IOWA STATE UNIVERSITY
Digital Repository

2008

Characterization of geomaterials with X-ray computed tomography (X-ray CT)

Clinton Halverson
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Civil Engineering Commons

Characterization of geomaterials with X-ray computed tomography (X-ray CT)

by

Clinton Halverson

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Civil Engineering

(Geotechnical Engineering)

Program of Study Committee:
David J. White, Major Professor
Joseph Gray
Igor Beresnev
Kejin Wang

Iowa State University

Ames, Iowa

2008

UMI Number: 1453125

UMI

UMI Microform 1453125

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

THESIS DEDICATION

This thesis has been brought to fruition due to the dedication and support of
my family, my friends, and the ISU faculty who have graced my life.
Without their generous encouragement, guidance, and commitment this
research project could not be possible. This thesis is dedicated to all of you.
God bless you all.

A special thanks to:

My parents who vocalized and exemplified the importance of education.

Dr. David White and the staff at Town Engineering Building whose
kindness and time was there when I needed it.

Dr. Joseph Gray and the staff at ASC who always made me a priority even
while their time was so divided.

TABLE OF CONTENTS

GENERAL ABSTRACT

The application of 3-dimensional volumetric imaging using non destructive evaluation procedures opens doors in researching geomaterials, allowing study of internal characteristics. X-ray computed tomography (CT) is one such procedure which is especially well suited to geomaterials due to the high density contrasts often found between solid materials and their surrounding environments and/or pore spaces. Although X-ray CT has been applied to geomaterial research in the past, there is a lack of tested X-ray CT methodologies which can be readily applied to many important research tasks. Methods developed and applied herein concern the analyses of soil particle movement during application of stress and pore space characterization in terms of size, shape and pore continuity.

The general objectives undertaken in this research were to:

1.) Develop a methodology which can be used to quantify soil particle directional movements and rotations during incremental loading.
2.) Apply the above methodology to cylindrically shaped triaxial samples of poorly graded aggregate under incremental strain.
3.) Develop a methodology whereby high quality cross sectional maps of density variation can be produced and void spaces correlated to hydraulic conductivity in terms of pore size, shape and continuity.
4.) Apply the above pore space characterization methodology to cylindrical samples of Portland cement pervious concrete (PCPC) mix designs.

In order to obtain the goals, specialized methods to conduct X-ray CT scans and new post processing techniques were developed for both digital image processing and data reduction.

It was found that PCPC can be successfully imaged using X-ray CT and that the measurements can be filtered to remove artifacts associated with X-ray CT scanning. Blurring at the edges of scanned images can be mitigated by shielding the sample during the scanning process with a ceramic or cement material around its perimeter. Resulting X-ray CT cross sections can be divided into pore space and solid space regions by correlating a cutoff image voxel intensity value to known void ratio distributions within sample volumes. Void space cross sections produced can be post processed to remove grainy edges and further remove ring artifacts.

A specialized methodology for computing three dimensional particle movements was conceived and consisted of fixing three identifiable segments of lead solder to the particle. X-ray scanning conducted at incremental strain intervals allowed tracking of these markers. The alignment of markers to one another created vectors which could be tracked for quantifying particle movement including rotations. The movements of particles were computed and displayed as a function of position according to height and radial distance within the radially symmetric samples.

Results of these methods show that in the case of PCPC samples, larger pore size, pore shape (perimeter/area), void ratio, and pore continuity are traits of samples which demonstrate higher permeability. Also, regions of significantly higher permeability were found at the edges of samples which were determined to be due to boundary effects resulting from sample preparation within a cylindrical mold.

Key findings from tracking aggregate particle movements within triaxial samples includes the following:

> 1.) Particle movements are generally concentrated in bulging regions within samples.

2.) Average aggregate grain motion becomes increasingly horizontal in direction until a point just before peak deviator stress is achieved after which the averaged inclination of radial to vertical displacements becomes predictable.

3.) Higher confining pressures result in less horizontal particle movement but greater rotations.

These research results bring greater understanding to the relation of pore characteristic to permeability and particle movements under triaxial stress. More importantly, though, are the versatile methods developed and tested in this research which may be useful to a variety of applications in analyzing geomaterials. Implementation of these methods has the capability of further quantifying, comparing, and understanding systems relevant to civil engineering to an extent only limited by the imagination and technical capability of researchers.

CHAPTER 1. GENERAL INTRODUCTION

Project Statement

This thesis focuses on two different applications of X-ray computer tomography (X-ray CT) to geomaterials. The pore spaces within samples of Portland cement pervious concrete (PCPC) were characterized, and the movements of aggregate particles within a triaxially compressed sample were quantified. During the development of project approaches special attention was given to developing repeatable methodologies which can be applied to a variety of granular behavior and pore space analyses.

Movement of aggregate particles under triaxial compression was characterized by calculating the translation and rotation of a discrete number of particles within a triaxially compressed granular sample. To do so, computer algorithms were developed to calculate movements relative to orthogonally oriented vertical, radial, and tangential vectors. Findings include the quantification and comparison of particle translation and rotation in different samples as well as discussion of generalized movement trends. The methodology developed is generally applicable to the analysis of specimens composed of aggregate and subject to incremental strains.

X-ray CT is utilized to investigate the pore structure of four PCPC mix designs. Various procedures were developed to remove undesirable artifacts, calibrate volumetric files for quantitative analysis, and characterize pore distribution, size, and shape. Void continuity is quantified using a newly devised continuity index. Analysis results are compared to laboratory measurements of permeability. The methods generated to compute the necessary findings can be applied in part to virtually any X-ray CT analysis requiring artifact removal or X-ray image intensity distribution calibration but furthermore are specifically applicable in analyzing tortuous, interconnected pore spaces common to geomaterials.

Thesis Organization

The body of this thesis is organized into two chapters (Chapters 3 and 4). Each chapter contains its own abstract, introduction, and detailed descriptions of project methods, procedures, results, conclusions, and recommendations for future research. A general abstract and literature review precedes these chapters. General conclusions and recommendations sections are also included.

CHAPTER 2. LITERATURE REVIEW

Internal Characteristic Determination Methods

To characterize soil particle movements subject to loading and incremental strain it is necessary to view samples internally. There are several means to accomplish internal characterization of soil samples in an undisturbed state including: thin sectioning, ultrasonic imaging, electrical resistivity tomography, magnetic resonance imaging, and X-ray computed tomography (X-ray CT).

Thin sectioning is conducted by injecting an epoxy of relatively low viscosity and curing expansion into a matrix of coarsely grained soil. The sample must remain in the state intended for analysis until the epoxy hardens. The sample may then be cut into thin slices or coupons. This procedure is described and applied by Ayoubian and Robertson (1998), Oda (1972), Kuo and Frost (1996), and Jand et. al. (1999). Thin sectioning can be particularly useful because the finite two dimensional regions can be analyzed closely with the use of powerful microscopes and computer algorithms. Disadvantages of thin sectioning compared to other means of internal visualization are that it is destructive, is labor intensive to cut and polish surfaces, and only provides a representation of a limited number of planes within the sample.

Ultrasonic imaging, also called travel time tomograghic imaging, uses the travel times of ultrasonic waves to locate high and low density inclusions within analyzed samples. It does so by positioning an array of receivers along the outside of a sample's volume, such as a soil core sample, so that a wave source emits waves which must travel through a known vector towards each receiver. By rotating this array/source configuration around the sample, new vectors may be analyzed. Using computer algorithms to compile and analyze travel times at each of these vectors, a map of travel times may be produced for a sample cross section. Since ultrasonic travel time is density dependent, the travel time map may be interpreted also as a density map. Ultrasonic imaging is non-destructive, but compared to

X-ray CT ultrasonic imaging provides much lower quality and poorer resolution images (Daigle et. al. 2005).

Electrical resistivity tomography uses electrodes located around a sample's exterior, such as a soil core sample, to determine the resistance to electrical flow from one electrode to another. By altering the position of these electrodes a large number of resistivity vectors may be compiled and analyzed to produce cross sections of sample resistivity. Since sample density is inversely proportional to resistivity, the cross sections may be used to inspect samples for regions of high density, low density, high porosity, etc. Electrical resistivity tomography has an inherent disadvantage compared to wave based tomography in that electricity does not travel along a strait line through sample volumes but rather takes a path of minimum resistance. This adds difficulty in determining the position of low density inclusions. Electrical resistivity tomography is a nondestructive inspection method but does not produce image quality such as can be produced using X-ray CT. Electrical resistivity tomography is described in detail by Binley et. al. (1995).

Magnetic resonance imaging (MRI) uses the excitation of atomic poles to characterize samples internally and nondestructively. Information regarding MRI here is summarized from Liang and Lauter (2000). Atoms with odd atomic weights or atomic numbers have angular momentum referred to as spin. In MRI a magnet creates a strong, uniform static field roughly equal to thousands or even tens of thousands times the magnetic field at the earth's surface (typically 0.2 to 1.5 Tesla) which aligns atoms. Radio frequency waves cause magnetic excitation so atoms rotate from their equilibrium position and subsequently relax. Difference in relaxation time is the major factor in MRI image contrast. Also, atomic excitation reaches maximum efficiency at a frequency which is a function of both atom type and magnetic field strength. This allows imaging according to spin density.

Relaxation time of atoms may be either of spin-spin relaxation time or spin lattice relaxation time depending upon if reaction time is based from a loss of alignment away from the magnetic field or the reconstitution of net magnetization in the direction of the

magnetic field, respectively. This differentiation between relaxation phases allows for imaging contrast between materials that relax similarly otherwise.

Since the magnetic field varies linearly from its source a Fourier transformation is required to provide spatial information and images are created tomographically, directly analogous to X-ray CT. In comparison to X-ray CT, MRI may attain image contrast from three differentiating measures of atomic excitation density or time allowing the characterization of samples which would be impossible or difficult otherwise. X-ray CT measures x-ray attenuation and has difficulty separating materials of similar attenuation coefficients. In some applications MRI has the ability to suppress stationary spins so only moving material is imaged, X-ray CT cannot. MRI can prove very useful in applications where ionizing radiation is problematic, attenuation coefficient contrast is low, or material movement characterization is required. X-ray CT has the benefit that X-rays interact with all atoms. Also, X-ray CT has the ability to quickly produce hundreds of slices through a sample in less than an hour while MRI can take hours to create a few high quality slices in typical applications. Both analysis methods are nondestructive and produce high quality internal cross sections.

Computer Aided Tomography

X-ray CT measures the attenuation of X-rays as they interact with matter by using X-ray production and detection hardware. This information is analyzed and spatially assembled using computer algorithms. The result is three dimensional representations of attenuation coefficients throughout a sample's volume. A brief description of X-ray physics is given here along with a description of X-ray CT processes and some examples of its use in soil mechanics.

X-ray production

A typical X-ray tube produces electromagnetic waves by applying a high voltage which accelerates electrons to a correspondingly high velocity, only to have them impact an anode

and decelerate rapidly. The anode is referred to as the target. The energy of such electrons is expressed by the equation:

$$\text{Kinetic energy} = e\text{V} = \tfrac{1}{2}\,m v^2 \qquad (2.1)$$

where e is the charge of an electron (1.6×10^{-19} coulomb), V is the voltage, m is the mass of an electron (9.11×10^{-31} kg), and v is the velocity expressed in meters per second of an electron prior to contacting the target material. Of all the energy applied, less than one percent is transformed to X-rays (Cullity, 1978) and most is converted into heat. These X-rays, being electromagnetic radiation, have a given wavelength, frequency, and energy according to the following relationships:

$$e = h\nu \qquad (2.2)$$
$$c = \nu\,\lambda \qquad (2.3)$$

where e is the photon energy, h is planks constant (6.63×10^{-34} joule seconds), ν is the frequency of the wave, c is the speed of light (3×10^8 m/s) and λ is the wavelength of the wave in meters. The energy of an emitted photon may be no greater than the energy of the incident electron causing the emission. However, most of the photons will have less energy. A continuous spectrum, also called white or bremsstrahlung spectrum, is produced which incorporates many different energies. This spectrum has virtually no radiation at the peak incident electron energy but photon count increases and then in turn drops as wave energy is decreased. An example spectrum is illustrated in Figure 2.1.

Figure 2.1. Continuous spectrum of X-ray photon intensity, tungsten at 105 keV.

As well as the continuous radiation spectrum caused by dissipating electron energy under high voltage, there also exists characteristic radiation. Characteristic is characteristic of a given target material and is caused by an incident electron colliding with an atom's orbiting electron with enough energy to expel that electron. If this occurs in an internal electron shell, the atom is left in an excited state. This vacancy is filled by an electron orbiting in an outer shell. Since different atoms have different shell geometries this photon has different characteristic energy depending upon what type of material composes the X-ray tube's target material. Also, atoms may have multiple shells from which electrons can be expelled or vacancies replaced. Therefore, a given material may produce multiple characteristic energies.

Photon attenuation

X-rays, initially discovered by the German physicist Roetgen, are much more penetrating than visible light and more readily pass through solid objects such as wood, rock, or even metal. However, a sample placed in an X-ray beam still has the ability to attenuate X-ray photons passing through it. This is done by either Compton scattering or absorbing photons.

Scattered radiation is redirected from the primary beam by scattering events while absorbed photons are removed from the primary beam and transferred to the atomic lattice as heat (Barrett and Swindell, 1981). Attenuation of a monochromatic X-ray beam through a homogenous sample is quantified by Beer's Law:

$$\Phi_x = \Phi_0 exp^{(-\mu x)} \tag{2.4}$$

where Φ_0 is the incident x-ray beam fluence, Φ_x is the fluence at a given distance within the sample, x is this distance, and μ is the linear X-ray attenuation coefficient which is characteristic of a sample material. Fluence is the strength of an x-ray beam and has the units of photons per unit cross sectional area. Fluence per unit time increment is termed flux and may be used interchangeably with fluence in describing Equation 2.4. This interaction gives definition to the linear X-ray attenuation coefficient. The mass attenuation coefficient is also used to characterize the attenuation properties of matter and is defined as the linear attenuation coefficient divided by the material's density. Mass attenuation coefficient is highly dependent upon the atomic number of the material according to the following experimentally derived equation (Barrett and Swindell, 1981):

$$\mu_m \approx \frac{k}{A} \frac{Z^m}{(hv_0)^n} \tag{2.5}$$

where k is a constant the depends upon which electron shell is involved, Z is the atomic number of the material, A is the material's atomic weight, h is plank's constant (6.63 x 10^{-34} joule seconds), v_0 is the frequency of the wave, and m and n are parameters roughly equal to 4 and 3 respectively. Equation 2.5 reveals two important points about X-ray attenuation, the first being that high energy photons become more difficult to attenuate. The second is that a high atomic number material becomes an increasingly more effective X-ray attenuator. This explains why X-rays pass through materials more easily than lower

frequency visible light and why lead (atomic number = 82) is a powerful X-ray shielding material.

Another method of describing a material's ability to attenuate X-rays is the Hounsfield unit (H). The Hounsfield unit represents a tenth of a percent of the linear X-ray attenuation coefficient of water. The Hounsfield unit value of a given material is described by Equation 2.6 (Barrett and Swindell, 1981). From Equation 2.6, water results in a Hounsfield unit equal to 0, air results in a Hounsfield unit approximately equal to -1000, and objects denser than water would have a positive Hounsfield unit.

$$H = \frac{\mu_{material} - \mu_{water}}{\mu_{water}} \times 1000 \qquad (2.6)$$

As an additional point, there is a sharp rise in attenuation coefficient when incident photons have energy roughly equal to the binding energy of a deep atomic core level (Ham and Willson, 2005). This energy edge is specific to a material and therefore a significant increase in X-ray attenuation as incident X-ray energy falls below this edge can map the location of said material. Monochromatic X-ray sources are especially adept in this type of analysis.

X-ray CT hardware
Computer aided tomography uses specialized hardware and software to organize X-ray attenuation data into three dimensional maps related to sample attenuation coefficient. The hardware required for scanning includes an X-ray source, a means of X-ray detection, a rotary positioner, and a computer for data allocation/reduction and control. The ability to reduce data on a network of computers is a time efficient addition. The X-ray source, sample positioner, and detector are kept inside an X-ray shielded chamber to protect technicians from ionizing radiation exposure (See Figure 2.2).

Figure 2.2. X-ray CT hardware.

X-ray sources are typically X-ray tubes which use a target material to decelerate high velocity electrons and convert some of that energy towards X-ray production. These tubes utilize voltages of hundreds of kilovolts in X-ray production. Also available are synchrotron radiation sources which circulate electrons in a storage ring to produce X-rays. Synchrotron radiation has higher intensity than traditional X-ray tubes and emits a highly collimated beam to avoid blurry projections (Ham and Willson, 2005). Some X-ray tubes have very small spot size beams and are referred to as microfocus X-ray CT. True microfocus X-ray CT utilizes x-ray beam sources with spot sizes of only two to five microns and can produce volumetric digital files with effective pixel sizes of a few microns (Zhang et al. 2003).

A rotational positioner is required in order to provide multiple vantages through a sample. As the X-ray data is collected a two dimensional projection is created on the X-ray detector. By rotating the sample a small increment a new projection is created. By rotating the sample a cumulative 180° (minus a single rotation increment) a sufficient number of vantages can be acquired to start a reconstruction. Rotary positioners are typically powered by quality servo or stepper motors.

An X-ray detector for X-ray CT consists of a scintillator and a means to decipher the two dimensional distribution of X-ray photon flux into an electric output signal. An image intensifier may be used as a scintillator which produces a visible light display from incident X-ray flux. This display is then imaged by a camera and digitally recorded using a frame grabber (Kini, 1994). Also available are flat panel imagers which are able to produce digital output without the need of a camera such as amorphous silicon panels.

CT image processing

Several different means of analyzing X-ray attenuation data exist, but the cone beam filtered back projection method will be discussed here. A filtered back projection algorithm differs by the geometry of the X-ray beam being attenuated. X-ray "beams" may be line beams, fan beams, or cone beams. Line beams have the simplest geometry as they pass through a sample since they are all parallel to each other. Data collected with line beams is done one point at a time. Fan beams pass through a slice of a sample and project the attenuation of X-rays for that slice. This makes data analysis more difficult geometrically because of the differing angles of X-rays through the sample. However, data collection time is reduced significantly. A cone beam is again more difficult geometrically but more time efficient. A cone beam can be thought of as a fan beam of fan beams. Here the entire volume of a sample is projected so only a single projection needs to be completed at each rotation step to collect sufficient attenuation data. See Figures 2.3, 2.4, and 2.5 for illustration.

Figure 2.3. Line beam projection of a point.

Figure 2.4. Fan beam projection of a slice.

Figure 2.5. Cone beam projection of a volume.

A cone beam filtered back-projection reconstruction involves the following five steps (Kini, 1994):

1.) Gather original X-ray projection data in the form of two dimensional projections
2.) Modify projections for cone beam angle geometry
3.) Apply a smoothing filter to remove noise
4.) Convolve filtered rows of projections to create back-projections of each sample slice
5.) Stack filtered back-projections upon each other to form a three dimensional volume. Weighting is required to compensate for the tilt of fan beams at each vertical interval.

To organize attenuation data taken at differing angles through a sample into two dimensional coordinates, Fourier slice theorem is used. The general concept of back-projection is summating (smearing) all gathered projections across a two dimensional space to recreate an image of the sampled object's cross section. This is illustrated in Figures 2.6, 2.7 and 2.8. As projections are smeared, locations of overlap are consistently reinforced, causing greater intensity which determines the locations of solid volumes.

Figure 2.6. Shadow of X-ray attenuation on photon detector.

Figure 2.7. Shadow of X-ray attenuation from multiple incident angles.

Figure 2.8. Smearing of shadows to produce a back-projection. (Reproduced from Barrett and Swindell, 1981). Regions of consistent back projection overlap are object locations.

Problems in CT scanning

There are several problems inherent to many X-ray CT systems. These include noise, partial volumes, artifacts, and beam hardening. The erroneous effects of these individual problems may be reduced using different techniques.

Noise is considered changes in photon flux from X-ray generation. Noise therefore causes uncharacteristic projections of X-ray attenuation. Since noise is not consistent, multiple frames of the same projection may be taken and frame averaging implemented to statistically average out noise effects (Kini, 1994).

The partial volume effect is a problem caused by X-ray CT volumetric images representing object volumes with a discrete number of voxels. A single voxel could contain within its extents areas of multiple individual densities. In this case the representative voxel intensity will be an average of the weighted intensities of these areas. This causes a thin intensity gradient around the extents of materials large enough to have one or more voxels completely located inside its volume. If material regions are sufficiently small, a voxel may represent multiple units of the material averaged with other materials found within the voxel's volume. The partial volume effect may be problematic when quantifying functions of density such as void ratio in granular samples (Alshibli and Alramahii, 2006(a)).

