Image Analysis of Granular Materials

Natasha Flowers
(Department of Physics, University of California, Davis)
(Dated: September 10, 2015)

This project seeks to examine the effects of shape, concentration, and placement of individual particles within a granular material on the angle of stability by analyzing large numbers of images taken just before an avalanching occurs. To do so, it is necessary to fully automate the identification of the location of different shapes within the material. This code was originally developed in IDL and was working at satisfactory levels of accuracy in 2009, but has since been translated to Python and seen an unacceptably large decrease in accuracy. Several bug fixes and some original modifications to the code were completed. The code employs both heuristic methods and neural networks to identify shapes, and the process of retraining the neural networks to improve accuracy was also begun. Although significant improvements were made, the code is still functioning below peak levels and there are suggested further corrections to be made.

I. INTRODUCTION

Granular materials are simple in definition and common to our everyday lives, but they exhibit interesting behavior that is very difficult to fully explain or predict. Granular materials are defined as a collection of discrete, macroscopic particles. They can be characterized as a form of matter distinct from solids, liquids, or gases, because they behave so differently under different conditions. For example, a pile of sand on an inclined plane will behave like a solid when the angle of incline is shallow: it will retain its shape and structure, and does not expand to fully fill the available space like a liquid would do. However, when the angle is steeper, it behaves like a non-Newtonian fluid, with no constant coefficient of viscosity. The causes of this interesting and dynamic behavior are the inelastic collisions that occur between the particles and the general unimportance of temperature to the overall system. The inelastic collisions mean that the energy of the system varies dramatically from moment to moment, and the temperature of the system is unimportant because the particles are macroscopic. The difficulty of defining an energy and temperature for the system means that using statistical mechanics to describe and predict the behavior is not helpful[1]. Because of this, more experimental methods of exploring the effects of variables such as shape of material, concentration or mix of different shapes, and location of those different shapes are helpful to gain a better understanding of granular materials. This project seeks to determine the effect of these variables on the angle at which avalanching occurs.

II. PROJECT DESCRIPTION

A. Setup

In this project, a round, rotating drum is partially filled with metal ball bearings that are welded into different shapes (see Fig. 1). The width of the drum is such that only one layer of ball bearings fits between the front and back, so it is effectively two-dimensional. The setup is designed such that the concentration of shapes as well as the shapes themselves can be easily varied by emptying the drum and refilling with the desired materials. A motor rotates the drum at a constant speed, and so the pile of ball bearings increases in angle until it eventually avalanches down. A video camera records the entire process, and we examine the frame directly before the avalanche occurs in order to relate variables such as material shape, concentration, and location to the angle of stability. This requires identifying the location of the different shapes present in the image, which needs to be automated in order to analyze large amounts of data efficiently. Code to perform this task was developed in IDL, reaching a high accuracy in 2009. It then needed to be translated to Python, and has not reached satisfactory performance levels since translation.

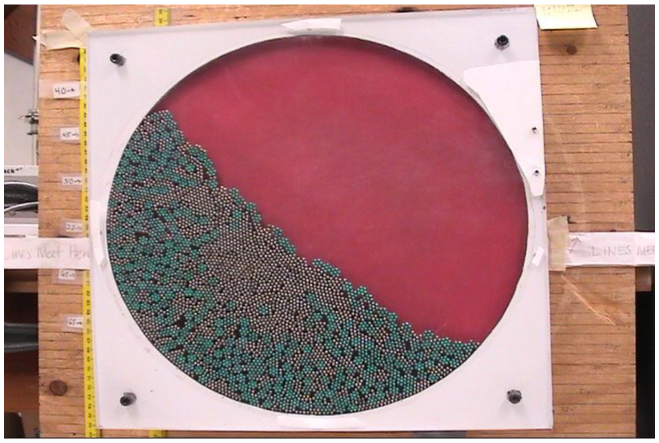


FIG. 1. The experimental setup consists of a rotating drum filled with metal ball-bearings, welded into different shapes.

In the example setup in Fig.1, there are two shapes present: hexagons, which consist of a single central ball with six balls welded around it, and doubles, which consist of two balls welded together. The hexagons are painted green for easier identification. The example image is a mix of 50 percent hexagons, 50 percent doubles by weight.

