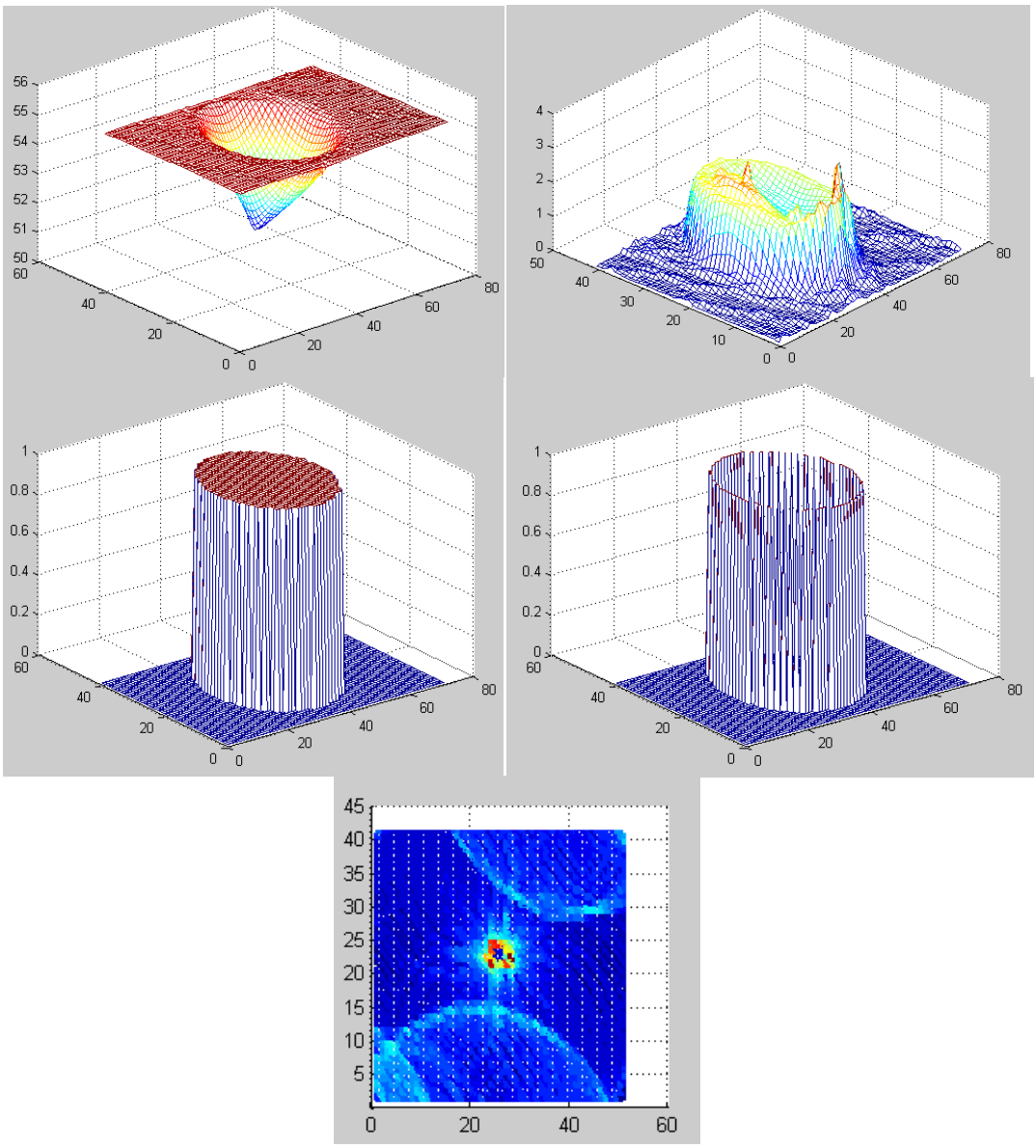


Figure 14—A sample data point; Centr: (381,285); Dia: 20µ; Cone Angle: 72.43°; Inc: Angle: 41.44°; Dep: 3.17µ