Artifacts are undesirable results not necessarily caused by noise. A common artifact is the ring artifact. This artifact is identified by a ring around the center of sample rotation and is caused by dead or overactive regions on the X-ray scintillator. Ring artifacts may be removed by post processing software.

Polychromaticity refers to having multiple frequencies of radiation. If X-ray production is not filtered with a monochromator, a white light spectrum is produced. As discussed earlier, X-ray attenuation is a function of X-ray frequency and sample atomic number. Therefore lower energy X-rays are attenuated early in sample penetration. This uncharacteristically high attenuation of X-rays at the sample's exterior X-ray is translated during volumetric

file production as a region of denser material than it actually is. This can prove problematic for quantitative analyses and is called beam hardening.

One means to reduce beam hardening is to attenuate low energy photons from the X-ray beam. This may be accomplished by covering the X-ray generator with a thin sheet of material such as copper or aluminum (Kini, 1994). Other ways to reduce beam hardening are preprocessing projection data, post processing of reconstructed images, or dual energy imaging. In dual energy imaging a sample is radiated with two different energy spectra. By identifying and decomposing differences in results from these two spectra, an energy independent solution may be found (Barrett and Swindell, 1981).

CT applications for geomaterial characterization
Due to X-ray CT's ability to non-destructively image sample density distributions in three dimensions, many material engineers have turned to X-ray CT as a research tool. Specific to geotechnical engineering is its applications towards characterizing fluid infiltration, density distributions of intact samples, and particle size, shape, and relative movement under strain. The following summarizes several case histories and examples of geomaterials characterization using X-ray CT.

X-ray CT was employed to image water movement in soil mixed with air foam (Kikuchi et al. 2006). Permeability in lightweight treated soil mixed with air foam (LWS) is a critical issue. Its use as a water retention structure may be jeopardized by water infiltration, especially by harmfully corrosive seawater. X-ray CT was used to determine the amount of air present in the soil mixture. The study substantiated water channel formation when entrained air levels were too high. This supports permeability measurements which range from a coefficient of permeability of 1.47 x 10^{-7} with a density of 1.10 g/cm^3 to a permeability of only 1.87 x 10^{-3} when density falls to 0.60 g/cm^3.

Water movement was traced in sandstone samples using X-ray CT (Fukahori et al. 2006). A potassium iodide tracer was added to infiltrating water to track its headway into

sandstone. The water permeation was graphically recorded at time intervals and the coefficient of permeability recorded accordingly. By altering the lateral confinement of the sandstone, a graph of permeability versus confining pressure was produced.

Monochromatic energy was used to differentiate multiple phases of fluid (Ham and Willson, 2005). Synchrotron X-ray generation coupled with a monochromator produced the monochromatic X-ray beam. A saturated porous sand matrix had its original fluid, or wetting fluid, displaced by nonwetting fluid of a different element. By varying the energy of incident X-rays above and below the attenuation energy edge of the fluid elements, the locations of said elements where indicated. The end result was the ability to qualitatively and quantitatively describe the distribution of multiple fluids inside the porous matrix.

Aperture widths in rock were quantifiably gathered using X-ray CT (Walters et al. 1998). Sand cores of a 500 meter deep oil reservoir in Canada were scanned to determine fracture geometry. The density gradient between rock and air is large so sufficiently sizable cracks were revealed and spatially measured. Such pore distribution analysis of undisturbed samples is important to geotechnical engineers (as well as the petroleum industry), and X-ray CT is well equipped to investigate failure geometry and porosity of samples.

Fracture geometry in granite was investigated by a process of subtracting scans from one another (Sato et al. 2006). In initial scans samples were dry. Samples were then allowed to be infiltrated with water and scanned again. By subtracting consecutive scans, regions of differing attenuation coefficient, the infiltrated cracks, were delineated. Stacking (frame averaging) was utilized to reduce noise and thereby allow small cracks to be seen more readily. Crack dimensions were measured through both the CT scans and by using a profilometer for comparison. It was concluded that detailed crack geometry could be analyzed using X-ray CT.

High resolution micro CT technology was employed to find porosity in synthetic quartz rock cores created under varying confining pressures (Alramahi et al. 2005). As samples

were densified under higher pressures their porosities decreased and their attenuation coefficients increased. Scans were interpreted according to ASTM guidelines (ASTM E1935, 1997). This guideline outlines a procedure to scan a disk with inclusions of know density so CT produced images may be calibrated accordingly. Different combinations of sample materials and confining pressures were analyzed. Their different frequency curves of porosity measurements from millions of locations inside their volumes were compared. The experiment revealed that consolidation takes place up to around 4000 psi confining pressure, but as pressures reach 5000 psi the samples became more porous, possibly due to the breaking of particles.

A research study altered CT images with a method of removal and expansion (removing then adding a layer of specified thickness around specified regions of an image) to determine the distribution of particle sizes in X-ray CT slices of a granular sample (Mokwa and Nielson, 2006). The process, called granulometry, removed sufficiently small particles in the removal step. During the expansion step, only particles which remained had the outer layer re-applied. By summing particle areas before and after this process, the difference is considered the area of spaces small enough to be removed by the process. By varying the removal and expansion thickness, a database of particle areas was collected. Since an X-ray CT slice may cross through any random horizontal plane of a particle, statistical analyses methods needed to be implemented to find the likely grain size distribution of the scanned sample, called stereology. The results resembled actual grain size distributions but no measure of deviation between the two was provided.

The zone of influence from cone penetration into a sand sample was analyzed using X-ray CT (Ngan-Tillard et al. 2005). A uniformly sized sand with 0.150 mm mean grain diameter was penetrated with a 6 mm diameter pointed rod. At regular intervals the penetration was halted. The sample was then removed from the test mechanism (while holding the rod static and taking care not to cause sample disturbance) and CT scans were conducted. As soil was sheared, dilation occurred. This dilation was depicted in CT scans as regions of lower X-ray attenuation. By subtracting subsequent scans from each other, regions of soil which

were disturbed between scans were revealed. The CT equipment was calibrated prior to scanning by graphing hounsfield unit values versus sand density, which revealed a linear relationship with R^2 deviation of 0.992. The relationship allowed quantification of sand density directly from X-ray CT scan outputs. The project revealed that most shearing occurred under the rod's cone tip, not along its shaft. It also showed that the zone of disturbance (in this case roughly 2.6 cone diameters around the penetrating rod) shows a "small decrease" in size by halving the sample's confining pressure but shows a "significant decrease" from lowering the sand density.

The failure region resulting from lateral pile loading was modeled using X-ray CT (Pham et al. 2006). A flat aluminum rod imbedded into a pressure confined sand sample was slowly moved by adding constant lateral deflection to its top. Measurements of resisting force against movement, strain gauge readings along the rod's axis, and X-ray CT scans were collected to characterize the pile loading. CT imaging showed that a skewed cone shape of displaced soil was formed. The skewed cone extended out from the rod in the direction of its travel and was oriented with its widest failure region at the upper surface of the sand sample. X-ray CT also allowed the failure cone's volume to be measured as a function of rod deflection. This information reveals to soil engineers the failure geometry of laterally loaded piles in comparable conditions.

Failure zone development in triaxially stressed, fine grained sand samples was modeled using X-ray CT (Alshibli et al. 2000). In order to avoid the effects of gravity on the sample during loading, the experiments were conducted in outer space in SPACEHAB on the Orbiter. Sand samples were created using air pluviation prior to departure and triaxial compression was conducted at low confining pressures of 0.05, 0.52, and 1.3 kPa (0.007, 0.0754, and 0.189 psi) in zero gravity. During the experiment all normal macroscopic measurements were taken including volume change, deviator stress, and vertical strain. X-ray CT was used to view internal dilation. Since the resolution of CT scans was roughly the size of the smallest particles, a blurring effect was created and regions of dilation could be seen as regions of lower CT number (lower X-ray attenuation). The CT data revealed that

two conical shaped shear zones formed, beginning at the end plates and narrowing as they approach the sample center. The cones were shaped slightly convex outward. Outside of this region, radial cracks formed, possibly caused by tension in the expanding soil.

Another study characterized dilatant behavior in sand samples in a similar way (Desrues et al. 1996). Samples of fine sand (0.32 mm mean diameter) were prepared and underwent triaxial compression. By calibration X-ray CT outputs against known mass densities, volumetric files could be analyzed and translated in terms of void ratio inside the sample volumes. By systematically halting loading to scan samples, failure plane geometry and void ratio therein could be characterized. The results revealed that loosely packed samples (20% relative density) collapsed throughout their volumes to a rather homogenous density. Dense samples (90% relative density) acted very differently as dilation was concentrated on the failure planes which formed. The void ratios in these failure zones were comparable to the void ratios found throughout the volumes of loosely packed samples.

In order to track individual particle movements, an experiment was conducted using white plastic spheres with holes drilled through their diameters (Alshibli and Alramahi, 2006(b)). The holes created an axis from which rotation could be measured. Also, the location of each individual sphere was monitored to determine directional movements. Cylindrical samples were created using hundreds of spheres and subjected to axisymetric compression. A computer analysis of particle locations at different scanning intervals was conducted, and a movement database was gathered and illustrated with frequency curves. The curves depicted a high degree of variance with standard deviations sometimes almost equal to the mean values recorded.

Image analysis
After data collection and volumetric file production are completed there are many processes by which the quality of data can be increased for analysis purposes. Many such

concepts have already been mentioned such as smooth filtering, removal and expansion, and image subtraction. In further discussion on this topic there is the use of binary images.

It is often important to distinguish between different materials when analyzing digital image data. In X-ray CT data, a CT number cutoff can be developed above which certain materials can be identified and below which other materials are identified. This process is called thresholding (Mokwa and Nielson, 2006). Examples of geomaterials components subject to thresholding include rock, air, water, soil and cement. In order to apply thresholding, beam hardening trends must often be removed. Once the subject material has been separated from other materials using thresholding, it may be depicted as a single color, and all other space may be allotted a different color. The result is a binary image showing the subject material alone.

Binary images are useful because they identify blob regions (areas within the intended threshold) for program analysis. Programs commonly use binary images to quantify blob size, location, and shape. Size is determined as the number of connected voxels inside a blob. Location is the center of such a blob. Shape can be quantified by a blob's perimeter or by using a shape factor. Shape factors are useful because they quantify the roundness of a pore as described in Equations 2.7 and 2.8 (McClain et al. 2003)

$$SF = \frac{P^2}{4\pi A} \qquad\qquad (2.7)$$

$$SF_{AW} = \frac{\sum_{i=1}^{n} SF_i A_i}{\sum_{i=1}^{n} A_i} \qquad\qquad (2.8)$$

where SF is the shape factor of a blob, A is the area (or size) of a blob, P is the perimeter of blob, and SF_{AW} is the area weighted average of all shape factors for blobs numbered 1 through n. Using Equation 2.7 the shape factor of a circle is equal to 1 and is the lowest

possible shape factor. Having a higher shape factor than 1 indicates the ratio of blob perimeter to the perimeter of a circle with equal area. The weighted average of all shape factors is a means to organize multiple shape factors found inside a given region for comparison with shape factors found in other regions.

Drained and Unsaturated Granular Reaction to Vertical Stress

Granular, coarse or cohesionless materials are all names which refer to soils having large enough grain sizes to act frictionally upon each other but effectively do not have the cohesive interaction of smaller particles (silt and clay sized particles). In geotechnical engineering, granular material may be considered any rock larger than a silt sized particle (See Table 2.1).

Table 2.1. Grain size classification according to USCS (Das, 1998)

Unified Soil Classification System	
Classification	Grain Size (mm)
Clay	< 0.002
Silt	0.002 to 0.075
Sand	0.075 to 4.75
Gravel	4.75 to 76.2

Granular samples are usually composed of three phases: solid, fluid, and gas. Solid phase is the rock material itself and the voids between particles may contain gas (usually air) or fluid (usually water). Granular samples react differently when they are saturated (voids effectively filled with fluid) than when unsaturated. One difference between saturated and

unsaturated samples in the influence of water pressure upon the effective stress state (σ') of soil samples as defined by Equation 2.9.

$$\sigma' = \sigma - \mu \tag{2.9}$$

where σ is the total stress (weight of all materials pressing upon the position of interest) and μ is the pore water pressure at that height. Pore pressure builds with depth when the soil is saturated with water. Under dynamic loading such as earthquakes or landslides, pore pressures may build up beyond static water pressure. If drainage of water from soil is not permitted during soil strain, the movement may induce changes in water pressure. Undrained triaxial compression tests can model this behavior. However, by testing granular materials in unsaturated and drained conditions any difficulties caused by water pressure may be avoided as total stresses are equal to effective stresses.

During triaxial compression, samples tend to form two zones of shearing failure of conical shape. These slightly curved, convex outward cones extend from the cylindrical sample's ends and touch tips near the sample's center. If samples are too short (heights of triaxial specimens are typically twice the measure of their diameters) a single cone may form extending the height of the sample (Desrues et al. 1996). Research using lead shot revealed that at high confining pressures a shear plane may cross diagonally through the sample (Newland and Allely, 1957). In the case study, confining pressures reached roughly 80 psi. It was noted though that identically prepared and confined samples above 80 psi confining pressure could also develop a bulging failure pattern characteristic of a dual cone failure.

Shearing failure criteria

Mechanics of materials uses Mohr's circles to model shear stress (τ) developed from major and minor principle stresses (σ_1 and σ_3) (See Figure 2.9). Under most foundation loading conditions it is assumed that σ_1 is oriented in the vertical direction while σ_3 is assumed to be radially equivalent in all directions on the horizontal plane (all directions perpendicular

to the vertical axis). Any point on the Mohr's circle is considered a stress state within a sample. As the minor principle stress (the confining pressure in triaxial compression testing) is altered the major principle stress and shear stress which the sample can undergo varies proportionately (See Figure 2.10). This effect is defined numerically by the angle of the line which is tangent to consecutive Mohr's circles, referred to as the angle of friction, ϕ. By geometry, this relationship for granular materials is defined by Equation 10. This is the concept of Mohr-Coulomb failure criteria.

$$\tau = \sigma \tan \phi \qquad\qquad (2.10)$$

Figure 2.9. Mohr's circle presenting principle stresses and shear stress at failure

Figure 2.10. Friction angle of sand under axial compression

Stress strain response

The stress versus strain relationship of soils is often modeled linear elastically according to Equation 2.11 known as Hook's law

$$E = \frac{\sigma}{\varepsilon} \qquad (2.11)$$

where σ is the stress upon the soil, ε is the strain of the soil in the direction of stress, and E is the soil's modulus of elasticity or Young's modulus. Modulus of elasticity is measured at the initial, roughly strait line portion of stress strain response (See Figure 2.11). It can be seen that this model is relatively reliable for modeling strain up until the stress strain curve tapers off near failure, at which point the initial linear relationship becomes unrepresentative (Lade, 2005).

Instead of using elasticity, soil compression is commonly represented as a change in void ratio, Δe, of a soil under increases stress, Δσ. Void ratio is simply the volume of voids in soil divided by the volume of solids. A graph of void ratio versus pressure develops a logarithmic relationship as increased pressure compresses the soil. This plot is

characterized by a consolidation index, Cc, and a swell index, Cs, as shown in Figure 2.12 where the curve containing Cc is the soil's reaction to compression and the line containing Cs is the soil's reaction to releasing pressure . The value of these indexes as pressure moves soil from one void ratio to another is defined by Equation 2.12. These parameters are important to understand soil compression behavior and are used directly in soil deformation models such as the Cam Clay model.

$$Cc, s = \frac{\Delta e}{\log \dfrac{(\sigma_0 + \Delta\sigma)}{\sigma_0}} \qquad (2.12)$$

Figure 2.11. Linear elastic stress strain relationship
(Reproduced from Lade, 2005)

Figure 2.12. void ratio versus log stress curve. Slope Cc is loading and Cs unloading. (Reproduced from Das, 1998)

Other stress strain models exist which attempt to correct for non-linearity in the void ratio versus the log of pressure curve. Examples include the Modified Cam Clay and hyperbolic models. The Modified Cam Clay model incorporates Cc, Cs, friction angle, and preconsolidation pressure with its corresponding void ratio into a model that is designed to describe isotropically consolidated soils tested under triaxial conditions (Chang et al. 1999). The product is a confining pressure dependent function for stress versus strain curve construction. There are many variations of Cam Clay that have been developed over the years for various uses.

Hyperbolic model depicts the stress versus strain relationship of soils as having a hyperbolic graphical shape. This hyperbola is defined by Equation 2.13.

$$(\sigma_1 - \sigma_3) = \frac{\varepsilon}{\frac{1}{E_i} + \frac{\varepsilon}{(\sigma_1 - \sigma_3)_{ult}}} \qquad (2.13)$$

where $(\sigma_1 - \sigma_3)$ is equal to the deviator stress, $(\sigma_1 - \sigma_3)_{ult}$ is the ultimate (maximum achievable) deviator stress, E_i is the initial modulus of the soil, and ε is the variable strain level at which the deviator stress needs to be calculated. These parameters may be gained from performing stress versus strain tests on the soil.

Drained triaxial compression of granular media

The term triaxial refers to three planar axes whose planes exert pressure onto a sample. These axes are all perpendicular and include a vertical axis and two axes which lie on the horizontal plane. In traditional soil mechanics (in the United States) the two horizontal axes are often assumed to exert equivalent pressure and therefore lateral pressures on a sample (termed confining pressure) are equal in all directions. A triaxial stress state therefore consists of vertical pressure (confining pressure plus a deviator stress) and confining pressure. These pressures act upon a cylindrical soil sample with a height equal to twice its diameter. By making confining pressure static and increasing the deviator stress a foundation loading situation may be modeled (see following section on stress paths in drained soils).

A triaxial soil compression chamber is shown in Figure 2.13. A clear tube is placed around the sample and allows confining pressures to be exerted on the sample. A latex membrane is used to keep the sample from being permeated by the pressurized medium that gives confinement. The membrane typically has a pair of O-rings fastening both ends by elastic tension. The cylindrical soil sample has plates at its ends of equal diameter to the sample. Through these plates run hoses which can apply or monitor pressures inside the sample if needed. They can also allow venting to the outside atmospheric pressure. Another hose runs through the chamber's base to the region between the soil sample and the clear tube. This is where the liquid or gas used to confine the soil sample is supplied. Typically water is used to add confinement, in which case monitoring the change in water volume inside the triaxial chamber reflects the macroscopic volume change behavior of the sample.

Figure 2.13. Triaxial test chamber and sample setup.

Although there is no ASTM standard for conducting a drained triaxial test on cohesionless soils, it is a practice commonly conducted in researching granular soil behavior as exemplified by Jang and Frost (2000), Desrues et al. (1996), and Alshibli et al. (2000).

Stress path in drained soils

As stresses in a sample move towards reaching peak strength it is often important to understand the progression of those stresses. A graphical plot of stresses within a sample under loading is referred to as a stress path. Different loading situations create different paths. Stress paths may be (and generally are) modeled in p-q space, as defined in Equations 2.14 and 2.15.

$$p = \frac{\sigma_1 + \sigma_3}{2} \tag{2.14}$$

$$q = \frac{\sigma_1 - \sigma_3}{2} \tag{2.15}$$

where σ_1 is the major principle stress on a soil element and σ_3 is the minor principle stress. Stress paths in different loading situations are illustrated in Figure 2.14 (total stresses). If a soil element is in a triaxial testing apparatus which has been consolidated over time (or has a coarse grained structure) it lies at an initial position on the horizontal p axis since horizontal and vertical stresses are equal. By allowing drainage, pore pressure is held constant at atmospheric pressure. A sample sustains stress until it breaches its failure envelope at which point it will fail. The failure envelope angle of inclination in p-q space (α) is related to the friction angle of soil (ϕ') according to Equation 16 (Lambe and Marr, 1979).

Figure 2.14. Stress paths of loading situations from initial consolidation and under drained conditions (from Lambe and Marr, 1979).

$$\sin(\alpha) = \tan(\phi') \qquad\qquad (2.16)$$

The stress path of a soil retention wall results from the increase or decrease in lateral stress from passive or active earth pressures respectively. In the case of passive earth pressure the wall is moving into the soil mass, increasing lateral stress while vertical stress remains constant. Therefore p increases and q decreases in equivalent amounts during loading and the stress path moves with inclination of 45 degrees below the p axis. Soil in an active stress situation is loosing lateral stress and therefore its stress path moves in an opposite direction.

In foundation loading, vertical stress is increased from structural weight while lateral stress is held constant causing both p and q to increase an equal amount until failure may occur. The stress path in this case moves up and to the right at 45 degrees up from the horizontal axis. This is analogous to drained triaxial compression testing. An example of this stress path development can be referenced using ASTM graded Ottawa sand in triaxial compression (Jang and Frost, 2000). The opposite of this loading situation is an excavation whereby vertical load is reduced as overlying soil is removed and the stress path moves in an opposite direction towards failure.