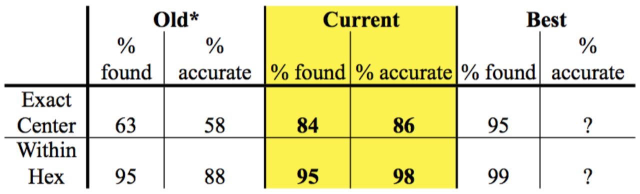
B. Automated Analysis

The program needs to be able to identify the location of each different shape in the image. The automation was developed on images similar to that shown in Fig. 1, which consist of 50 percent hexagons and 50 percent doubles by weight. In this case, it is sufficient to locate the center of each hexagon in the image (once all individual balls have been located) to determine whether any given ball is a part of a hexagon or a double, and therefore to be able to see trends in the location of the different shapes. To that end, the program first determines the radius of the drum and the rightmost edge of the pile of balls, to place limits on where to look for balls. It then locates each individual balls by searching for the spots of brightness that are reflected by each individual ball. Once all the balls have been identified, the program conservatively labels green balls and silver balls (only balls which are very certain to be that color are labeled as such, to avoid difficulty later on). There is also a "fuzzy" ball category for those balls which could be green, but are not clear enough to definitively determine their color. Much of this uncertainty is due to the reflections of other nearby balls, which can easily distort the apparent color, and the relatively low resolution of the camera, which was a necessary sacrifice because of the memory taken up by the large amount of footage.

Once the balls have been identified and labeled by color, there are two different methods for locating hexagons that are combined to yield the final results. First, a collection of neural networks looks for hexagons. It uses the RGB values and various parameters associated with the location of the ball center to the ball centers around it. It labels some as definitive hexagons, while others are labeled as candidates. Then, a deterministic approach (looking at similar parameters) scores each ball on its likelihood of being a hexagon, taking into account the score from the neural network as well.

III. PERFORMANCE

Over the course of the summer, various changes were made to improve the performance of the program. Many of these changes were related to problems that were introduced when translating the code from IDL to Python, such as differences in how the two languages iterate through lists, denote empty arrays, or concatenate arrays. In several cases, these mistakes had gone unnoticed because they were only affecting the second or third pass through the image, and the first pass was still occurring normally. This meant that the program could successfully run, but was not nearly as sophisticated as it had been before translation. One original improvement introduced this summer involved expanding the radius in which the code searches for the balls to be the maximum detected radius rather than the average detected radius. This is significant because the drum is not an exact circle, and so depending on the orientation of the drum, the outer layer of balls could be missed. This leads to complications in the later stages, as the position of a ball relative to its neighbors is a key piece of hexagon identification. These changes resulted in a significant improvement in performance, as shown in Fig. 2.



	Old*		Current		Best	
	% found	% accurate	% found	% accurate	% found	% accurate
Exact Center	63	58	**84**	**86**	95	?
Within Hex	95	88	**95**	**98**	99	?

FIG. 2. Current performance statistics. "Exact centers" means the program correctly identifies the ball at the center of the hex, while "within hex" means it identified the hex but might have placed the center in the incorrect ball. "Percent found" gives the percentage of the true hex centers that were correctly identified, and "percent accurate" gives the percentage of the hex centers that were found that are true hex centers. *The "current" and "best" numbers are based on averages, while the "old" numbers are based on a single image.

The program is still not up to its peak level of performance: although it is nearly as good at finding a hex in general, it is still significantly worse at finding the exact hex centers. Fig. 3 shows the location of all the true hexes, which were manually located for comparison. Fig. 4 shows that the program frequently places the hex centers adjacent to the true center.

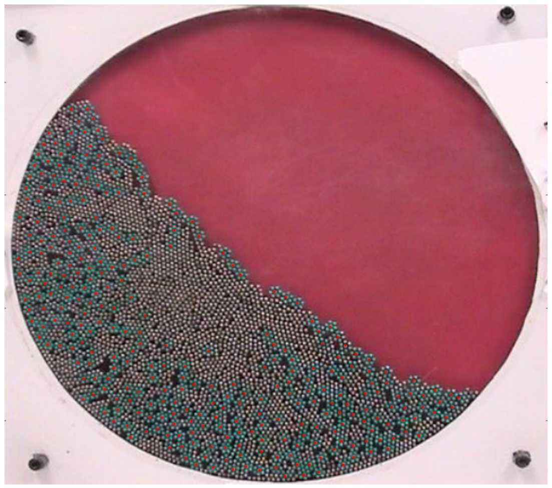


FIG. 3. This image shows the true hex centers (selected manually) in red.