Critical state soil mechanics
The critical state of a soil is reached when there is no volume change due to straining a sample. This concept offers another means to model and understand soil deformation behavior. A cohesionless soil mass compacted to a loose state will collapse upon itself during strain the therefore will show a decrease in void ratio. In contrast, a densely packed granular soil will tend to expand or dilate during compression. The later case results in greater strength and subsidence resistance. There is a density where a sample is neither contractive nor dilative during strain and therefore roughly no volume change occurs. This condition, the critical state, may be determined by plotting volume change versus void ratio. The void ratio where this plot crosses the horizontal axis (zero volume change) may be considered the critical void ratio (e_c) (Seed and Lee, 1967).

The critical void ratio is not a constant for a given soil sample but is dependent upon confining pressure. With the addition of greater confining pressures a soil becomes less able to expand and the critical void ratio increases, as is shown in Figure 2.15. By this relationship we can hold void ratio constant and test to find the corresponding critical confining pressure (σ_{3crit}). This relationship is illustrated in Figure 2.16.

Figure 2.15. Plot of volume change versus void ratio used to determine critical void ratio at differing confining pressures (Reproduced from Seed and Lee, 1967).

Figure 2.16. Plot of volume change versus confining pressure used to determine critical confining pressure at differing void ratios (Reproduced from Seed and Lee, 1967).

The idea of a critical state is useful in understanding stress versus strain behavior and liquefaction potential of cohesionless soils. Stress versus strain curves in granular soils have different shapes dependant upon soil density. Soils denser than the critical state need to expand prior to failure and therefore develop higher deviator stresses than samples which are loose of critical. The stress versus strain plots of sands loose and dense of critical resemble those shown in Figure 2.17. Dense samples not only reach a higher maximum deviator stress but reach peak stress resistance at a lower strain level (i.e. higher initial modulus of elasticity), aiding in soil settlement resistance. The curves for the dense and loose soils approach an equivalent stress versus strain response once critical state has been achieved in the shearing zone. The loss in strength of the densely packed sample after peak stress is termed strain softening while the consistent increase in strength of the loose sample is termed strain hardening.

Figure 2.17. Illustrative vertical stress versus strain results from densely and loosely packed sand samples.

During triaxial strain, samples prepared loose of the critical state will contract and reach critical void ratio homogenously throughout their volumes. Samples prepared dense of the critical state will dilate but the sample's entire volume with not reach critical void ratio. Instead, critical void ratio exists in a discrete number of failure planes within the sample's volume. These effects have been revealed using both X-ray CT (Desrues et al. 1996) and thin sectioning (Jang and Frost, 2000).

Particle movement during shearing

When soil specimens are subjected to shearing, the regions inside the sample where particle movement primarily takes place may be referred to as the zone of failure, shear band, failure plane, or shear zone. There is some ambiguity about how particle movement occurs within these regions and multiple theories have been proposed. Two leading theories in this area are discussed here.

A theory was proposed that modeled particle movement as grains moving mostly independently of each other and rolling over each other on shear planes to accommodate movement under strain (Newland and Allely, 1957). This theory was devised in order to

account for the difference between peak and residual vertical stresses, the condition known as strain softening. In order for sand particles to move relative to each other, they would have to move roughly tangentially to one another (and exactly tangentially when idealized using spheres as representatives of sand grains). This model is illustrated in Figure 2.18 where particles A, B, and C are moving towards the right. Initial tangential movement is inclined upwards from horizontal at angles θ_A, θ_B, and θ_C respectively. The tangential inclination of particle movement is at an initial maximum compared to that at further strains, explaining the initially high and subsequently declining resistances to deviator stress. In this model, particles are anticipated to move independently of each other except some "bridging" may occur where highly inclined sand grain movement may cause adjacent grains to diverge from contact with the shear plane. During the tangential movement and bridging, void spaces between particles are enlarged causing an increase in void ratio at these locations.

Figure 2.18. Theoretical trajectory of particles during shear (Newland and Allely, 1957).

A second theory proposed decades after the first looks at densely packed sand samples as if they are composed of granular columns supporting vertical stress (Oda and Hazama, 1998). During shear zone development these columns buckle to the side, loosing strength and

opening voids, as illustrated by Figure 2.19. This theory was tested by failing sand samples (0.206 mm mean diameter Toyoura sand and 0.527mm mean diameter Ticino sand) in a plane strain testing apparatus. A rectangular cross section and lubricated sample ends insured that a shear plane would form horizontally across the sample as opposed to bulging failure patterns consistent with many triaxial tests. Looking at thin sections microscopically revealed the presence of very large voids and void ratios of 1.01 to 1.13 (dependant upon whether thin sections were oriented in vertically or aligned with the shear plane, respectively) inside the shear zone for Toyoura sand and 1.09 to 1.14 for Ticino sand. These void ratios are particularly high compared to the maximum void ratio for the sands which were 0.97 and 0.96 respectively. The buckling of granular columns is proposed to cause the opening of void spaces. Orientation of particles in the shear band further supported this theory as originally horizontally oriented particles (as is typical of particles deposited under air pluviation) became oriented toward the direction of the shear plane.

Figure 2.19. Theoretical buckling of sand grain columns during shear (Oda and Hazama, 1998).

Previous studies (Oda and Konishi, 1982) captured the formation and buckling of such columns using photoelactically sensitive polyurethane rubber particles. These rod-like

particles of oval cross section were stacked by hand to form a 370mm wide by 370mm tall sample that was compressed to a residual stress state (compressed until failure was reached at which point less deviator stress is required to strain the sample). Photoelastic pictures were taken at different strains levels and revealed that up to the peak deviator stress, vertically oriented columns of particles supported the load. As residual stress states were reached load bearing columns became curved (buckled) and large void spaces opened.

Tensile reinforcement

The inclusion of one or more layers of earth reinforcement is a popular means to mechanically strengthen or stabilize soils. Initially metallic strips were used to reinforce soil. Corrosion problems and the development of polymer materials caused metallic strips to loose popularity and geosynthetics (or geotextiles) became predominant (Cazzuffi et al. 1993). Another attractive trait of geotextiles is that they have relatively low stiffness which corresponds well to soil movement (Haeri et al. 2000). The type of geotextile most suitable for structurally supporting coarse materials is geogrid. Geogrid is an open mesh of geosynthetic high-strength material in the form of multiple strands woven together or a continuous mesh (termed woven and unwoved geogrid, respectively). The openings within mesh units interlock with adjacent granular edges to resist movement relative to the soil and the geogrid.

Geogrid applications include column reinforcement (when encapsulating stone columns), spanning voids or soft soil areas, and foundation soil reinforcement. Foundation reinforcement will be discussed further. For foundation reinforcement, layers of geogrid are placed horizontally beneath a structure during placement of granular fill (frictional fill) on which the structure resides, see Figure 2.20.

With the inclusion of horizontal reinforcement soil gains the ability to provide additional confining forces. A soil that begins to strain will be resisted by the tensile strength of the geogrid refusing to deform. By increasing reinforcement (such as increasing the number of layers) strength also increases, volume change from strain is decreased (whether

contractive or dilative), and the strain level at failing stress increases (Cazzuffi et al. 1993). These benefits begin once strains have occurred great enough to mobilize tensile resistance in the geogrid. Tests have shown that sometimes two and a half to five percent strains may be needed to mobilize strength gains from reinforcement, increasing strain as confinement pressure increases (Cazzuffi et al. 1993). Strength gain, once mobilized, results in a consistently sized magnitude increase in shear strength, regardless of the state of confinement pressure (Jones, 1996) as shown in Figure 2.21. Tests of shear box tests with angled geotextile reinforcement revealed that this consistent strength gain is seen regardless of the orientation of reinforcement relative to the plane of shear stress development (Bauer and Zhao, 1993).

Figure 2.20. Placement of geotextiles reinforcement for a foundation setting.

Figure 21. Increase of shear strength due to tensile reinforcement raises the Mohr-Coulomb failure envelope.

In a foundation reinforcement situation the foundation load can shear the soil underneath itself or otherwise cause settlement to occur by straining, consolidating or compacting underlying soil. There are three failure modes that may occur in such a situation. The first failure mode consists of the foundation failing above reinforcement so-that the geotextile offers little reinforcement to the failure region as illustrated by Figure 2.22.a. This failure mode is the result of burying reinforcement too deep below the depth of the foundation. The second failure mode mobilizes stresses in the geotextile until it shears, allowing failure to occur through the reinforced soil as is shown in Figure 2.22.b. This failure mode is caused by not using enough layers or by not using a sufficiently thick geotextile. The third failure mode also mobilizes stresses in the geotextile but in this case the extents of the reinforcement pull out from the nearby soil allowing the geotextile to sag and the foundation to sink (Jones, 1996) as illustrated in Figure 2.22.c. This failure mode is allowed because an ample amount of reinforcement layers were not used or the pullout resistance of geotextile layers was not high enough. Pullout resistance of geogrids is a function of geosynthetics characteristics, soil properties (density, grain size distribution and moisture content), compaction procedure, and confining pressure (Farrag and Griffin, 1993).

Figure 2.22. Failure modes: a.) shear failure above reinforcement b.) shear failure through reinforcement c.) failure resulting from tensile pullout (Reproduced from Jones, 1996).

Tests to determine the effects of soil reinforcement commonly include shear boxes where soil is in line with the developed shear plane as exemplified by Farrag and Griffin (1993) and Bauer and Zhao (1993). These tests model geogrid pullout conditions for a single side of the geogrid. An ASTM standard (ASTM D6716, 2001) exists which describes the proper method to determine pullout resistance in geosynthetics.

Another test configuration is horizontal reinforcement placed in soil and later subjected to axial compression such as in a triaxial compression test as exemplified by Cazzuffi et al. (1993) and Haeri et al. (2000). These tests model a vertical loading situation. In reinforced triaxial compression tests the geotextile layers may only fail at very high confining pressures (dependent upon geotextile strength). Also since the soil mass has no adjacent soil to pull geogrid out from a pullout failure is not a possibility. Therefore failure is forced to occur in the soil between layers.

Portland Cement Pervious Concrete
Portland Cement Pervious Concrete uses a gap graded aggregate mix (missing most or all of the sand fraction) to produce a concrete that is porous and permeable. This concrete, which resembles a rice crispy bar of aggregates connected together with cement paste, has various benefits. These benefits include high permeability, noises reduction and economy.

The Clean Water Act and other Environmental Protection Agency (EPA) regulations require a reduction in storm water runoff from new construction as well as treatment of collected water. Storm water collection can occur within and below the pervious concrete layer, dampening the intensity of storm water runoff. Settlement of particulates during containment and percolation through coarse media filters larger particulates from the storm water. The percolation of surface water also aids traffic safety by reducing skids from hydroplaning.

Pervious concrete reduces vehicle noise. The main mechanism of noise dampening is altering the path length of reflected waves from their source. "Flat" concrete directs reflected sound waves in roughly the same vector to intercept a listener at roughly the same time. In contrast, the irregular surface of pervious concrete reflects waves from many various locations which in turn reach the listener at slightly different times, decreasing the intensity of reflected sound waves (Olek et al. 2003).

Pervious concrete may be an economical choice over ordinary concrete. The pervious concrete volume itself may be cheaper to purchase concerning materials and installation costs (NRMCA, 2004). This is not including the costs avoiding by not having to install water retention facilities as otherwise required by the EPA.

Some mix design examples are found in Table 2.2. These mixes have been found successful in resisting damage from cyclic freeze/thaw in either the lab or in the field. Mix designs in pervious concrete are a tradeoff of strength versus permeability. As void ratio increases, permeability increases exponentially, unit weight decreases linearly, and compressive strength decreases linearly (Schaefer et al. 2006). Another tradeoff is freeze/thaw durability. As void ratio increases, resistance to freeze/thaw damage is decreased (Schaefer et al. 2006).

Table 2.2. Example mix designs and properties.

Coarse Aggregate (pcy)	Fine Aggregate (pcy)	Cement (pcy)	Polymer emulsion (pcy)	Water (pcy)	Void ratio	28 day compressive strength (psi)	Permeability (in/s)	Reference source
2288	153	475	95	95	0.38	3770	NA	Beeldens, 2003
2700	0	578	0	156	0.34	1722	0.57	Schaefer, 2006
2700	0	525	52.5	115.5	0.19	3349	0.07	Schaefer, 2006
2500	168	571	0	154	0.18	3661	0.04	Schaefer, 2006
2570	0	300	106	166	0.25 to 0.32	NA	NA	NRMCA, 2004
2700	0	400	0	172	NA	NA	NA	NRMCA, 2004

One way to increase the strength of a mix design is to add more sand fraction to the mix, which in turn results in lowering the void ratio. Additives may also help the design's strength, especially towards freeze/thaw resistance. Examples of additives for pervious concrete include various liquid polymer emulsions (Beeldens et al. 2003), (Schaefer et al. 2006), water reducing admixtures, and air-entraining agents (Schaefer et al. 2006). These additives help the cement paste react more elastically to strain and therefore are less likely to suffer brittle failure.

CHAPTER 3. CHARACTERIZATION OF GRANULAR MOVEMENT UNDER TRIAXIAL COMPRESSION USING X-RAY COMPUTED TOMOGRAPHY

Abstract

Movement of sand grains subject to triaxial compression testing was characterized by calculating the translation and rotation of a discrete number of particles. Grains were marked using three lead solder points fixed to their extents. Cylindrical specimens were imaged three dimensionally using X-ray computed tomography (X-ray CT) at incremental loading steps to track the movement of marked particles. Samples having varying confining pressure and horizontal reinforcement were tested. Data reduction was completed using computer algorithms to calculate movement relative to orthogonally oriented vertical, radial, and tangential vectors. Findings include the quantification and comparison of particle translation and rotation and discussion of generalized movement trends.

Introduction

Much is unknown to date about how granular particles move during strain. Theoretical models of particle movement have been proposed by Newland and allely (1957) and Oda and Hazama (1998) which require verification. Both thin sectioning and X-ray CT techniques have been used to characterize sand grain movement behavior under axial strain by Alshibli et al. (2000), oda and Hazama (1998), Desrues et al. (1996) and Jang and Frost (2000) but only applied to characterizing the development of void spaces and/or the orientations of particles after test completion, not measuring the actual movements or rotations of particles. X-ray CT has been used to measure movements of plastic spheres under axial strain (Alshibli and Alramahi, 2006), but plastic spheres do not necessarily demonstrate the behavior of natural materials and the research approach was not adequate to accurately analyze particle rotations.

This experimental study was conducted to develop procedures applicable to characterize particle movements. Presented are analysis findings as well as the methodology developed. Using X-ray CT as a tool, both the directional movements and rotations of coarse grained sand grains were quantified at a discrete number of points throughout the volume of triaxially strained cylindrical samples. The research results are trialed methods to image and quantify particle movements as well as the results of three specimens tested under triaxial compression.

Background

As cohesionless samples are loaded under compression, shearing occurs inside the sample and particle movement occurs. Different means have been used to analyze shearing inside such samples. One method which allows scientists to view samples internally is X-ray computer tomography (X-ray CT).

X-ray computed tomography

X-ray CT measures the attenuation of X-rays as they interact with matter by using X-ray production and detection hardware. This information is analyzed and spatially assembled using computer algorithms. The result is three dimensional representations of attenuation coefficients throughout a sample's volume. Since these X-ray attenuation coefficients are proportional to material density, the imaged results of X-ray CT analyses may be interpreted as maps of density.

X-ray CT has been used to characterize geotechnical materials and is typified by analyses which differentiate void space from particle regions if resolution is sufficient. Such analyses include determining the grain size distribution of sand (Mokwa and Nielson, 2006) and the measure of crack apertures in sand cores (Sato et al. 2006).

If voxel (three dimensional pixel) sizes are large enough to encapsulate multiple particles then the intensity of these voxels is indicative of an averaged density within that space which may be correlated to porosity. One use of this type of analysis for geotechnical

engineering is in designating regions of dilation during the shearing of soil samples. Examples include visualizing regions of failure from laterally loading a model foundation pile (Pham et al. 2006) and delineating shear plane geometry in triaxially compressed fine sand samples (Alshibli et al. 2000).

Although shear regions may be spatially quantified by these studies, they do not characterize the motion of particles under the influence of shearing conditions. A project conducted by Alshibli and Alramahi (2006) used X-ray CT to track the movement of plastic spheres under compression. Each sphere had a hole drilled through its interior. The hole was used as a reference to determine bead rotation and the center of particles was also recorded to quantify directional movement. By conducting X-ray CT scans at consecutive loading steps the movement of beads under compression was characterized.

Particle movement

There are multiple theoretical models explaining the formation of shear bands. One such model proposed that particles move as grains independent of each other, rolling along shear planes to accommodate macroscopic strain (Newland and Allely, 1957). This theory rationalizes that in order for sand particles to move relative to each other, they would have to move roughly tangentially to one another. This model is illustrated in Figure 3.1 where particles A, B, and C are moving towards the right. Initial tangential movement is inclined upwards from the horizontal at angles θ_A, θ_B, and θ_C respectively. The tangential inclination of particle movement is at an initial maximum compared to that at further strains, explaining the initially high and subsequently declining resistances to deviator stress exhibited by strain softening behavior. In this model, particles are anticipated to move mostly independently of each other except that some "bridging" may occur where sand grain movement causes adjacent grains to diverge from the shear plane. During the tangential movement and bridging, void spaces between particles are enlarged causing an increase in void ratio at these locations.

A second theory proposed decades after the first looks at densely packed sand samples as if they are composed of granular columns supporting vertical stress (Oda and Hazama, 1998). During shear zone development these columns buckle to the side, loosing strength and causing dilation, as illustrated by Figure 3.2. This theory was tested by straining sand samples in a plane strain testing apparatus. Viewing thin sections revealed the presence of very large voids and void ratios inside the shear zone. Void ratios in this region were found to exceed that of the maximum void ratio (a negative relative density). Orientation of particles in the shear band further supported this theoretical model as originally horizontally oriented particles (as is typical of particles deposited under air pluviation) became oriented toward the direction of the shear plane.

Figure 3.1. Theoretical trajectory of particles during shear (Reproduced from Newland and Allely, 1957).

Figure 3.2. Theoretical buckling of sand columns during shear (Reproduced from Oda and

Hazama, 1998).

Although these two theories both seem intuitively plausible, the actual movement of
particles located in shearing zones has not been adequately characterized to determine if
either of these theories, or both, hold true and if so, under what conditions.

It has been found that soils of similar material and subjected to similar stress conditions can
behave very differently according to their initial densities. These effects have been revealed
using both X-ray CT (Desrues et al. 1996) and thin sectioning (Jang and Frost, 2000).
During triaxial strain, samples prepared loose of the critical state will contract and reach
critical void ratio rather homogenously throughout their volumes. Samples prepared dense
of the critical state will tend to dilate but the sample's entire volume will not reach critical
void ratio. Instead, critical void ratio exists in a discrete number of failure planes within the
sample's volume.

Experimental Study

Sample composition and preparation

Coarse granular material used as the soil medium in this experiment consisted of a uniformly graded crushed limestone. All limestone particles passed the No. 4 sieve but were retained on the No. 8 sieve (openings 4.75 mm and 2.36 mm respectively). Manual measurements using a caliper showed that of forty sand grains (the grains tracked in this paper to characterize particle movements) the mean average length to width ratio was 1.6.

Forty sand grains were chosen and at their extents were fixed three solder pieces. These sand grains were some of the larger particles which passed the No. 4 sieve and therefore made available more surface area for attaching solder pieces. The lengths and widths of the forty particles are provided in Table 3.1.

Table 3.1. Dimensions of solder marked crushed limestone particles.

Limestone Particle Dimensions					
particle	Length (mm)	width (mm)	particle	Length (mm)	width (mm)
1	10.0	6.3	21	8.5	5.0
2	11.6	5.3	22	7.8	5.4
3	7.1	6.5	23	10.7	5.6
4	9.5	5.4	24	8.1	6.4
5	9.1	7.5	25	8.4	5.3
6	8.3	5.7	26	8.9	6.9
7	9.2	4.9	27	9.1	5.7
8	8.3	6.0	28	11.5	6.3
9	11.4	5.3	29	10.0	6.6
10	7.6	5.3	30	10.7	6.1
11	11.1	5.9	31	7.8	6.0
12	10.5	6.4	32	9.1	5.3
13	10.3	6.3	33	10.2	5.3
14	12.3	5.6	34	10.5	5.6
15	8.1	6.4	35	9.1	6.0
16	9.1	5.6	36	8.0	6.0
17	10.2	5.8	37	11.3	5.4
18	11.5	6.0	38	7.2	5.5
19	11.6	6.5	39	9.5	6.5
20	9.2	6.0	40	8.0	5.4

Lead solder was chosen as a marking material due to its high density. The three solder pieces each had a distinguishing shape which allowed for their individual identification during the post processing of X-ray CT images. Points are designated as B, L, or T. B pieces are made from a 0.81 mm (0.032 in) diameter lead solder and have a greater length than that of pieces T but not of L. Pieces L and T are made from 0.38 mm (0.015 in) diameter lead solder. Solder points were manually placed roughly equidistant from each other around the perimeter of the grain and fixed to the surface using a commonly found glue which boasts having strength comparable to that of a large primate. No solder pieces became dislodged or loose during any of the testing procedures. Solder marked particles are illustrated in Figure 3.3 which depicts all such particles and zooms in to show examples of B, L and T pieces.

Figure 3.3. Solder marked particles used in analysis with zoom into B, L, and T solder piece examples. Ruler scale is in centimeters.

Cylindrical specimens tested had dimensions of 5.1 cm (2.0 in) diameter and 10.2 cm (4.0 in) height. They were subjected to triaxial compression under drained conditions. Each sample was compacted to a relative density of 93 percent. Sand was placed in four lifts, each containing ten randomly placed marked grains. Sand was tested in a dry state. One sample contained a layer of reinforcement consisting of a 4.8 cm (1.9 in) diameter, 0.32 cm (0.13 in) thick aluminum disk extending horizontally across the sample's mid-height. Specimens were vibrated manually by knocking on the specimen mold until the target density was achieved.

X-ray CT scans were taken at progressive strain intervals. The sample chamber needed to be moved from the loading frame to the X-ray CT sample positioner without altering confining pressure. Nitrogen gas was utilized to supply this pressure.

Three different samples were created. Variables differentiating samples are confining pressure and the presence of reinforcement inclusion. Variation in analysis is due to the strain increments at which scans were conducted. Samples are numbered one through three. Sample 1 was compressed at a confining pressure of approximately 27 kPa (3.9 psi) and was scanned at 0, 5%, 10% and 20% axial strain. Sample 2 was compressed at a confining pressure of approximately 14 kPa (2.1 psi), was scanned at 0, 5%, 10% and 20% strain, and contained a horizontally inclined aluminum disk at its mid-height. Sample 3 was compressed at a confining pressure of roughly 48 kPa (7.0 psi) and was scanned at 0, 2%, 4%, 6% and 8% strain.

X-ray computer tomography scanning
At regular strain intervals samples were scanned. The X-ray beam was produced at 100 kV and 5.0 mA and was found to provide adequate beam intensity and penetration. Projections were collected by a 10" by 17" CCD camera detector capable of 1920 x 2048 pixel resolution. To insure that consecutive scans would be unvaryingly positioned in the fan beam a base plate was machined which fit snugly around the triaxial chamber base. The base plate was equipped with a pin to use as an angular reference. Also, the sample

positioner was made to rotate exactly one complete rotation during CT data collection so scans always began at the same initial rotational reference. A high strength plastic material was used to create the chamber tie rods. Since the rods would be within the cone beam projection, steel rods would have cast shadows across the resulting CT data. These rods can be seen in Figure 3.4 alongside the X-ray production and detection hardware.

Figure 3.4. X-ray CT beam projection and detection hardware and sample staging

Volumetric X-ray CT files were created using a filtered back projection reconstruction technique which utilized an advanced cone beam reconstruction algorithm. Specifics about this reconstruction can be found in Kini (1994). Scans were reconstructed to have a cross sectional area of 640 x 640 voxels and a height of 360 voxels. These volumetric data files were reasonably sized for computer imaging (did not exceed available computer memory) and provided adequate resolution (could distinguish between three solder piece shapes). To demonstrate volumetric X-ray imaging, Figure 3.5 illustrates sample 2 as pictured using both a camera and X-ray CT at differing voxel intensity thresholds.

Figure 3.5. Images of sample 2 from a.) a camera image, b.) X-ray CT, and c.) X-ray CT with a voxel intensity threshold removing everything except lead solder.

Data collection and analysis

Post processing X-ray CT data began with recording the three dimensional coordinates all solder points found within scanned samples. The three dimensional coordinates of the center of each solder marker on each marked particle supply all inputs necessary to compute particle movement. These points were gathered by viewing volumetric X-ray CT files and manually determining which voxel best represents the central portion of solder points.

Once coordinates were collected, a movement analysis could be conducted. A computer algorithm was written to perform the numerous calculations. The first step in the program is to determine the location of particles, which is equal to the averaged three dimensional coordinates of the three solder points. As the location of sand particles move in incremental scans these coordinates change accordingly. Thusly, directional movement is quantified. The program computes directional movement in terms of vertical, radial, and tangential vector components as well as total magnitude (See Figure 1.6).

The next step in the program is to create vectors according to the relation of pieces B, L and T. These vectors will serve as benchmarks in computing particle rotation. One vector

travels from point B to L, designated as $\overline{BL}$. A second vector, $\overline{TM}$, passes through point T and is perpendicular to $\overline{BL}$. A third vector, $\overline{MX}$, is perpendicular to both $\overline{BL}$ and $\overline{TM}$ (See Figures 3.7 and 3.8). Next, six new vectors are created, each of which bisects one of the first three vectors. For instance, vectors $\overline{BL}$ and $\overline{TM}$ are bisected by vectors $\overline{BLTM1}$ and $\overline{BLTM2}$ as is shown in Figure 3.9. Now a total of nine vectors, whose inclinations are directly dependant upon the position of solder points, have been created (See Figure 3.10). Rotations occurring between loading phases can be computed referencing these benchmark inclinations.

Figure 3.6. Illustration of radial, tangential, and vertical directions relative to a particle and the specimen center as well as rotation around these vectors.

Figure 3.7. Representation of solder pieces fixed to the side of a sand grain.

Figure 3.8. Benchmark vectors in relation to solder pieces B, L and T.

Figure 3.9. Additional benchmark vectors which bisect vectors $\overline{BL}$ and $\overline{TM}$.

Figure 3.10. Complete array of nine benchmark vectors.

Total rotations as well as rotations around the tangential, vertical, and radial vectors as is relative to sand particle locations (θ, α and β respectively as shown in Figure 3.6) are computed.

The total rotation (Φ) from one unit vector inclination (a) to a new unit vector inclination (b) is equal to the inverse cosine of the dot product of these vectors (Equation 3.1). In this calculation, each reference vector may provide a different resultant. This is because for every rotation of a grain there is a plane on which rotation is maximized. The more closely a vector is aligned to this plane the closer the rotation of that vector will be to mimicking the rotation of the particle and likewise the rotation of that vector is greater than those vectors at further divergence from the rotational plane. Total particle rotation is considered the maximum total rotation of all reference vectors.

$$\Phi = \cos^{-1}(a \cdot b) \qquad (3.1)$$

Rotations θ, α and β are computed as the rotation of vector projections upon the plane perpendicular to the axis of rotation. This type of angular measurement was used by Alshibli and Alramahi (2006) to quantify rotation on a horizontal plane. Reference vectors which are most closely aligned to the reference plane are chosen to compute rotations on that plane.

As a caveat, all rotations and movements calculated in this analysis are the shortest path between two alignments or locations. Particles could travel back and forth and the movement would not be detected beyond the resulting movement between scanning intervals. Therefore, rotation to any point can be completely mimicked by an infinite number of other combinations of rotations.

Accuracy of Movement Quantification

The voxel representing the center of the solder pieces was determined manually by viewing volumetric X-ray CT files. Therefore, the author's accuracy in making this determination is a source of error. Also requiring discussion is the level of accuracy of the analysis approach which obtains rotation results from nine reference vectors.

During data collection the author recorded the positions of five particles twice to obtain a representation of data collection accuracy. X and Y data coordinates are rounded to the nearest voxel (which resides in a single pixel on a computer monitor) while Z coordinates are recorded to the nearest half voxel since vertical resolution is 2.55 times the dimension of X and Y resolution. The variation in determining these points is shown in Table 3.2 as well as the resulting change of particle position and particle inclination. Based on these results, angular rotations are easily varied by several hundredths of a radian and particle translation (directional movement) by tenths of a millimeter. Author's average X and Y voxel determination erred by an average of 0.50 voxels while Z coordinate input erred by an average of 0.13 voxels, corresponding to spatial distances of 0.057 mm and 0.039 mm respectively.

Table 3.2. Rotation and translation causes by input errors during data collection.

	Particle number				
	1	2	3	4	5
X voxel diff. (B,L,T)	0, 0, 0	1, 1, 0	2, 0, 1	1, 1, 0	2, 2, 0
Y voxel dif. (B,L,T)	0, 0, 0	1, 1, 0	0, 0, 0	0, 1, 0	0, 1, 0
Z voxel dif. (B,L,T)	0, 0, 0	0.5, 0, 0	0, 0, 0	0, 0, 0	0.5, 0.5, 0.5
Rotation (rad)	0	0.015	0.048	0.027	0.037
Translation (mm)	0	0.12	0.11	0.09	0.22

The rotation realignment of a vector aligned away from a plane of maximum rotation is less than the realignment of a vector existing on said plane. A vector existing on the plane of maximum rotation is considered by this research approach to be representative of true particle rotation. The relationship of the rotation measured from a reference vector and the true particle rotation is a function of the total rotation magnitude and the divergent angle between the plane of maximum rotation and the reference vector. Using nine reference

vectors as was done in this research, the largest possible angular divergence from the plane of maximum rotation is 0.393 radians (22.5 degrees), half of the maximum angle between reference vectors. The relationship of divergent angle versus the ratio of angular rotation measured from a reference angle over true rotation (rotational fraction) is presented in Figure 3.11. The largest single loading phase particle rotation (excluding a particle which rolled off the side of the porous stone in sample 2) is equal to 0.637 radians, recorded during the 10% strain to 20% strain stage of loading sample 2. Referencing Figure 3.11, the maximum error from such a rotation, assuming maximum reference vector divergence from the plane of maximum rotation, is approximately 7.9%.

Figure 3.11. Rotational fraction versus true rotation at various angles of divergence from the plane of maximum rotation.

Further evaluation of using nine vectors to calculate total rotation is presented in Figure 3.12 which was created using the results of the conducted research. In Figure 3.12 averaged rotations calculated using different numbers of reference vectors are illustrated for each of the three samples analyzed where averaged rotations are equal to the mean average rotation of all marked particles in all load steps. When one vector is used the rotation in Figure 3.12 is equal to the averaged resulting rotation of all particles in all loading phases by using vector $\overline{BL}$ alone. When two vectors are used both $\overline{BL}$ and $\overline{TM}$ are analyzed and the

maximum result of these individual vectors is used. The third point adds $\overline{MX}$. This continues until all nine benchmark vectors are included. Figure 3.12 shows that after five or six reference vectors are used, the averaged rotation becomes relatively consistent.

Figure 3.12. Averaged benefit of using additional reference vectors to compute rotation.

Figure 3.12 presents averaged results. The effect to individual rotation measurements are presented in Figure 3.13. This histogram's horizontal axis is composed of percentage ranges. These percentages are the increases in total rotation resulting from using 9 reference vectors instead of 1, 3, or 6 vectors. The histogram's vertical axis counts the number of instances in which total particle rotations increased within percentage ranges. Results presented in Figure 3.13 depict each marked particle at each scanned loading phase for sample 1, resulting in a sum of 160 counts. Based on Figure 3.13, it is not uncommon for rotations to increase by over 100 percent when 9 vectors are used instead of only 1. By using 3 reference vectors, resulting rotations are mostly within 2 percent of the values calculated using 9 vectors. At 6 reference vectors, the increase in total rotation from using 9 vectors was never over 6 percent.

Figure 3.13. Benefit of using additional reference vectors to compute individual particle rotations.

Results and Discussion

The three samples subjected to triaxial compression can be viewed in Figure 3.14 at the maximal strain state at which X-ray CT scans were conducted. The deviator stress versus strain responses from the samples are shown in Figure 3.15. All samples exhibited strain softening behavior and peak deviator stress in the range of 4 to 6 percent axial strain.

a.)   b.)   c.)

Figure 3.14. X-ray CT produced images of a.) sample 1, b.) sample 2 and c.) sample 3 at their maximal strain states.

Deviator Stress Vs. Strain

Figure 3.15. Compression test results

Particle directional movement

Particle directional movement (translation) was characterized for incremental stages of strain. The radial and vertical positions of marked particles at increasing loading stages are shown in Figure 3.16 for each sample. Radial positions are in respect to the specimen central axis while vertical positions are relative to the sample base. Particle positions at each loading stage are represented. Points belonging to the same particle are connected with lines. It should be noted that relativity to the sample base gives an initial illusion of greater vertical movement at the sample top, while actual sample strain may be rather homogenously distributed. Also, since samples are being shortened with increasing strain, the initial particle position is at the highest point and consecutive loading increments typically move a particle closer to the sample base. The initial location of horizontal reinforcement in sample 2 is depicted as a gray line. It can be seen in Figure 3.16 that particle motion becomes more oriented in the radial direction towards the mid-height of samples or, in the case of sample 2, at the mid-height of the region between the reinforcement and the sample top or base. These radial motion concentrated areas correlate well spatially to the regions of lateral deformation seen in Figure 3.14.

Figure 3.16. Solder marked particle positions at each loading increment for a.) sample 1, b.) Sample 2 and c.) sample 3.

To synthesize the movement within motion concentrated areas the mean average translation of particles in these regions are illustrated in Figure 3.17 for each strain stage. The particles used to create Figure 3.17 are documented in Table 3.3 where particle 1 is the

lowest particle in each sample and particle 40 the highest. It can be seen that movement within sample 1 becomes increasingly oriented in the radial direction after the 1st strain interval. Sample 2 exhibited similar results in its upper half but varies in its lower half. From 5% to 10% strain (2nd strain interval) very little vertical strain is seen in the sample's lower half showing that vertical strain in the sample is occurring predominantly above the horizontal reinforcement. The low strain increments analyzed in sample 3 show that once 2% strain has been reached, the averaged orientation of particle motion seems to become predictably linear with consistent radial velocity. By referencing Figure 3.15, at 2% strain the sample's peak strength has not yet been reached but the stress strain modulus has begun weakening from its original value.

To further illustrate granular radial and vertical directional movement (translation) and present tangential translation, figures have been prepared which show radial and tangential translations alone as well as combined radial and vertical translations. Radial and vertical particle translations are visually quantified using vectors which parallel particle translation and are weighted in length to the respective magnitude of particle translation. These vectors extend from points corresponding to particle starting positions. For each sample, the movements occurring in the first test stage as well as the total movements throughout all test stages are illustrated (Figures 3.18 through 3.20). Tangential motion, since it cannot extend upon the chosen reference plain, is depicted as a radial extension. In order to aid in visualizing small particle movements, a movement multiplier has been applied to make a weighted vector length in some translation figures. The averaged tangential movement of particles within the motion concentrated areas (See Table 3.3) and within the entire samples is displayed in Table 3.4.

From Figures 3.18 through 3.20 and Table 3.4 it becomes apparent that tangentially inclined particle translation accelerates once the sample strains beyond their initial loading stage and that tangential motion generally collects in the same regions as radial movement concentrations. Exceptions include sample 2 which shows two strain increment instances where tangential motion was not concentrated centrally.

Figure 3.17. Averaged movement of motion concentrated particles in a.) sample 1, b.) sample 2 and c.) sample 3.

Table 3.3. Particle numbers used to create Figure 3.17 and Table 3.4.

	Particle number range	
	Lowest	Highest
Sample number 1	15	30
Sample No. 2, upper	25	35
Sample No. 2, lower	5	15
Sample number 3	15	30

Figure 3.18. Translation at marked particle locations of sample 1 for a.) radial and vertical translation at 5% strain, b.) radial translation along at 5% strain, c.) tangential translation at 5% strain, d.) radial and vertical translation at 20% strain, e.) radial translation along at 20% strain, and f.) tangential translation at 20% strain.

Figure 3.19. Translation at marked particle locations of sample 2 for a.) radial and vertical translation at 5% strain, b.) radial translation along at 5% strain, c.) tangential translation at 5% strain, d.) radial and vertical translation at 20% strain, e.) radial translation along at 20% strain, and f.) tangential translation at 20% strain.

Figure 3.20. Translation at marked particle locations of sample 3 for a.) radial and vertical translation at 2% strain, b.) radial translation along at 2% strain, c.) tangential translation at 2% strain, d.) radial and vertical translation at 8% strain, e.) radial translation along at 8% strain, and f.) tangential translation at 8% strain.

Table 3.4. Averaged tangential movement in motion concentrated sample areas.

| Sample | Strain interval (%) | Tangential movement (cm) | |
		Concentrated area	Total sample
No. 1	0 - 5	0.037	0.027
	5 - 10	0.064	0.039
	10 - 15	0.065	0.049
	15 - 20	0.042	0.038
No. 2 upper	0 - 5	0.033	0.035
	5 - 10	0.067	0.055
	10 - 20	0.305	0.266
No. 2 lower	0 - 5	0.027	0.021
	5 - 10	0.046	0.036
	10 - 20	0.076	0.079
No. 3	0 - 2	0.012	0.009
	2 - 4	0.008	0.007
	4 - 6	0.026	0.017
	6 - 8	0.025	0.015

To compare radial to tangential sand grain translation, Figure 3.21 presents their ratio for each marked particle as distributed vertically throughout the specimens. Particles near sample ends seem to exhibit ratios which can be either greater or less than 1. However, as particles move closer to the sample mid-height the radial component becomes increasingly dominant and ratios less than 1 are rare. In the case of sample 2, the aluminum reinforcement acts like a sample end.

Particle velocity in radial and tangential directions seems to grow as axial strain progresses after the initial loading stage. This concept is originally presented in Figure 3.17 and Table 3.4. This effect is found to be dependant upon particle location within the specimen as shown in Figure 3.22. This figure plots initial stage horizontal movement ratio (R_i) versus particle height as defined in the following equation:

$$R_i = \frac{\Delta_{H1}}{\Delta_{HT}} \tag{3.2}$$

where Δ_{H1} is the horizontal particle translation resulting from the first loading stage and Δ_{HT} is the horizontal particle translation total of all loading stages. Horizontal movement is the hypotenuse formed by orthogonal radial and tangential translation components. Since the primary loading stage causes a quarter of the cumulative axial strain of all loading stages, a ratio of ¼ would indicate steady horizontal velocity during subsequent compression steps. Ratio values of less than ¼ would be a sign of acceleration after the initial load step and visa versa. Regions of consistent values less than ¼ have been detailed in Figure 3.22. Bulging regions viewed in Figure 3.14 include the upper portion of Sample 1 the upper half of sample 2 which correlate well to these detailed regions.

Horizontal movement within the triaxially loaded samples was found to be dependent upon confining pressure. Samples with greater confining pressure exhibited less horizontal movements. This relationship is presented in Figure 3.23 where the average lateral movement per vertical strain ($\Delta_{\varepsilon v}$) is plotted versus confining pressure where $\Delta_{\varepsilon v}$ is defined by the following formula:

$$\Delta_{\varepsilon v} = \frac{\sum_{1}^{n}\Delta_{HT,i}}{n \times \varepsilon} \tag{3.3}$$

where n is the number of particles analyzed (40), $\Delta_{HT,i}$ is the total horizontal translation of particle i experienced in all loading stages, and ε is the sample strain after its final loading stage (as a fraction). Since lower confining pressures should theoretically result in more dilatant behavior according to critical state soil mechanics, the relationship of greater horizontal movement with lower confining pressure makes sense.

Figure 3.21. Ratio of radial over tangential translation as vertically distributed within a.) sample 1, b.) sample 2 and c.) sample 3.

Figure 3.22. Ratio of horizontal particle movement during in the first loading stage over the total of all loading stages within a.) sample 1, b.) sample 2 and c.) sample 3.

Figure 3.23. Horizontal movement versus confining pressure.

Sand particle rotation

Particle rotation has been visually quantified by grouping ranges of rotation magnitudes at regular intervals, or radian increments, and assigning each range a different symbol. These symbols are placed at the radial and height coordinates of the particles as they existed at the beginning of the test. Figure 3.24 is a legend to interpret rotation figures wherein rotation units are in radians. Total rotations as well as rotations around the tangential, vertical, and radial vectors (θ, α and β respectively as illustrated in Figure 3.6) are presented in Figures 3.25 through 3.27.

In Figures 3.25 through 3.27 it appears that marked particles which show little rotation can be very closely located, in terms of height and radial distance, to particles showing much larger rotation. Also notable is that of the three rotation planes analyzed, no single one seems to dominate. This shows the random nature of particle rotation. However, patterns do develop. Rotations in sample 1 seem to clearly designate a failure region which mirrors that seen in Figure 3.14.

Due to the bisection of sample 2 with reinforcement, creating two regions of shear development, the effective loss of resolution makes visualizing a distinct failure area within a sample half difficult. Even so, high rotation regions do seem to be found near the halves' central regions where sample bulging is greatest. High rotations are also found near the far

lateral extent of sample ends. This is due to these sand grains slipping outside of the triaxial loading chamber pedestal or base. Slipping grains can be seen in Figure 3.5.