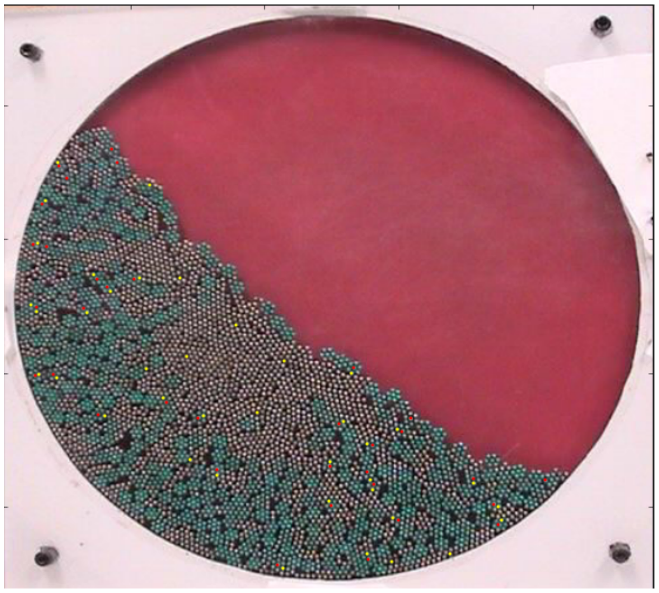


FIG. 4. Visual depiction of current performance, with both neural networks and the heuristic approach to finding hexes. All hex centers that were correctly identified are omitted from the image. Red shows the true hex centers, while yellow shows the incorrect hex centers identified by the program.

To gain more insight into the ongoing problems, we analyzed the image using the neural networks and the deterministic algorithm separately. Fig. 5 shows the results of using the neural network alone, while Fig. 6 shows the deterministic results alone (to see the combination, refer back to Fig. 4).

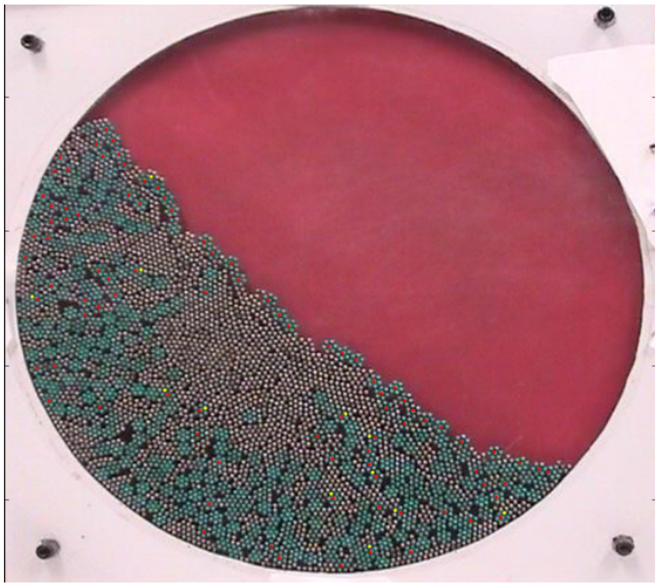


FIG. 5. This image was analyzed using only the neural networks to locate hexes (excluding the heuristic approach). All hex centers which were correctly identified are omitted from the image. Red shows the true hex centers, while yellow shows the incorrect hex centers identified by the program.

There are very few red-yellow "pairs" in this image (which occur when the program identifies a ball in the hex that is not the true center as the center), which shows that the neural network is good at identifying the true center. However, there are many red centers that are not even near a yellow, which indicates that the neural network is entirely missing those hexagons. This is especially prevalent along the rightmost edge of the pile of balls, as well as in the top portion.

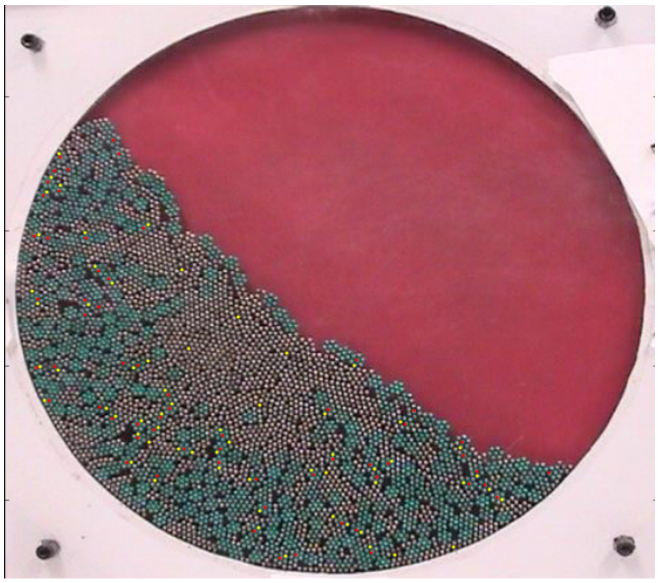
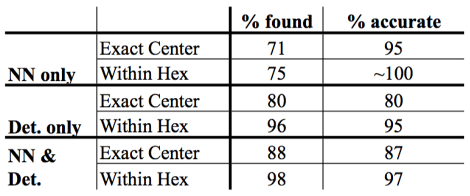