Sample 3 does not have an apparent pattern to particle rotations, even up to 8% sample strain. This may be due to the additional pressure confining the sample. Additional pressure causes a less dilatant failure mode and a lessened predisposition to weaken in disturbed (dilated) areas, allowing additional disturbance and subsequent rotation to concentrate in those areas. Sample 3's less dramatic strain softening behavior can be seen in Figure 3.15. It may also be the case that since the accuracy of rotation calculation, probably several hundredths of a radian, is approaching the magnitude of rotations seen by the particles during 2% axial strain intervals, the results could be only appearing random.

Radian increment	
△	0.00 to 1x range
◇	1x to 2x range
■	2x to 3x range
●	3x to 4x range
◆	4x to 5x range
⊞	>5x range
— —	Original sample extent
⌇⌇⌇	Aluminum plate
••••••	Failure zone extents

Figure 3.24. Legend for interpreting particle rotation Figures 3.25 through 3.27.

Figure 3.25. Rotations at marked particle locations of sample 1 for a.) θ at 5% strain, b.) α at 5% strain, c.) β at 5% strain, d.) total rotation at 5% strain, e.) θ at 20% strain, f.) α at 20% strain, g.) β at 20% strain, and h.) total rotation at 20% strain.

Figure 3.26. Rotations at marked particle locations of sample 2 for a.) θ at 5% strain, b.) α at 5% strain, c.) β at 5% strain, d.) total rotation at 5% strain, e.) θ at 20% strain, f.) α at 20% strain, g.) β at 20% strain, and h.) total rotation at 20% strain.

Figure 3.27. Rotations at marked particle locations of sample 3 for a.) θ at 2% strain, b.) α at 2% strain, c.) β at 2% strain, d.) total rotation at 2% strain, e.) θ at 8% strain, f.) α at 8% strain, g.) β at 8% strain, and h.) total rotation at 8% strain.

Particle rotation showed a strong correlation to confining pressure. As was the case with the translation comparison (Figure 3.23), the averaged quantities were divided by sample strain to account for varying total strain magnitudes. The results are shown in Figure 3.28 where the average rotation per vertical strain ($\Phi_{\varepsilon v}$) is defined as follows:

$$\Phi_{\varepsilon v} = \frac{\sum_1^n \Phi_i}{n \times \varepsilon} \qquad (3.4)$$

where n is the number of particles analyzed (40), Φ_i is the total rotation of particle i experienced in all loading stages, and ε is the sample strain after its final loading stage (as a fraction). Figure 3.28 shows increasing particle rotation with increasing sample confinement. It could be reasoned that as samples become more contractive with increasing confining pressure there is greater re-orientation of sand particles.

Figure 3.28. Averaged rotation versus confining pressure.

Rotation comparisons to translation

Marked particle translation and rotation characteristics have been quantified. Comparisons between the two provide proportional relativity. Rotations are compared to translations by assigning rotations a resultant rotational movement. Rotational movement (Δ_R) of a particle is defined by the following equation:

$$\Delta_R = \frac{\frac{1}{2}(L_{ave} + W_{ave})}{2} \times \Phi \qquad (3.5)$$

where L_{ave} is the average length dimension of all 40 solder marked particles, W_{ave} is the average width dimension of all 40 solder particles, and Φ is equal to the total rotation of a particle in radians. The value of half the average length plus the average width is equal to 0.77 cm for particles used. By dividing rotational movement by horizontal movement a comparative ratio of rotation to directional movement is made. This ratio is termed the horizontal rotation motion ratio. The horizontal rotation motion ratios of each particle averaged throughout all loading stages are presented in Figure 3.29. By averaging these rotation motion ratios for every particle in a sample, the values presented in Table 3.5 are produced.

Figure 3.29. Horizontal rotation motion ratios for each particle in samples 1, 2 and 3 throughout all loading stages, ordered from lowest to highest ratio value.

Table 3.5. Averaged horizontal rotation motion ratios for each sample.

Sample	Rotation motion ratio
No. 1	0.62
No. 2	0.49
No. 3	1.13

Figure 3.29 and Table 3.5 propose that particle rotations would move sand grains an amount significantly less than calculated horizontal movements in samples 1 and 2. Rotation induced movements in sample 3 are slightly higher than horizontal movements. A pattern developed is that samples with greater rotation motion ratios also have greater confining pressure. This is likely the combination of effects presented in Figures 3.23 and 3.28 which show that increasing confining pressure reduces lateral movement and increases particle rotation.

Conclusions

Conclusions are made based upon the limited amount of samples and analyses provided by this research project. Further research should be performed before the following conclusions presented in this paper are determined repeatable and predictable:

1.) A nine vector approach referencing three discernable points on a particle and applied using the methodology presented in this paper can adequately characterize particle translations and rotations, although the accuracy of data collection may be problematic when low strain interval loading stages are used due to the small particle movements occurring therein.

2.) The radial component of horizontal translation becomes more dominant in bulging regions within samples. Radial translations also accelerate with axial strain in the bulging regions. Radial acceleration is seen to a point prior to sample peak strength after which particle horizontal velocity becomes relatively constant.

3.) Tangential translations collect in the same regions where radial translations collect. Tangential translation accelerates to a consistent velocity after peak strength is achieved.

4.) Horizontal translations are greater in samples subjected to lower confining pressures.

5.) Particles showing little rotation can be very closely located, in terms of height and radial distance, to particles showing much larger rotations.

6.) Of the rotation planes analyzed (θ, α and β) no single one seems to be dominant in any of the samples.

7.) Rotations are concentrated in bulging regions within samples and are greater in samples subjected to higher confining pressures.

8.) Rotational movements are roughly equal to horizontal translations in the highest confining pressure sample 3. As confining pressure reduces, the ratio of rotational movement to horizontal translation lessens.

9.) High rotation movement ratios are seen in bulging regions within samples with exception of the high confinement, low strain interval sample 3.

Research Recommendations

Future work in this area would be useful in validating or contradicting the conclusions made in this research. This research was developed with the intention of presented a methodology for an outcome and the results were an afterthought. The methods used in this paper can surely be adapted to characterize particle movements in a number of modeled geotechnical engineering applications as well as applications outside the field. Through conducting this research project, experience has been gained. The following advice is presented to future researchers:

1.) When a number of marked particles are located closely to one another it can become very time consuming to determine which markers correspond to which particles. Therefore:

a. Using larger samples to distribute marked particles within will ease analysis as well as allow more marked particles to be present. Powerful X-ray production equipment and large detectors screens, such as that used in this research, can easily accommodate significantly larger specimens.

b. Sample preparation means that results in a more homogenous distribution of marked particles will ease analysis as well as give a more inclusive representation of all sample regions.

2.) Smaller particle marker pieces would make coordinate determination easier. Those used in this research often spanned numerous horizontal slices and determining the central slice was often tedious. The resolution obtained in the analysis of this research and the quality of resulting images could easily accommodate smaller lead solder pieces.

3.) Although in theory the adjacent edges of high density markers could be used as individual marker points, it is advised that this approach is not attempted as the edges of lead solder points often become blurred and the exact extent of the lead segments become objective.

4.) Analyzing the three samples used for this project required collecting over 5,000 coordinate components. A computer application to collect data point location would ease analysis greatly. This, however, would be subject to great difficulties due to the blurring of high intensity inclusions and the extreme amount of accuracy required to reliably quantify rotations in the range of hundredths of a radian.

References

Alshibli, K.A., S. Sture, N.C. Costes, M.L. Frank, M.R. Lankton, S.N. Batiste, R.A. Swanson, 2000. "Assessment of Localized Deformations in Sand Using X-Ray Computed Tomography." Geotechnical Testing Journal, Vol. 23, No. 3.

Alshibli, K.A., B.A. Alramahi, 2006. "3D Visualization of Shearing in Granular Materials." ASCE, GeoCongress 2006.

Desrues, J. R. Chambon, M Mokni, F. Mazerolle, 1996. "Void Ratio Evolution inside Shear Bands in Triaxial Sand Specimens Studied by Computed Tomography," Geotechnique, Vol. 46.

Jang, D.J., D.J. Frost, 2000. "Use of Image Analysis to Study the Microstructure of a Failed Sand Specimen," Canadian Geotechnical Journal, Vol. 37.

Kini, V.R., 1994, "Tomographic Inspection System Using X-rays," Iowa State University, Master of Science thesis.

Mokwa, R.L., B. Nielson, 2006. "Nondestructive Measurements of Soil Geotechnical Properties Using X-Ray Computed Tomography." ASCE, GeoCongress 2006.

Newland, P.L., B.H. Allely, 1957. "Volume Changes in Drained Triaxial Tests on Granular Materials." Geotechnique, Vol. 7.

Oda, M., H. Hazama, 1998. "Microstructure of Shear Bands and Its Relationship to the Mechanism of Dilatency and Failure of Dense Granular Soils." Geotechnique, Vol. 48, No. 4.

Pham, K.D., J. Otani, Y. Watanabe, T. Makunoki, 2006. "Application of X-Ray CT on Boundary Value Problems in Geotechnical Engineering – Research on Ground Failure Due to Lateral Pile Loadings." ASCE, GeoCongress 2006.

Sato, A., D. Fukahori, A. Takebe, A. Sawada, D. Sugawara, 2006. "Analysis of Open Crack in Rock by X-Ray CT." ASCE, GeoCongress 2006.

CHAPTER 4. PORE CHARACTERIZATION OF PORTLAND CEMENT PERVIOUS CONCRETE USING X-RAY COMPUTED TOMOGRAPHY

Abstract

The analysis of pore spaces in geomaterials is important to understanding permeability. In this paper, the shape, distribution, and continuity of voids in Portland Cement Pervious Concrete (PCPC) were mapped using X-ray Computed Tomography (X-ray CT). Volumetric files produced from X-ray CT were post processed to remove beam hardening artifacts and analyzed to characterize pore spaces. Void space continuity was quantified with a continuity index parameter, and X-ray CT slices were processed to document the shape and size of void areas therein. Results of four mix designs are plotted against lab permeability measurements to show their respective relationships. Mapping void space revealed that samples prepared in cylinders show a layer of higher void ratio at their extents. The X-ray CT testing and data analysis techniques developed for this research and described in this paper have broader applications to other geomaterials.

Introduction

Pore space quantity, size, and continuity affect water permeation through geomaterials. Permeability is an important characteristic of Portland Cement Pervious Concrete (PCPC), a concrete mix designed to have large void content due to a reduction of sand fraction. It is therefore important to develop procedures to analyze pore characteristics of PCPC.

Much research has been conducted to develop mix designs of Portland Cement Pervious Concrete (PCPC) including that by Beeldens et al. (2003), the National Ready Mixed Concrete Association (2004), and Schaefer et al. (2006). Research in PCPC has typically been catered to testing strength, resistance to freezing and thawing cycles, and/or permeability. This research project is meant to analyze pore space quantity, size, and continuity within PCPC samples. In order to characterize this tortuous system of voids, new

methods and procedures were developed to enable X-ray computed tomography to spatially model PCPC specimens. Methods and procedures developed process X-ray data to remove unwanted artifacts and systematically apply analysis procedures. The resultant research approach has relevance to further PCPC research as well as other geomaterials research applications.

Background

X-ray Computed Tomography

X-ray computer aided tomography (X-ray CT) measures the attenuation of X-rays as they interact with matter by using X-ray production and detection hardware. The result is three dimensional maps of material density throughout a sample's volume. However, X-ray CT can produce erroneous representations of attenuations coefficients, and requires special testing and analysis techniques to maximize image quality. Erroneous effects corrected for this analysis include beam hardening, ring artifacts, and noise. For a more complete description of X-ray CT see Barrett and Swindell (1981) or for a description of the system used in this analysis see Kini (1994).

Portland Cement Pervious Concrete

Due to Environmental Protection Agency regulations such as the Clean Water Act, storm water mitigation has become an increasingly important issue. Storm water collection can occur within and below PCPC parking lots, dampening the intensity of storm water runoff and allowing particulate settlement during containment. Other benefits to PCPC include noise reduction (Olek et al. 2003) and draining surface water to prevent hydroplaning.

Portland Cement Pervious Concrete consists of an open-graded concrete mix design with little or no sand fraction (Schaefer et al. 2006). This allows rapid water permeation through continuous void channels. Strength of PCPC can be increased by adding more sand fraction to the mix. As sand is added, the void ratio is lowered and permeability is reduced. Thusly, strength can be inversely related to permeability. It is therefore important to understand the

means by which permeability is developed within PCPC samples so optimally strong yet pervious concretes may be developed.

Beeldens et al. (2003), the National Ready Mixed Concrete Association (2004), and Schaefer et al. (2006) have conducted research to develop mix designs for PCPC. Example mix designs developed by these research projects are presented in Table 4.1. Although pore characteristic such as void ratio and permeability are research factors, measurements of pore space size, shape, and continuity were not attempted.

Table 4.1. Example mix designs and properties.

Coarse Aggregate (pcy)	Fine Aggregate (pcy)	Cement (pcy)	Polymer emulsion (pcy)	Water (pcy)	Void ratio	28 day compressive strength (psi)	Permeability (in/s)	Reference source
2288	153	475	95	95	0.38	3770	NA	Beeldens, 2003
2700	0	578	0	156	0.34	1722	0.57	Schaefer, 2006
2700	0	525	52.5	115.5	0.19	3349	0.07	Schaefer, 2006
2500	168	571	0	154	0.18	3661	0.04	Schaefer, 2006
2570	0	300	106	166	0.25 to 0.32	NA	NA	NRMCA, 2004
2700	0	400	0	172	NA	NA	NA	NRMCA, 2004

Experimental Study

PCPC samples

Four PCPC mix designs were analyzed. These samples performed well in terms of freeze/thaw resistance during the first two phases of a research project at Iowa State University (Schaefer et al. 2006). Mix designs, designated here as mix numbers 1 through 4 with corresponding samples 1 through 4, were reported in the results of that research project as having those properties found in Table. 4.2. Samples used in permeability tests were 15.2 cm (6 in) tall and 7.62 cm (3 in) in diameter. Samples were each cut roughly in half vertically after permeation and one of the halves was used for this X-ray CT analysis, see Figure 4.1. These samples, now roughly 7.4 cm (2.9 in) tall, were quartered (Figure 4.2) and crushed to conduct volumetric analyses on sample quarters.

Table 4.2. Properties of mix designs used in analysis (Schaefer et al. 2006).

Mix	Aggregate type	Aggregate size	k
1	Limestone	9.5 mm (3/8 in)	14 mm/s (0.57 in/s)
2	River gravel	4.75mm (3/16 in)	1.8 mm/s (0.071 in/s)
3	River gravel	4.75mm (3/16 in)	1.1 mm/s (0.043 in/s)
4	River gravel	4.75mm (3/16 in)	2.5 mm/s (0.098 in/s

Figure 4.1. PCPC sample mixes 1 through 4 from left to right. Samples are approximately 3.0 inches in diameter and 2.9 inches tall.

Figure 4.2. PCPC sample samples 1 through 4 after cutting into four sections. Sample quarters are approximately 3 inches in diameter and 0.8 inches thick.

X-ray CT Scanning

PCPC samples illustrated in this report were scanned to produce volumetric files of 640x640x310 voxel size, resulting in cross sectional resolution of 0.15 mm (0.060 in) and vertical resolution of 0.30 mm (0.012 in). Scans were conducted at 130 kV and 0.11 mA initially for continuity analysis but later rescanned at 130 kV and 0.45 mA prior to void ratio distribution and pore size/shape analyses. A 0.005 inch copper filter was placed over the aperture of the X-ray tube to help attenuate low energy X-rays which cause beam hardening effects. Data was collected by sending an X-ray cone beam through the specimen casting X-ray projections onto a flat amorphous silicon detector. Four frames from each back projection were averaged to reduce noise. Cone beam reconstruction software was utilized to reconstruct X-ray CT files using a filtered back projection approach. Due to the material density contrast between concrete and air voids and the large size of voids found therein, X-ray CT is well suited to image PCPC. For illustrative purposes, sample regions from volumetric data sets may be viewed in Figure 4.3. X-ray CT produced cross sections are compared to pictures of samples taken after scanned specimens were cut in half vertically (Figures 4.4 through 4.7)

Figure 4.3. Illustrative view of X-ray CT volumetric files for each sample 1 through 4 with both void and solid regions together (sample #.1), Solid regions only (sample #.2), and void regions only (sample #.3).

Figure 4.4. Saw cut surface (left) and X-ray CT produced cross section (right) showing void, aggregate, and cement phases in sample 1.

Figure 4.5. Saw cut surface (left) and X-ray CT produced cross section (right) showing void, aggregate, and cement phases in sample 2.

Figure 4.6. Saw cut surface (left) and X-ray CT produced cross section (right) showing void, aggregate, and cement phases in sample 3.

Figure 4.7. Saw cut surface (left) and X-ray CT produced cross section (right) showing void, aggregate, and cement phases in sample 4.

As previously stated, scans were repeated prior to void ratio distribution and pore size/shape analyses. This was due to an erroneous gradient of voxel intensities within the volumetric X-ray CT files. It was found that somewhere during data collection or digital cone beam reconstruction an artifact was introduced. The artifact was identified as a blurring of lower intensity regions into adjacent higher intensity regions (void space intensities into concrete sample region intensities). This blurring occurred on the very outside boundaries of the sample as illustrated in Figure 4.8 (sample's top) and Figure 4.9 (sample's radial extents). It was found that these issues were resolved by stacking concrete above and below the samples (removing the air/concrete boundary) and by incasing the sample in a ceramic tube (causing the effect to occur in the ceramic tube).

Figure 4.8. Voxel intensity along an X-ray image reference line depicting an erroneous voxel intensity gradient at the sample top (original scan of mix 3).

Figure 4.9. Horizontal slices from both unenclosed (a) and encapsulated (b) X-ray CT scans of mix 4 exemplifying that the ceramic tube's presence remedies the intensity smearing of external aggregates.

Beam hardening trend removal

The beam hardening effect introduces an erroneous voxel intensity trend which causes voxel intensity to be larger (scan appears to have denser material) at further distances from the sample's center. To conduct a quantitative analysis using voxel intensities this artifact needed to be removed. This was done by mimicking the shape of the artifact using filters and subtracting that shape from the shape of the original data. All filtering processes were completed using POD Version 1.0 (computer program referenced in the References section of this chapter) which can be sourced for more thorough assessment of filtering techniques

discussed here. Beam hardening trend removal was completed according to the following eight computational steps:

1. Export individual CT slices from the original X-ray CT volumetric file. Each slice was analyzed separately. The X-ray CT produced slice of the first sample analyzed (the base of sample 4) is illustrated here with a reference line running through its center (Figure 4.10). The corresponding intensity values along the reference line are plotted below the image and void, solid, and ceramic casing features are noted with arrows.

Figure 4.10. Original X-ray CT cross sectional image with intensity values found along a centrally located reference line.

2. Make an image which is composed of the average of horizontal, upper limit windows. These windows are pixels in a line which are all equal to the highest point on the original image in the area on which they reside. Windows are assigned a length and all combinations are applied to the image so-that each pixel has many windows which reside on their space to average together. These averaged pixel lines, or window filters, effectively smear high regions

horizontally (Figure 4.11). It is important to choose a window length that can bridge void spaces so the intensity trend of the solids phase is mimicked, not the intensity contrast of the original image. It was found that a window length of 80 pixels worked well in this analysis and was used on all samples.

Figure 4.11. Effect of a horizontal window filter on the original image with intensity values found along a centrally located reference line.

3. Repeat step two using vertically inclined windows (Figure 4.12).
4. Add the images produced in steps two and three together and filter the produced image with a median area filter (Figure 4.13). This filter is two dimensional and finds the median value of pixels found within the extents of a prescribed size area. This median value is assigned to the centrally located pixel for the area. An area of 15x15 pixels was found adequate in this analysis. A median filter was found to work better than a mean filter because abnormally high or low intensity values, such as those caused by ring artifacts, are not averaged into the produced image.

Figure 4.12. Effect of a vertical window filter on the original image with intensity values found along a centrally located reference line.

Figure 4.13. Effect of a median area filter on the sum of images found in Figures 4.11 and 4.12 with intensity values found along a centrally located reference line.

5. Assign very high pixels values (well above the intensities of pixels found previously in the image) to those pixels found on the original image which are below a threshold value. This threshold should be below the value of all rock regions in the sample and above void regions outside the sample's extents. This important step marks these regions so they may be deciphered as voids in step 6.

6. Subtract the beam hardening trend produced in step four from the image produced in step five (Figure 4.14). This image is now the original image with the beam hardening artifact trend removed and many void spaces assigned a high pixel intensity.

7. Assign regions of very high pixel intensity a low pixel intensity so they have less intensity than those pixels designating solid sample material (Figure 4.15). This is the end product of the beam hardening artifact removal process.

8. Once these previous steps have been applied to each cross sectional slice of an X-ray CT volumetric file, the images produced in step seven may be imported together back into a volumetric data file.

Figure 4.14. Effect of subtracting the beam hardening trend determined in Figure 4.13 from the original data once the original data has had its external void regions marked by high intensities. Intensity values found along a centrally located reference line are included.

Figure 4.15. Effect of assigning low intensity values to void spaces marked in Figure 4.14 with high intensities. Intensity values found along a centrally located reference line are included.

Void ratio distribution

Once the beam hardening artifact has been removed from the X-ray CT files, differentiation between void and solid regions requires finding a threshold voxel intensity that differentiates these two phases. Once determined and applied the quantity of void and solid voxels can be summed. The ratio of void voxels over solid voxels within any region defines the void ratio of that region.

To calculate the voxel intensity threshold that separates solid from void, the actual void ratios of sample quadrants was determined. All concrete samples were each cut into four horizontal sections and a volumetric analysis was conduction on each quadrant using the following three equations:

$$V_{solids} = \frac{W_{dry} - W_{submerged}}{\gamma_{water}} \qquad (4.1)$$

$$SG_{solids} = \frac{W_{dry}/V_{solids}}{\gamma_{water}} \qquad (4.2)$$

$$e = \frac{V_{total} - V_{solids}}{V_{solids}} \qquad (4.3)$$

where V_{solids} is the volume of solid material in a sample quadrant, W_{dry} is the dry weight of a sample quadrant, $W_{submerged}$ is the weight of a sample quadrant submerged in water, γ_{water} is the unit weight of water, SG_{solids} is the specific gravity of solid materials, e is the void ratio of a sample quadrant, and V_{total} is the total volume of a sample quadrant. The total volume was calculated from careful caliper measurements. Void ratio was determined for each quadrant independently, accounting for the difference in rock densities found in the river gravel samples (some coal gravel was encountered) as well as the different proportions of concrete and rock found throughout the specimens. Before being weighed in water, sample quarters were crushed to release entrapped air voids.

Voxel intensity thresholds were determined by matching void ratios from X-ray CT volumetric files to void ratios produced from laboratory volumetric analysis. The thresholds of each sample quadrant were averaged together and the mean value used as the threshold for analyzing all samples. Using this mean threshold, void ratios from CT data were compared to void ratios from volumetric density analysis (Figure 4.16), showing a mean void ratio difference of 0.021. Variance could be caused by either errors in X-ray CT data or errors in physical measurements.

Void ratio distribution was computed as a function of height within samples. Void ratio measurements were taken from X-ray CT data at 16 roughly equally spaced horizontal sections resulting in vertical resolution of roughly 4.6 mm (0.18 in), roughly the size of river gravel sample aggregates.

Figure 4.16. Histogram comparing void ratios measured using volumetric analysis to void ratios from X-ray CT analysis. Sample number designation refers to samples 1 through 4, decimal quarters 1 though 4 (sample base is 1, top 4).

Void ratio was calculated at the exterior regions of samples to determine the edge-effect of casting samples in a cylindrical mold. In each sample, the quadrant exhibiting the lowest void ratio was analyzed. Void ratio was computed in the outer 2.38 mm (0.094 in) of the quadrant. This thickness is roughly half the size of the aggregates in the river gravel samples. This exterior region composes roughly 11% of the cross sectional area of samples.

Continuity index

To quantify pore continuity in a sample, pore spaces were tracked vertically starting from a centrally located horizontal cross section. Tracking pore spaces was completed by viewing cross-sectional slices spatially in vertical order, one atop another, so that both vertical and lateral pore pathways could be followed within consecutive slices.

Results are presented in terms of successful continuity as pores lead to radial (horizontal) or end (vertical) exits and also which direction (up or down) from the central cross section the successful continuity propagates towards. Pore labels and locations are shown in Figure

4.11. One central cross section is analyzed per sample of which all voids dissected by the cross sectional slice are included in the analysis. These areas are sometimes split into multiple pore sections if the pore forks in close vertical proximity (see voids OO and PP in the lower right corner of sample 3 in Figure 4.17).

Figure 4.17. Pore spaces on each PCPC sample central cross section, designated alphabetically. Each lettered void was followed to find whether or not it led to radial or end exits in both upward and downward directions.

When following pores, continuity moving vertically was considered but not continuity requiring vertical backtracking. This concept is illustrated by Figure 4.18. This was done as a simplification in the analysis but also has practical implications as follows: (1) backtracking pore continuity may become clogged by settling sediments in field conditions similarly to a trap in a sink and (2) backtracking pores would provide slower drainage compared to a more direct route.

Once radial and end exit continuity was determined the total areas attributed to successfully continuous voids along the central cross section were computed and compared to the total cross sectional void area. A ratio (expressed as a percentage) of successfully continuous pore areas over total cross sectional area is the continuity index of a sample. Samples may be compared by radial continuity index, end continuity index, or an average of radial and end continuity indexes.

Figure 4.18. Continuity index illustration exemplifying upward radial continuity, downward end continuity but not upward end continuity because it would require backtracking.

Image analysis

Image analysis was used to prepare images and obtain quantitative, comparable data representative of pore size and shape. Image analysis was conducted using Visilog 4.1.5 software (Noesis) available at the Materials Analysis Research Laboratory (MARL) at Iowa State University. Image preparation consisted of three steps as follows:

1. Binary image production. Use a calculated and calibrated threshold voxel intensity to separate solid space from void space in X-ray CT produced images, and assign each a single color (Figure 4.19.a).

2. Removing bordering void spaces. This consists of removing all void colored areas open to the picture's exterior (Figure 4.19.b). Air space surrounding the sample is removed from the analysis as well as some pores that connect to the sample perimeter. For all samples except sample 1, many representative pore remained after removal of border void spaces. Due to the high interconnectivity of void spaces in sample 1, the removal of exterior pores removed large portions of void space. Large voids at the edge of sample 1 were therefore separate from the exterior of the sample with a bridge of solid colored pixels prior to removing border void spaces (Figure 4.20). Although this manipulated the size and shape of the existing voids, the error is eclipsed by the inherently large size and perimeter of the voids.

3. Opening of void spaces. This step subtracts and then adds a pixel layer around void colored regions in the image, otherwise known as erosion and dilation (Figure 4.19.c). Opening has been used to process X-ray CT scans of rock in other studies (Mokwa and Nielson, 2006). The benefit is voids of sufficiently small size will be removed from the image and void edges are smoothed as fingers of void space moving away from void areas are flattened. The removed void spaces are either due to noise in X-ray CT files, very small pores in the concrete, or void channels which are very narrow. Opening will remove regions of pores which have a smaller width than 0.30 mm (0.012 in). Another additional benefit to opening is that many ring artifacts will be removed from images (Figure 4.21).

4. Closing of void spaces. As an image is closed it undergoes dilation then subtraction (Figure 4.19.d). This helps further smooth void edges as fine fingers moving into pore spaces are flattened.

Figure 4.19. Image analysis steps including the original image (a) after border removal (b), adding a single pixel layer opening step (c) and finally adding a single pixel layer closing step (d). Example picture series is from sample 3's uppermost cross section.

Figure 4.20. Border removal problem for sample 1. Original image (a) has most of its void space removed due to high continuity of voids which contact the sample perimeter (b). By blocking off large voids from the outside of the sample (c) many of these voids are preserved during border removal (d). Example picture series is from sample one, three quarters up the sample's height.

a.   b.

Figure 4.21. Ring artifact removal from opening step of image analysis preparation. Ring artifact is visible as concentric rings of apparent void space (a). These artifacts get removed by the opening process (b). Example picture series is from sample three, 3/16[ths] up the sample's height.

Sixteen cross sectional images were taken from volumetric X-ray CT files for each sample. The resulting vertical resolution is 4.8 mm (0.19 in) between analyzed layers, roughly the size of river gravel aggregates.

The product of these image preparation steps is the ability to analyze void size and shape. Voids are now represented by island-like areas called blobs. Void sizes are equal to the number of pixels composing a blob in the image. The shape is defined by a shape factor. This shape factor is defined by Equation 4.4. To compare between samples, a weighted average of shape factors defined by Equation 4.5 (McClain et al. 2003) was employed.

$$SF = \frac{P^2}{4\pi A} \tag{4.4}$$

$$SF_{AW} = \frac{\sum_{i=1}^{n} SF_i A_i}{\sum_{i=1}^{n} A_i} \qquad (4.5)$$

where A is the pixel area of a void, P is the perimeter distance around this area and n is the number of areas analyzed in the region of interest (an X-ray CT produced cross section). The lowest shape factor possible is from a circle and has a value of one. As voids grow more irregular in shape, their shape factor grows accordingly. Shape factor is therefore equivalent to the number of circular perimeters a void has around its perimeter if the encompassed areas are equivalent. As voids become connected, the shape factor of the resulting void increases. By Equation 4.4 if two similar voids come into contact by a very small (negligibly sized) channel, the resulting pore has twice the area and twice the perimeter, resulting in a doubled shape factor. Therefore shape factor is influenced by pore interconnectivity.

To comparable different concrete samples, averaged void sizes and shape factors were used. The average void size is defined as the median void area of all analyzed sections of a sample. The averaged shape factor conforms to Equation 4.5 but excludes pores having an area less than 0.065 cm^2 (0.010 in^2) because the resolution of X-ray CT images and the surface smoothing processes can't produce the shape of these small pores (the result of these small pores would be uncharacteristically small shape factors). The exclusion of these small void spaces is important as samples two through four all have over 10% of their void areas represented in this void size fraction, the greatest of which is sample three having 16.7%. Sample 1 shows only 3.5% voids of this size.

Results and Discussion

Continuity index

Successful continuity for the four mix samples was found for radial and end exits in both the upward and downward directions form a reference center slice (Figures 4.22 through 4.25). Sample 1, with a larger aggregate size, has the most successful continuity by far with over half of its void areas finding their way to the sample ends. Sample 1 also has the highest permeability out of the four samples. Samples 2 and 3 both have zero end continuity in one direction and have the two lowest permeabilities. Sample 4 shows vertical continuity in almost half of its void areas and exhibited the second highest permeability. Average continuity indexes (mean averages of a specimen's successful continuities in the two exit directions) are plotted on log scale against permeability measurements in Figure 4.26.

Figure 4.22. Continuity success of void areas found along a central slice in sample 1.

Sample 2

3.9%
Radial up

0.0%
End up

35.0%
Radial down

40.7%
End down

Figure 4.23. Continuity success of void areas found along a central slice in sample 2.

Sample 3

43.8%
Radial up

27.1%
End up

10.3%
Radial down

0.0%
End down

Figure 4.24. Continuity success of void areas found along a central slice in sample 3.

Figure 4.25. Continuity success of void areas found along a central slice in sample 4.

This data shows an increase in permeability with increasing end continuity index. Radial continuity seems more random. Although a very limited amount of data is present, this is sensible as radial continuity is more intensely affected by the relative distance of large pores to the sample extents (close proximity to the edge increases the chance of continuity out from that edge) as well as the randomness caused by varying path lengths prior to encountering a radially exiting pore (radially exiting, highly connected pore spaces found closer to the central reference slice increases the chance of continuity).

Figure 4.26. Log scale graph of end, radial, and averaged end and radial continuity indexes each versus permeability. In order of lowest to highest permeability are samples 3, 2, 4, and finally 1.

Void ratio distribution

Using an intensity threshold that is calibrated to differentiate solid from void space in X-ray CT volumetric files, void ratio was computed at height intervals within four mix design samples. Void ratios were determined in sample quarters and sixteenths. Results are presented in Figure 4.27. This data supports the outcome of the continuity analysis and laboratory permeability measurements. The consistently highest void ratio sample is mix 1 with sample 4 following. Samples 2 and 3, the samples with no end continuity in a direction, show low void ratio measurements in the respective direction of the lacked continuity. These low continuity directions contain two sample quarters each exhibiting void ratios under 0.20.

The void ratio distributions show a wide variation of void ratios along the height of the samples. Unbiased standard deviations for samples one through four are 0.070, 0.076,

0.043 and 0.060 void ratio respectively. It was investigated whether the low void ratio sections might control permeability behavior. Comparing plots of both sample average void ratios (Figure 4.28) and lowest sixteenth of a sample void ratios (Figure 4.29) versus permeability suggests this may not be the case for the limited number of samples analyzed. An R^2 least regression value for the lowest sixteenth analysis was significantly less than that for the average void ratio, 0.71 to 0.93 respectively. They did, however, mimic each other in shape, having increasing permeability with void ratio increase with the exception of the second lowest permeability sample (sample 2) which has the lowest void ratio.

Samples 2 and 3 both have directions which exhibited end continuity indexes of 0. This lack of continuity points to the possibility that water flow may be concentrating itself somewhere other than through the sample's internal volume. An investigation into the void ratio of exterior sample regions (outer 2.38 mm (0.094 in)) versus internal regions revealed a large discrepancy as shown in Figure 4.30. The outer extents of these regions have void ratios averaging 5.3 times that of internal void ratios in the river gravel samples two through four and 5.4 times in the limestone sample 1. This effect may be the result of casting the samples in a cylindrical mold. It should be noted that during permeability testing, water flow around the edges was reduced by applying a gum to the outer roughly 1.5 centimeters of the sample's top and bottom and a rubber/latex membrane was compressed part of the way into the sample sides. Despite these measures, water could likely travel in the sample's open void structure exterior through much of the sample height and enter or exit radially into or from the internal void space.

Figure 4.27. Void ratio distributions for both vertical quarters (bars) and sixteenths (points) of sample 1 (a), sample 2 (b), sample 3 (c) and sample 4 (d).

Figure 4.28. Entire sample void ratio as determined from volumetric density
analysis logarithmically plotted against sample permeability.

Figure 4.29. The lowest void ratio measure from a sixteenth sample section as determined
from X-ray CT logarithmically plotted against sample permeability.

Figure 4.30. Exterior and interior void ratios found in the lowest void ratio quarter of samples as determined from X-ray CT plotted against sample permeability.

Image analysis

From sixteen cross sectional images taken from each sample using X-ray CT the average void size and shape of each sample was quantified. Sample average areas and shape factors are plotted against permeability measurements (Figures 4.31 and 4.32). Increasing permeability with larger pore size and shape is reasonable as larger pores or pores with more irregular (and therefore more outstretching) shapes would aid in sample continuity. The very large void shown in Figure 4.20 for instance has a size of 1.06 in^2 and a shape factor of 35.6 and is an obvious example of well interconnected voids.

Figure 4.31. Sample average void area versus permeability.

Figure 4.32. Sample average shape factor versus permeability.

Conclusions

The following achievements and findings are derived from or supported by this research project:

1.) Stacking and shielding samples during X-ray CT scans can remove blurring of rock/air interfaces.

2.) Beam hardening artifacts may be removed by subtracting the artifact trend from CT scan images using the methodology developed for this project.

3.) Thresholding was used to successfully differentiate solid from pore space in PCPC samples after beam hardening artifact removal.

4.) A continuity index was devised which quantifies successfulness void continuity.

5.) Continuity, void ratio, pore size and pore shape were quantified and compared to permeability. Each showed a positive correlation to permeability.

6.) Void ratio around the outer extents of concrete samples was determined, showing a drastic increase in void ratio over that of internal regions. This presents the necessity for further research to use only core samples for PCPC permeability testing.

Research Recommendations

Further research of PCPC should optimize the use of pore spaces for water permeation. Obtaining maximum permeation from a minimum void content would increase concrete strength, freeze-thaw resistance, and service life. X-ray CT analyses such as those conducted in this research can characterize pore space qualities of different mix designs and placement procedures which demonstrate a positive influence on void matrix permeation. It is therefore recommended that further research applying X-ray CT to PCPC mix design should be conducted.

Several aspects of the PCPC analysis carried out in this research were not ideal. Such notable aspects are as follows:

1.) Samples which were laboratory tested for permeability were halved vertically before being supplied for X-ray CT analysis.

2.) Analyzed samples were cast in molds. The mold likely caused open pore spaces between aggregates at the sample perimeter which has the potential to alter both lab permeation and X-ray CT analysis results.

3.) Only four samples were analyzed. Of these, three of the four had relatively similar permeability results.

It is the author's opinion that further research of PCPC should be conducted using X-ray CT. Samples should be of full depth cores on which lab permeability tests have been conducted and which represent a variety of experimental mix designs and placement procedures.

References

Barrett, H.H., W. Swindell, 1981. "Radiological Imaging, The theory of Image Formation, Detection, and Processing," Academic Press, Department of Radiology and Optical Sciences Center, University of Arizona, Vol. 1 and 2.

Beeldens, A.D., V. Gemert, C Caestecker, 2003. "Porous Concrete/ Laboratory Versus Field Experience." Proceedings of the 9th International Symposium on Concrete Roads, Istanbul, Turkey.

Kini, V.R., 1994, "Tomographic Inspection System Using X-rays," Iowa State University, Master of Science thesis.

McClain, S.T., J.T. Berry, B. Dawsey, 2003. "A Study of Porosity and Pore Morphology in Aluminum A356.2 Step Castings." American Foundry Society.

Mokwa, R.L., B. Nielson, 2006. "Nondestructive Measurements of Soil Geotechnical Properties Using X-Ray Computed Tomography." ASCE, GeoCongress 2006.

National Ready Mixed Concrete Association (NRMCA), 2004. "Freeze-Thaw Resistance of Pervious Concrete."

Olek, J, W.J. Weiss, N. Neithalath, A. Marolf, E. Sell, W.D. Thornton, 2003. "Development of Quiet and Durable Porous Portland Cement Concrete Paving Materials."

POD (Probability of Detection), Version 1.0, ©1999-2003 NDE Technologies, Inc., 2501 North Loop Drive, Suite 1614A, Ames, Iowa 50010.

Schaefer, V.R., K. Wang, M. Suleiman, J. Kevern, 2006 "Mix Design Development for Pervious Concrete in Cold Weather Climates." Center for Transportation Research and Education, Iowa State University.

CHAPTER 5. GENERAL CONCLUSIONS

This research project developed methods to analyze systems of continuous pore spaces and sand grain movement. These methods allowed investigation into the characteristics of pore spaces found within Portland Cement Pervious Concrete and the movement of sand particles within a granular sample subjected to triaxial compression. The most significant product of this research is likely the methods applied to conducting X-ray CT scans, processing the resultant 3-dimensional output files, and computing the necessary results. Both the methods developed and the test results obtained will be described in short in this section. For an expanded discussion of conclusions, refer to the respective research project chapters.

Method Development

The characterization of sand grain movement under triaxial compression required developing several procedural and analytical methods. These methods resolved the following issues:

1.) A system of marking sand particles was required. Particles needed to be referenced not only in terms of location within the sample but angular orientation.

2.) Conventional hardware components used for conducting triaxial compression tests are not well suited for conducting X-ray CT scans. Specifically, the tie rods which hold the top of the triaxial chamber to its base are typically metallic and block X-rays from reaching the sample, creating shadows in resultant data.

3.) A means of scanning samples at consecutive strain intervals, which required rotating the sample 360 degrees, was necessary.

4.) An algorithm for computing various components of particle movement needed to be developed.

5.) A way of organizing 3-dimensional data 2-dimensionally was required to visually illustrate movement patterns within the samples.

These concerns were addressed using the following methods, in respective order:

1.) Sand grains were marked using three individually identifiable solder pieces. Solder showed up well in X-ray CT images due to its high density. Solder piece identification was allowed by using different lengths and widths of solder wire. By using three pieces, a three dimensional reference was made. Using only two solder pieces, for instance, would disallow quantifying rotations around the axis formed by the two pieces. The three pieces were located about the particle's exterior at a maximum distance between each other to increase the accuracy of computing the alignment of vectors extending from one piece to another.

2.) By using custom tie rods made from high strength plastic, the issue of shadows across the sample was mitigated. It should be noted that relatively low confining pressures were used for the experiment so the bars would not become plastically deformed.

3.) By using a nitrogen air tank to supply confining pressure, the triaxial compression tests could be conducted in close proximity to the X-ray CT scanning chamber. Also, an aluminum plate was milled which held the cylindrical base of the triaxial sample in place. The plate was equipped with a screw to use as a reference for initial rotational alignment of the chamber. The plate allowed the sample to be placed in an approximately identical starting position for every X-ray CT scan. By moving the sample from the loading frame to the X-ray chamber and back, the sample could be scanned at successive strain intervals.

4.) The algorithm developed is very long and complex. For an expanded explanation of the algorithm mechanics and accuracy please visit Chapter 3 of this thesis. To concisely explain the steps, the mean average position of the three solder points is tracked from one strain increment to another to quantify

particle translation. Rotation is computed from nine reference vectors. These vectors are derived from the alignment of the three solder pieces to one another. The maximum angular change seen by one vector is considered the critical vector from which rotation is computed.

5.) From visual inspection and published research, granular triaxial samples tend to fail with radial symmetry. Therefore, illustrating sample movement in terms of radial and vertical distance effectively depicts 3-dimesional data in a convenient 2-dimensional plot.