FIG. 6. This image was analyzed using only the heuristic approach (excluding the neural networks). All hex centers which were correctly identified are omitted from the image. Red shows the true hex centers, while yellow shows the incorrect hex centers identified by the program.

The deterministic approach is less accurate, but locates additional hexagons that the neural networks did not, such as the hexagons along the rightmost border and in the top section. There are more "pairs" of red and yellow dots, indicating the program identified the hex correctly but placed the center in the wrong location.

These visual results are summarized numerically in Fig. 7. The neural network is significantly more accurate than the deterministic method, but misses more of the true hexes as well. The deterministic method is fairly good at identifying true hexes and at doing so accurately. Based on the reports from when the program was working at peak performance, it appears the deterministic portion of the code is working at capacity, while the neural network portion still needs a lot of improvement to get back to the accuracy it had in 2009. Also, the combination of the neural networks and the deterministic approach is still significantly improving the overall results, as was found in the original implementation in IDL.



		% found	% accurate
NN only	Exact Center	71	95
	Within Hex	75	~100
Det. only	Exact Center	80	80
	Within Hex	96	95
NN & Det.	Exact Center	88	87
	Within Hex	98	97

FIG. 7. This table shows the overall results of analysis on a single sample image using the neural networks alone, the deterministic method alone, and then the combination of both.

IV. CONCLUSIONS AND FUTURE WORK

Significant improvement has been made in the code, but it still needs to be more accurate before it can be used to examine the variables of interest. From comparison to the paper from 2009[2], it appears that the code up until identifying the actual hexagons is all working properly (including the identification of the area containing balls, identifying the balls themselves, and labeling the balls as green/silver/fuzzy). The deterministic method of finding hexes alone also appears to be working at around peak performance (perhaps slightly under, although it is difficult to tell because there is not complete data to compare to). This means that the main area that still needs improvement is the neural network. It's possible that the network files themselves still have some problems left over from translation, or that the code that runs them was translated incorrectly. They do run, and obviously work fairly well, so it is most likely a small error or perhaps a problem with one iteration of a loop. It is also possible that the networks simply need to be retrained after the earlier changes to the code.

Regardless of whether the networks need to be retrained to analyze this set of images, there is a new set of images that were taken under different lighting with a new camera, and it seems like it will be necessary to retrain the neural networks once those images need to be analyzed. Currently, the networks can be "retrained" in that the retraining process successfully produces new networks which can be run by the code without producing errors. However, they typically find somewhere in the range of 0 to 4 hexes, rather than over 200 as the old neural networks did. The source of this problem is unclear, as the old code to retrain networks (which was presumably the code used to train them in the first place) is being used as much as possible. There may be some issue with the way the data is being fed into the networks, or a disparity between the data being fed in to train the networks and the data the networks are being asked to analyze.

Once the networks are again working satisfactorily and the retraining process is successful, the code should be adapted to be able to handle different shapes and concentrations of those shapes (it is more difficult to analyze an image which is almost all one shape, so those cases may require neural networks trained on images of that concentration, rather than on the 50/50 case). The code could then be used to analyze large amounts of data in relatively short times (each image takes approximately 3 minutes to run), leading to conclusions about the effect of different variables on the angle of stability of the pile.

V. ACKNOWLEDGMENTS

Thanks to Rena Zieve for her guidance on this project, as well as the opportunity to participate in the REU program. Thanks to the NSF for funding the REU program.

[1] H. Jaeger, S. Nagel, R. Behringer. *The Physics of Granular Materials*, Physics Today*49*(4), 32(1996).

[2] Eldridge, Justin. http://london.ucdavis.edu/ reu/REU09/eldridge.pdf