To adequately apply X-ray CT to Portland Cement Pervious Concrete (PCPC) a variety of concerns needed to be addressed. Many of these concerns are related to producing images of sufficient quality to properly analyze void structure and therefore are concerns for various research applications utilizing X-ray CT. Critical concerns are as follows:

1.) A beam hardening artifact which causes an erroneous gradient of increasing voxel intensity with increasing radial distance was present in volumetric images. The beam hardening artifact made segregation between pore and solid space radially dependent.

2.) The perimeter, top and bottom of the cylindrical samples were found to exhibit a blurring of high intensity voxel regions into low intensity voxel regions and visa versa. This intensity smearing introduced problems when attempting to differential solid space from void space.

3.) Ring artifacts within volumetric images caused erroneous rings of high voxel intensity. These rings' intensities exceed those of solid spaces, and can run through imaged void spaces making identification of individual pore cross sections difficult.

4.) A calibrated method of differentiating between solid and pore space needed to be developed and applied. This would require a single voxel intensity which serves as a threshold above which voxels are recognized as solid space and below which voxels are recognized as void space.

5.) The edge of void spaces often exhibit voxel intensities similar to the intensity threshold, causing blurring of pore space perimeters in thresholded images. In actuality, the pore space edges are clean and smooth as a thin layer of cement coats aggregates. This effect makes proper quantification of pore space perimeters difficult.

6.) A repeatable means of relating pore space distribution to permeability was required to evaluate the effective use of pore space towards water transport.

These issues were addressed using innovative scanning methods and digital post processing techniques as follows, in respective order:

1.) The beam hardening artifact was removed from the data by first exporting each horizontal slice from the volumetric image. Falling window and area filters were used to obtain the trend of the beam hardening artifact. By subtracting the trend from the original image, a resultant image without the artifact was produced.

2.) The approach used to remove edge smearing was to make the external edges not be the edges of the sample. By shielding the sample in a ceramic tube and placing other samples above and below the significant sample, the smearing occurred between the peripheral masses and the external air environment instead of affecting the critical sample.

3.) The removal of ring artifacts took place during horizontal cross section post processing using a process called erosion where a single pixel layer from around the ring artifact voxel space was removed. Due to their inherently small thickness, this post processing step often removed the artifacts completely.

4.) All specimens were of equal cross section and where scanned at equivalent voltage and amperage levels. Therefore, the voxel intensity cutoff between void and solid space was relatively equivalent for all samples. Once all sample quadrants where tested in the laboratory to determine the volume of solids found within, all X-ray CT scans of the quadrants were analyzed to find what

voxel intensity cutoff adequately reflected the known solid volumes. Once the most accurate intensity threshold was established in each sample quadrant, the intensities were mean averaged to arrive at a calibrated cutoff voxel intensity which can be effectively applied to all sample sections.

5.) By removing and subsequently adding a voxel layer around void spaces, the blurry edges of voids smooth out. This process, known as opening, was applied to 2-dimensional cross sections of volumetric X-ray CT files.

6.) A newly devised continuity index quantifies the ratio of pore spaces found at a vertically centralized horizontal cross section which successfully and directly lead to radial or end-of-sample exits over the total pore space area of the cross section. This measure of pore space continuity reflects the ability of existing pore spaces to transport water from one end of a sample to another.

Research Analysis Results

Findings from tracking sand grain movements within triaxially compressed samples include the following results:

1.) Particle movements, both directional and rotational, are concentrated at the height within a sample where bulging can be seen. An exception is rotation within the highest confinement pressure, low strain interval sample 3.

2.) Particles showing little rotation can be very closely located in terms of height and radial distance to particles showing much larger rotations.

3.) Average directional sand grain motion becomes increasingly horizontal until it becomes predictable in magnitude. This occurred before peak deviator stress was achieved.

4.) Sand grains in samples under higher confining pressure exhibited less horizontal movement but greater particle rotation.

5.) The axes of rotating particles were shown to be randomly inclined throughout samples.

6.) Rotational movements are roughly equal to horizontal translations in the highest confining pressure sample, 3. As confining pressure increases, the ratio of rotational movement over horizontal translation grows.

The analysis of PCPC revealed the following results:

1.) Pore space characteristics which were shown to be associated with greater permeability are the following:
   a. Large size.
   b. High quantity of perimeter area in relation to volume.
   c. High void ratio.
   d. Greater amounts of successfully continuity, judged using the newly devised continuity index.
2.) Sample edges were shown to contain higher void space and continuity than the inner sample regions.

CHAPTER 6. GENERAL RECOMMENDATIONS

The methods developed in this research project to analyze granular movements have great application to understanding movements within various geomaterials in many situations. Utilizing these methods is advised. However, during the progression of data analysis several potential refinements to the approach currently undertaken became evident. Firstly, the powerful X-ray equipment used was capable of accommodating larger samples which would allow for using larger grain sizes and subsequently increase the quality of quantifying various particle movements. Secondly, a means of creating a more homogenous distribution of a greater number of marked particles within the sample volume would result in a better representation of more sample regions. Thirdly, using smaller sand grain markers would ease the task of determining their central voxel locations. Lastly, a program which can collect data points accurately would allow more efficient analysis as time consuming manual means of data collection could be avoided.

Although lab permeability results are the definitive measure of pore space adequacy in effectively draining surface water, the use of X-ray CT analyses can help reveal the relationships between aspects of pore space and permeability. X-ray CT can then help develop and inspect concrete mixes and placement procedures which utilize positive pore space aspects. The result of optimizing available pore space is adequately providing pavement permeability while retaining strength, freeze-thaw resistance, and a long project life, all of which are benefited by reducing the amount of large voids. It is therefore recommended that further research applying X-ray CT to PCPC mix design should be conducted.

The analysis of PCPC was conducted using samples which were not ideal. The samples provided for analysis where halves of samples used to calculate laboratory permeability results. The samples were also cast in molds, leaving edge effects which alter lab permeability and X-ray CT analysis results. In addition, only four samples were analyzed

in the PCPC research, three of which exhibited comparable permeability rates which does not allow reliable correlations of sample traits to permeability measurements.

Void size and shape was characterized using sample cross sections. Analysis of these void characteristics could be expanded using three dimensional void sizes and shapes to obtain a more complete representation of void spaces within samples.

It is the author's opinion that further research utilizing a wider variety of samples should be conducted using X-ray CT on full-depth specimens that have been cored from slabs of experimental mix designs and/or placed using experimental procedures.

WORKS CITED

Alramahi, B.A., K.A. Alshibli, A.M. Attia, 2005. "Influence of Grain Size and Consolidation Pressure on Porosity of Rocks." Geo-Frontier 2005 conference, Site Characterization and Modeling, GSP 138.

Alshibli, KA., B.A. Alramahi, 2006(a). "Applications of Computed Tomography (CT) to Characterize the Internal Structure of Geomaterials: Limitations and Challenges." ASCE, Site and Geomaterial Characterization, Geotechnical Special Publication No. 149.

Alshibli, K.A., B.A. Alramahi, 2006(b). "3D Visualization of Shearing in Granular Materials." ASCE, GeoCongress 2006.

Alshibli, K.A., S. Sture, N.C. Costes, M.L. Frank, M.R. Lankton, S.N. Batiste, R.A. Swanson, 2000. "Assessment of Localized Deformations in Sand Using X-Ray Computed Tomography." Geotechnical Testing Journal, Vol. 23, No. 3.

ASTM E1935, 1997. "Standard Test Method for Calibrating and Measuring CT Density." American Society for Testing and Materials.

ASTM D6706, 2001 "Standard Test Method for Measuring Geosynthetic Pullout Resistance in Soil." American Society for Testing and Materials.

Ayoubian, A., P.K. Robertson, 1998. "Void Ratio Redistribution in Undrained Triaxial Extension Tests on Ottawa Sand." National Research Council Canada.

Barrett, H.H., W. Swindell, 1981. "Radiological Imaging, The theory of Image Formation, Detection, and Processing," Academic Press, Department of Radiology and Optical Sciences Center, University of Arizona, Vol. 1 and 2.

Bauer, G.E., Y Zhao, 1993. "Evaluation of Shear Strength and Dilatancy Behavior of Reifroced Soil from Direct Shear Tests." <u>Geosynthetic Soil Reinforcement Testing Procedures</u>, ASTM STP 1190.

Beeldens, A.D., V. Gemert, C Caestecker, 2003. "Porous Concrete/ Laboratory Versus Field Experience." Proceedings of the 9th International Symposium on Concrete Roads, Istanbul, Turkey.

Binley, A., B. Shaw, S.H. Poulter, 1995. "Flow Pathways in Porous Media: Electrical Resistance Tomography and Dye Staining Image Verification," Measurement Science and Technology. 7 pg. 384-390.

Cazzuffi, D., L. Picarelli, A. Ricciuti, P. Rimoldi, 1993. "Laboratory Investigations on fthe Shear Strength of Geogrid Reinforced Soils." <u>Geosynthetic Soil Reinforcement Testing Procedures</u>, ASTM STP 1190.

Chang, M-F, C.I. Teh, LF Cao, 1999, "Critical State Strength Parameters of Saturated Clays from the Modified Cam Clay Model," Canadian Geotechnical Journal, Vol. 36.

Cullity, B.D., 1978. <u>Elements of X-ray Diffraction</u>, 2nd edition, Addison-Wesley Publishing Company, Department of Metallurgical Engineering and Materials Science, University of Notre Dame.

Das, B.M. 1998. <u>Principles of Geotechnical Engineering</u>, Fourth edition, International Thomson Publishing Inc.

Desrues, J. R. Chambon, M Mokni, F. Mazerolle, 1996. "Void Ratio Evolution inside Shear Bands in Triaxial Sand Specimens Studied by Computed Tomography," Geotechnique, Vol. 46.

Diagle, M., D. Fratta, L.B. Wang, 2005. "Ultrasonic and X-ray Tomographic Imaging of Highly Contrasting Inclusions in Concrete Specimens," Geo-Frontier 2005 conference. Austin, TX.

Fukahori, D., A. Sato, K. Sugawara, 2006. "Development of a new X-ray CT Permeation Test System and Application to Porous Rocks," ASCE, GeoCongress 2006.

Ferrag, K., P. Griffin, 1993, "Pull-out Testing of Geogrids in Cohesive Soils," Geosynthetic Soil Reinforcement Testing Procedures, ASTM STP 1190.

Haeri, S.M., R. Noorzad, A.M. Oskoorouchi, 2000, "Effect of Geotextile Reinforcement on the Mechanical Behavior of Sand," Geotextiles and Geomembranes, Vol. 18.

Ham, K., C.S. Willson, 2005. "The Application of Monochromatic Energies to Investigate Multiphase Porous Media Systems using Synchrotron X-ray Tomography," Geo-Frontier 2005 conference, GSP 138.

Jang, D.J., D.J. Frost, J.Y. Park, 1999. "Preparation of Epoxy Impregnated Sand Coupons for Image Analysis," Geotechnical Testing Journal, ASTM, Vol. 22, No. 2.

Jang, D.J., D.J. Frost, 2000. "Use of Image Analysis to Study the Microstructure of a Failed Sand Specimen," Canadian Geotechnical Journal, Vol. 37.

Jones, C., 1996. Earth Reinforcement and Soil Structures, ASCE press.

Kikuchi, Y., T. Magatome, J. Otani, 2006. "Visualization of Permeability Phenomena in Lightweight Treated Soil Mixed with Air Foam using Micro-Focus X-ray CT Scanner," ASCE, GeoCongress 2006.

Kini, V.R., 1994, "Tomographic Inspection System Using X-rays," Iowa State University, Master of Science thesis.

Kuo, C.Y., D.J. Frost, 1996. "Uniformity Evaluation of Cohesionless Specimens Using Digital Image Analysis," Journal of Geotechnical Engineering, ASCE, Vol. 122, No. 5.

Lambe, W.T., A.W. Marr, 1979. "Stress Path Method: Second Edition," Journal of the Geotechnical Engineering Devision, ASCE, Vol. 105 No. GT6.

Lade, V.L., 2005. "Overview of Constitutive Models for Soils," Geo-Frontier 2005 conference.

Liang, Z.H., P.C. Lauterbur, 2000. <u>Principles of Magnetic Resonance Imaging – A Signal Processing Perspective</u>, Institute of Electrical and Electronics Engineers, Inc.

McClain, S.T., J.T. Berry, B. Dawsey, 2003. "A Study of Porosity and Pore Morphology in Aluminum A356.2 Step Castings." American Foundry Society.

Mokwa, R.L., B. Nielson, 2006. "Nondestructive Measurements of Soil Geotechnical Properties Using X-Ray Computed Tomography." ASCE, GeoCongress 2006.

National Ready Mixed Concrete Association (NRMCA), 2004. "Freeze-Thaw Resistance of Pervious Concrete."

Newland, P.L., B.H. Allely, 1957. "Volume Changes in Drained Triaxial Tests on Granular Materials." Geotechnique, Vol. 7.

Ngan-Tillard, D.J.M., X.H. Cheng, J. van Nex, P.L.J. Zitha, 2005. "Application of X-Ray Computed Tomography to Cone Penetration Tests in Sands." Geo-Frontier 2005 conference, Site Characterization and Modeling, GSP 138.

Oda, M., 1972. "The Mechanism of Fabric Changes During Compressional Deformation of Sand." Soils and Foundations, Japanese Society of Soil Mechanics and Foundation Engineering, Vol. 12, No. 2.

Oda, M., J. Konishi (1982). "Experimental Micromechanical Evaluation of Strength of Granular Materials: Effects of Particle Rolling." Mechanics of Materials Vol. 1.

Oda, M., H. Hazama, 1998. "Microstructure of Shear Bands and Its Relationship to the Mechanism of Dilatency and Failure of Dense Granular Soils." Geotechnique, Vol. 48, No. 4.

Olek, J, W.J. Weiss, N. Neithalath, A. Marolf, E. Sell, W.D. Thornton, 2003. "Development of Quiet and Durable Porous Portland Cement Concrete Paving Materials."

Pham, K.D., J. Otani, Y. Watanabe, T. Makunoki, 2006. "Application of X-Ray CT on Boundary Value Problems in Geotechnical Engineering – Research on Ground Failure Due to Lateral Pile Loadings." ASCE, GeoCongress 2006.

POD (Probability of Detection), Version 1.0, ©1999-2003 NDE Technologies, Inc., 2501 North Loop Drive, Suite 1614A, Ames, Iowa 50010.

Sato, A., D. Fukahori, A. Takebe, A. Sawada, D. Sugawara, 2006. "Analysis of Open Crack in Rock by X-Ray CT." ASCE, GeoCongress 2006.

Schaefer, V.R., K. Wang, M. Suleiman, J. Kevern, 2006 "Mix Design Development for Pervious Concrete in Cold Weather Climates." Center for Transportation Research and Education, Iowa State University.

Seed, H.B., K.L. Lee, 1967. "Undrained Strength Characteristics of Cohesionless Soils." ASCE, Journal of the Soil Mechanics and Foundations Division.

Walters, D.A., R.C.K. Wong, A. Kantzas, 1998. "The Application of Computer-Assisted Tomography in the Analysis of Fracture Geometry." Geotechnical Testing Journal, Vol. 21, No. 4.

Zhang, J., C., C.T. Jensen, J. Gray. 2005. "Development of a high resolution CT System and 3D Visualization Tools." To be published in Review of Progress in Quantitative NDE, American Institute of Physics, Vol. 24.

APPENDIX A: PCPC CROSS SECTIONAL IMAGE POST PROCESSING

The figures in Appendix A illustrate the effect of post processing X-ray CT cross sectional images of PCPC in order to develop suitable images for determining void sizes and shapes. The figures presented are of the first processed image of each sample before and after the processing. The post processing discards pore space from around the perimeter of the samples, smooths the fuzzy edges of pores, removes ring artifacts and eliminates very small pores from the analysis.

Figure A.1. Imitial image (left) and post processed image (right) from sample 1.

Figure A.2. Imitial image (left) and post processed image (right) from sample 2.

Figure A.3. Imitial image (left) and post processed image (right) from sample 3.

Figure A.4. Imitial image (left) and post processed image (right) from sample 4.

APPENDIX B: C++ COMPUTER PROGRAM, CALCULATION OF SOLDER PARTICLE MOVEMENT

The following computer program is the C++ program developed to analyze sand grain movements, both directionally and rotationally, as determined from the input of the three dimensional coordinates of reference solder pieces and sample ends. Note that the program often includes lines of syntax which exceed the page width.

```cpp
#include <fstream.h>
#include <iomanip.h>
#include <math.h>

void main (void)

{
        float XBcoord[5][50], YBcoord[5][50], ZBcoord[5][50];// declares two dimensional
        float XLcoord[5][50], YLcoord[5][50], ZLcoord[5][50];// arrays to store each coordinate type's
        float XTcoord[5][50], YTcoord[5][50], ZTcoord[5][50];// location for each particle and phase

        float Xpos[5][50], Ypos[5][50], Zpos[5][50]; // central location for each particle in each phase

        float resolution, ratio; // spatial resolution parameters so data may be presented in inches

        float Xcentertop[5], Ycentertop[5], Zcentertop[5]; // declares variables determining the central
coordinates at top
        float Xcenterbottom[5], Ycenterbottom[5], Zcenterbottom[5]; // and bottom for each testing phase

        float centerX[5][50], centerY[5][50]; // the center of the sample at a given height

        float rdis[5][50]; // the radial distance to a particle

        float centralV[2], centralUV[2]; // Vector and unit vector components to particle origin from sample
center
        float XcentralUV[2]; // a vector perendicular to both above vectors and vertical vector

        float BL[3], TM[3], MX[3]; // vector coordinates of B to L, T to mid of B and L, and their
perpendicular vector
        float OBL[3], OTM[3], OMX[3]; // last phase's vector coordinates of B to L, T to mid of B and L,
and their perpendicular vector
        float UVBL[3], UVTM[3], UVMX[3]; // now in unit vectors
        float OUVBL[3], OUVTM[3], OUVMX[3]; // now in unit vectors

        float UVBLTM1[3], UVBLTM2[3], UVTMMX1[3], UVTMMX2[3], UVMXBL1[3],
UVMXBL2[3]; // Additional vectors to increase accuracy
        float OUVBLTM1[3], OUVBLTM2[3], OUVTMMX1[3], OUVTMMX2[3], OUVMXBL1[3],
OUVMXBL2[3]; // These vectors are equispaced between those above

        float TUVBLTM1[3], TUVBLTM2[3], TUVTMMX1[3], TUVTMMX2[3], TUVMXBL1[3],
TUVMXBL2[3]; // Temporary holders of their respective vector components for use
```

C++ Program (Continued)

        float TOUVBLTM1[3], TOUVBLTM2[3], TOUVTMMX1[3], TOUVTMMX2[3], TOUVMXBL1[3], TOUVMXBL2[3]; // in a consistent denominator during UV transformation

        float vmult; // multiplier of vector BL which finds the point on the vector where a new vector may stretch to point T and be perpendicular to BL

        float deltaX, deltaY; // changes in X and Y position relative to central coordinates from last phase to current phase

        float Rmove[4][50], Vmove[4][50], XRmove[4][50]; // radial and vertical movement from one phase to another

        float alphadotBL, alphadotTM, alphadotMX, alphadotOBL, alphadotOTM, alphadotOMX; // dot product of above unit vectors and a vector perpendicular to angle alpha
        float betadotBL, betadotTM, betadotMX, betadotOBL, betadotOTM, betadotOMX; // dot produce of above unit vectors and a vector perpendicular to angle beta
        float thetadotBL, thetadotTM, thetadotMX, thetadotOBL, thetadotOTM, thetadotOMX; // dot product of above unit vectors and a vector perpendicular to angle theta

        float alphadotBLTM1, alphadotBLTM2, alphadotTMMX1, alphadotTMMX2, alphadotMXBL1, alphadotMXBL2; // Just like above but for extra vectors
        float alphadotOBLTM1, alphadotOBLTM2, alphadotOTMMX1, alphadotOTMMX2, alphadotOMXBL1, alphadotOMXBL2;
        float betadotBLTM1, betadotBLTM2, betadotTMMX1, betadotTMMX2, betadotMXBL1, betadotMXBL2;
        float betadotOBLTM1, betadotOBLTM2, betadotOTMMX1, betadotOTMMX2, betadotOMXBL1, betadotOMXBL2;
        float thetadotBLTM1, thetadotBLTM2, thetadotTMMX1, thetadotTMMX2, thetadotMXBL1, thetadotMXBL2;
        float thetadotOBLTM1, thetadotOBLTM2, thetadotOTMMX1, thetadotOTMMX2, thetadotOMXBL1, thetadotOMXBL2;

        float WalphadotBL, WalphadotTM, WalphadotMX; // finds which of the above dot products works the worst
        float WbetadotBL, WbetadotTM, WbetadotMX; // in terms of this phase or last phase
        float WthetadotBL, WthetadotTM, WthetadotMX; // (the best of these worst gets to define the angle)

        float WalphadotBLTM1, WalphadotBLTM2, WalphadotTMMX1, WalphadotTMMX2, WalphadotMXBL1, WalphadotMXBL2; // Just like above but for extra vectors
        float WbetadotBLTM1, WbetadotBLTM2, WbetadotTMMX1, WbetadotTMMX2, WbetadotMXBL1, WbetadotMXBL2;
        float WthetadotBLTM1, WthetadotBLTM2, WthetadotTMMX1, WthetadotTMMX2, WthetadotMXBL1, WthetadotMXBL2;

        float RXangle, XRXangle, CXangle, OXangle; // angle identifiers on XY plane for radial, cross radial, current, and old (measure from X axis)

        float Cadjacent, Oadjacent, Copp, Oopp; // adjacent and hypotenuse angle components of this and last phase

C++ Program (Continued)

```
        float Ctheta, Otheta, Calpha, Oalpha, Cbeta, Obeta; // current and old phase angles in theta, alpha
and beta
        float theta[4][50], alpha[4][50], beta[4][50]; // angular difference from one phases to another for
each particle
        float TR1[4][50], TR2[4][50], TR3[4][50], TR4[4][50], TR5[4][50], TR6[4][50], TR7[4][50],
TR8[4][50], TR9[4][50]; // total rotation of each of nine vectors during a phase increment

        int count, countb, phase; // Keep track of loops
        char looper, retry;  // let you contol how many particles you enter and allows you to correct is
misprinted

        ofstream Outputfile; // the output file for computed data

        cout << "This program computes particle rotation and translation \n";
        cout << "from a radial perspective using three identifying points \n";
        cout << "per particle in four different strain increments. \n \n \n";

        cout << "What is the horizontal resolution in voxels per inch?  \n";
        cin >> resolution;
        cout << "What is the ratio of vertical to horzontal resolution (inch per inch)?  \n";
        cin >> ratio;

      cout << "What is the X coordinate of the sample's top in the initial phase (smaller Z refers to the top)?
\n"; // here we enter the coordinates of the sample's center at its top and bottom
        cin >> Xcentertop[0];
        cout << "What is the Y coordinate of the sample's top in the initial phase? \n";
        cin >> Ycentertop[0];
        cout << "What is the Z coordinate of the sample's top in the initial phase? \n";
        cin >> Zcentertop[0];
        cout << "What is the X coordinate of the sample's bottom in the initial phase? \n";
        cin >> Xcenterbottom[0];
        cout << "What is the Y coordinate of the sample's bottom in the initial phase? \n";
        cin >> Ycenterbottom[0];
        cout << "What is the Z coordinate of the sample's bottom in the initial phase? \n";
        cin >> Zcenterbottom[0]; // input central coordinates

        looper = 'y';
        count = 0;
        countb = 0;
        phase = 1;  // Declare looping parameter initial values

        while (looper != 'n') // this begins the first loop which gathers data and computes for the first phase
        {
                cout << "enter the B coordinates of particle " << (count + 1) <<" phase 1\n";
                cin >> XBcoord[0][count] >> YBcoord[0][count] >> ZBcoord[0][count];
                ZBcoord[0][count] = (Zcenterbottom[0] - ZBcoord[0][count]) * ratio;

                cout << "enter the L coordinates of particle " << (count + 1) <<" phase 1\n";
                cin >> XLcoord[0][count] >> YLcoord[0][count] >> ZLcoord[0][count];
                ZLcoord[0][count] = (Zcenterbottom[0] - ZLcoord[0][count]) * ratio;
```

C++ Program (Continued)

```cpp
                    cout << "enter the T coordinates of particle " << (count + 1) <<" phase 1\n";
                    cin >> XTcoord[0][count] >> YTcoord[0][count] >> ZTcoord[0][count];
                    // finds coordinates for particles in phase 1
                    ZTcoord[0][count] = (Zcenterbottom[0] - ZTcoord[0][count]) * ratio;
            // makes their vertical height positive and beginning at zero

                    Xpos[0][count] = (XBcoord[0][count] / 3 + XLcoord[0][count] / 3 + XTcoord[0][count] /
3);
                    Ypos[0][count] = (YBcoord[0][count] / 3 + YLcoord[0][count] / 3 + YTcoord[0][count] /
3);
                    Zpos[0][count] = (ZBcoord[0][count] / 3 + ZLcoord[0][count] / 3 + ZTcoord[0][count] / 3);
    // finds average positions

                    centerX[0][count] = Xcenterbottom[0] + Zpos[0][count] / ((Zcenterbottom[0] -
Zcentertop[0]) * ratio) * (Xcentertop[0] - Xcenterbottom[0]); // finds centers at particle height
                    centerY[0][count] = Ycenterbottom[0] + Zpos[0][count] / ((Zcenterbottom[0] -
Zcentertop[0]) * ratio) * (Ycentertop[0] - Ycenterbottom[0]);


                    rdis[0][count] = sqrt(pow((Xpos[0][count] - centerX[0][count]), 2) + pow((Ypos[0][count]
- centerY[0][count]), 2)); //finds radial distance

                    count = count + 1;

                    cout << "Are these last inputs correct? y or n.";
                    cin >> retry;

                    if (retry == 'n')  // loop that lets you redo if you entered somthing wrong
                    {
                            count = count - 1;
                    }

                    cout << "want to do another particle? y or n:\n";
                    cin >> looper;
            }

        while (phase < 2) // this begins the second loop which gathers data and computes for the second
    phase and up and its change from the previous phase
        {


            while (countb < count) // note that as this loop starts, phases begins at 1 and countb at 0
            {
                    if (countb == 0) // input central top and bottom coordinates for each new phase
                    {
                            cout << "What is the X coordinate of the sample's top in phase " << (phase + 1) <<
    " \n"; // here we enter the coordinates of the sample's center at its top and bottom
                            cin >> Xcentertop[phase];
                            cout << "What is the Y coordinate of the sample's top in phase " << (phase + 1) <<
    " \n";
```

C++ Program (Continued)

```
                            cin >> Ycentertop[phase];
                            cout << "What is the Z coordinate of the sample's top in phase " << (phase + 1) <<
" \n";
                            cin >> Zcentertop[phase];
                        cout << "What is the X coordinate of the sample's bottom in phase " << (phase + 1)
<< " \n";
                            cin >> Xcenterbottom[phase];
                        cout << "What is the Y coordinate of the sample's bottom in phase " << (phase + 1)
<< " \n";
                            cin >> Ycenterbottom[phase];
                            cout << "What is the Z coordinate of the sample's bottom in phase " << (phase + 1)
<< " \n";
                            cin >> Zcenterbottom[phase]; // input central coordinates of this phase (because
top positions were found to waver with high sample strain)
                    }
                cout << "enter the B coordinates of particle " << (countb + 1) <<" phase " << (phase + 1)
<< " \n";
                cin >> XBcoord[phase][countb] >> YBcoord[phase][countb] >> ZBcoord[phase][countb];
                ZBcoord[phase][countb] = (Zcenterbottom[phase] - ZBcoord[phase][countb]) * ratio;

                cout << "enter the L coordinates of particle " << (countb + 1) <<" phase " << (phase + 1)
<< " \n";
                cin >> XLcoord[phase][countb] >> YLcoord[phase][countb] >> ZLcoord[phase][countb];
                ZLcoord[phase][countb] = (Zcenterbottom[phase] - ZLcoord[phase][countb]) * ratio;

                cout << "enter the T coordinates of particle " << (countb + 1) <<" phase " << (phase + 1)
<< " \n";
                cin >> XTcoord[phase][countb] >> YTcoord[phase][countb] >> ZTcoord[phase][countb];
//assign values for particles in phase 2 and up
                ZTcoord[phase][countb] = (Zcenterbottom[phase] - ZTcoord[phase][countb]) * ratio;

                Xpos[phase][countb] = (XBcoord[phase][countb] / 3 + XLcoord[phase][countb] / 3 +
XTcoord[phase][countb] / 3);
                Ypos[phase][countb] = (YBcoord[phase][countb] / 3 + YLcoord[phase][countb] / 3 +
YTcoord[phase][countb] / 3);
                Zpos[phase][countb] = (ZBcoord[phase][countb] / 3 + ZLcoord[phase][countb] / 3 +
ZTcoord[phase][countb] / 3); // finds average positions

                centerX[phase][countb] = Xcenterbottom[phase] + Zpos[phase][countb] /
((Zcenterbottom[phase] - Zcentertop[phase]) * ratio) * (Xcentertop[phase] - Xcenterbottom[phase]); // finds
center at particle height
                centerY[phase][countb] = Ycenterbottom[phase] + Zpos[phase][countb] /
((Zcenterbottom[phase] - Zcentertop[phase]) * ratio) * (Ycentertop[phase] - Ycenterbottom[phase]);


                rdis[phase][countb] = sqrt(pow((Xpos[phase][countb] - centerX[phase][countb]), 2) +
pow((Ypos[phase][countb] - centerY[phase][countb]), 2)); //finds radial distance

                Rmove[phase - 1][countb] = rdis[phase][countb] - rdis[phase - 1][countb];// Compute the
change in radial distance between this and last phases
```

C++ Program (Continued)

Vmove[phase - 1][countb] = Zpos[phase][countb] - Zpos[phase - 1][countb];// Compute the change in vertical position between this and last phases

deltaX = Xpos[phase][countb] - Xpos[phase - 1][countb] - centerX[phase][countb] + centerX[phase - 1][countb];
deltaY = Ypos[phase][countb] - Ypos[phase - 1][countb] - centerY[phase][countb] + centerY[phase - 1][countb];
XRmove[phase - 1][countb] = sqrt(deltaX * deltaX + deltaY * deltaY - Rmove[phase - 1][countb] * Rmove[phase - 1][countb] + .0001); // Compute absolute value of movement lateral to radial direction

centralV[0] = Xpos[phase - 1][countb] - centerX[phase - 1][countb]; // finds X distance to particle from center
centralV[1] = Ypos[phase - 1][countb] - centerY[phase - 1][countb]; // finds Y distance to particle from center

centralUV[0] = centralV[0] / sqrt(centralV[0] * centralV[0] + centralV[1] * centralV[1]); // X component of radial unit vector
centralUV[1] = centralV[1] / sqrt(centralV[0] * centralV[0] + centralV[1] * centralV[1]); // Y component of radial unit vector

XcentralUV[0] = centralUV[1]; // finds cross tangent vector to radial unit vector 90 degree clockwise from radial
XcentralUV[1] = 0 - centralUV[0]; // when looking down z direction and perpendicular to z direction also.

BL[0] = XLcoord[phase][countb] - XBcoord[phase][countb]; // vector BL from B to L
BL[1] = YLcoord[phase][countb] - YBcoord[phase][countb];
BL[2] = ZLcoord[phase][countb] - ZBcoord[phase][countb];

vmult = (BL[0] * (XTcoord[phase][countb] - XBcoord[phase][countb]) + BL[1] * (YTcoord[phase][countb] - YBcoord[phase][countb]) + BL[2] * (ZTcoord[phase][countb] - ZBcoord[phase][countb])) / (BL[0] * BL[0] + BL[1] * BL[1] + BL[2] * BL[2]);
TM[0] = XTcoord[phase][countb] - (XBcoord[phase][countb] + BL[0] * vmult); // vector perpendicular to BL passing through T
TM[1] = YTcoord[phase][countb] - (YBcoord[phase][countb] + BL[1] * vmult);
TM[2] = ZTcoord[phase][countb] - (ZBcoord[phase][countb] + BL[2] * vmult);

MX[0] = (BL[1] * TM[2]) - (TM[1] * BL[2]); // cross product vector is perpendicular to BL and TM
MX[1] = (BL[2] * TM[0]) - (TM[2] * BL[0]); // in such a way the if you point your finger in the direction
MX[2] = (BL[0] * TM[1]) - (TM[0] * BL[1]); // of BL and curl towards TM your thumb will point to MX

OBL[0] = XLcoord[phase - 1][countb] - XBcoord[phase - 1][countb]; // last phase's vector BL from B to L
OBL[1] = YLcoord[phase - 1][countb] - YBcoord[phase - 1][countb];
OBL[2] = ZLcoord[phase - 1][countb] - ZBcoord[phase - 1][countb];

C++ Program (Continued)

```cpp
                vmult = (OBL[0] * (XTcoord[phase - 1][countb] - XBcoord[phase - 1][countb]) + OBL[1]
* (YTcoord[phase - 1][countb] - YBcoord[phase -1 ][countb]) + OBL[2] * (ZTcoord[phase - 1][countb] -
ZBcoord[phase -1][countb])) / (OBL[0] * OBL[0] + OBL[1] * OBL[1] + OBL[2] * OBL[2]);
                OTM[0] = XTcoord[phase - 1][countb] - (XBcoord[phase - 1][countb] + OBL[0] * vmult);
// last phase's vector perpendicular to BL passing through T
                OTM[1] = YTcoord[phase - 1][countb] - (YBcoord[phase - 1][countb] + OBL[1] * vmult);
                OTM[2] = ZTcoord[phase - 1][countb] - (ZBcoord[phase - 1][countb] + OBL[2] * vmult);


                OMX[0] = (OBL[1] * OTM[2]) - (OTM[1] * OBL[2]); // cross product vector is
perpendicular to OBL and OTM
                OMX[1] = (OBL[2] * OTM[0]) - (OTM[2] * OBL[0]);
                OMX[2] = (OBL[0] * OTM[1]) - (OTM[0] * OBL[1]);



                if (fabs(BL[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        BL[0] = 0.00001;
                }

                if (fabs(TM[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        TM[0] = 0.00001;
                }

                if (fabs(MX[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        MX[0] = 0.00001;
                }

                if (fabs(OBL[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        OBL[0] = 0.00001;
                }

                if (fabs(OTM[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        OTM[0] = 0.00001;
                }

                if (fabs(OMX[0]) <= 0.00001) // Keep X axis from being zero to eliminate indeterminancy
later in program
                {
                        OMX[0] = 0.00001;
                }
```

C++ Program (Continued)

```
            if (fabs(BL[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    BL[1] = 0.00001;
            }

            if (fabs(TM[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    TM[1] = 0.00001;
            }

            if (fabs(MX[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    MX[1] = 0.00001;
            }

            if (fabs(OBL[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    OBL[1] = 0.00001;
            }

            if (fabs(OTM[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    OTM[1] = 0.00001;
            }

            if (fabs(OMX[1]) <= 0.00001) // Keep Y axis from being zero to eliminate indeterminancy
later in program
            {
                    OMX[1] = 0.00001;
            }


            UVBL[0] = BL[0] / sqrt(BL[0] * BL[0] + BL[1] * BL[1] + BL[2] * BL[2]); // scale BLs to
an unit vector
            UVBL[1] = BL[1] / sqrt(BL[0] * BL[0] + BL[1] * BL[1] + BL[2] * BL[2]);
            UVBL[2] = BL[2] / sqrt(BL[0] * BL[0] + BL[1] * BL[1] + BL[2] * BL[2]);

            UVTM[0] = TM[0] / sqrt(TM[0] * TM[0] + TM[1] * TM[1] + TM[2] * TM[2]); // scale
    TMs to an unit vector
            UVTM[1] = TM[1] / sqrt(TM[0] * TM[0] + TM[1] * TM[1] + TM[2] * TM[2]);
            UVTM[2] = TM[2] / sqrt(TM[0] * TM[0] + TM[1] * TM[1] + TM[2] * TM[2]);

            UVMX[0] = MX[0] / sqrt(MX[0] * MX[0] + MX[1] * MX[1] + MX[2] * MX[2]); // scale
    MXs to an unit vector
            UVMX[1] = MX[1] / sqrt(MX[0] * MX[0] + MX[1] * MX[1] + MX[2] * MX[2]);
            UVMX[2] = MX[2] / sqrt(MX[0] * MX[0] + MX[1] * MX[1] + MX[2] * MX[2]);
```

C++ Program (Continued)

```
                OUVBL[0] = OBL[0] / sqrt(OBL[0] * OBL[0] + OBL[1] * OBL[1] + OBL[2] * OBL[2]);
// scale last phase's BLs to an unit vector
                OUVBL[1] = OBL[1] / sqrt(OBL[0] * OBL[0] + OBL[1] * OBL[1] + OBL[2] * OBL[2]);
                OUVBL[2] = OBL[2] / sqrt(OBL[0] * OBL[0] + OBL[1] * OBL[1] + OBL[2] * OBL[2]);

                OUVTM[0] = OTM[0] / sqrt(OTM[0] * OTM[0] + OTM[1] * OTM[1] + OTM[2] *
OTM[2]); // scale last phase's TMs to an unit vector
                OUVTM[1] = OTM[1] / sqrt(OTM[0] * OTM[0] + OTM[1] * OTM[1] + OTM[2] *
OTM[2]);
                OUVTM[2] = OTM[2] / sqrt(OTM[0] * OTM[0] + OTM[1] * OTM[1] + OTM[2] *
OTM[2]);

                OUVMX[0] = OMX[0] / sqrt(OMX[0] * OMX[0] + OMX[1] * OMX[1] + OMX[2] *
OMX[2]); // scalelast phase's MXs to an unit vector
                OUVMX[1] = OMX[1] / sqrt(OMX[0] * OMX[0] + OMX[1] * OMX[1] + OMX[2] *
OMX[2]);
                OUVMX[2] = OMX[2] / sqrt(OMX[0] * OMX[0] + OMX[1] * OMX[1] + OMX[2] *
OMX[2]);

                UVBLTM1[0] = 0.7071 * (UVBL[2] - UVTM[2] - UVMX[2] * (UVBL[1] - UVTM[1]) /
UVMX[1]) / (UVMX[0] / UVMX[1] * (UVBL[1] * (UVTM[2] - UVMX[2] * UVTM[1] / UVMX[1]) -
UVTM[1] * (UVBL[2] - UVMX[2] * UVBL[1] / UVMX[1])) - UVBL[0] * (UVTM[2] - UVMX[2] *
UVTM[1] / UVMX[1]) + UVTM[0] * (UVBL[2] - UVMX[2] * UVBL[1] / UVMX[1]));
                UVBLTM1[2] = (0.7071 - UVBLTM1[0] * UVTM[0] + UVBLTM1[0] * UVMX[0] *
UVTM[1] / UVMX[1]) / (UVTM[2] - UVMX[2] * UVTM[1] / UVMX[1]);
                UVBLTM1[1] = (0 - UVBLTM1[0] * UVMX[0] - UVBLTM1[2] * UVMX[2]) /
UVMX[1];

                UVBLTM2[0] = (UVBLTM1[1] * UVMX[2]) - (UVBLTM1[2] * UVMX[1]);
                UVBLTM2[1] = (UVBLTM1[2] * UVMX[0]) - (UVBLTM1[0] * UVMX[2]);
                UVBLTM2[2] = (UVBLTM1[0] * UVMX[1]) - (UVBLTM1[1] * UVMX[0]);

                OUVBLTM1[0] = 0.7071 * (OUVBL[2] - OUVTM[2] - OUVMX[2] * (OUVBL[1] -
OUVTM[1]) / OUVMX[1]) / (OUVMX[0] / OUVMX[1] * (OUVBL[1] * (OUVTM[2] - OUVMX[2] *
OUVTM[1] / OUVMX[1]) - OUVTM[1] * (OUVBL[2] - OUVMX[2] * OUVBL[1] / OUVMX[1])) -
OUVBL[0] * (OUVTM[2] - OUVMX[2] * OUVTM[1] / OUVMX[1]) + OUVTM[0] * (OUVBL[2] -
OUVMX[2] * OUVBL[1] / OUVMX[1]));
                OUVBLTM1[2] = (0.7071 - OUVBLTM1[0] * OUVTM[0] + OUVBLTM1[0] *
OUVMX[0] * OUVTM[1] / OUVMX[1]) / (OUVTM[2] - OUVMX[2] * OUVTM[1] / OUVMX[1]);
                OUVBLTM1[1] = (0 - OUVBLTM1[0] * OUVMX[0] - OUVBLTM1[2] * OUVMX[2]) /
OUVMX[1];

                OUVBLTM2[0] = (OUVBLTM1[1] * OUVMX[2]) - (OUVBLTM1[2] * OUVMX[1]);
                OUVBLTM2[1] = (OUVBLTM1[2] * OUVMX[0]) - (OUVBLTM1[0] * OUVMX[2]);
                OUVBLTM2[2] = (OUVBLTM1[0] * OUVMX[1]) - (OUVBLTM1[1] * OUVMX[0]);

                UVTMMX1[0] = 0.7071 * (UVTM[2] - UVMX[2] - UVBL[2] * (UVTM[1] - UVMX[1]) /
UVBL[1]) / (UVBL[0] / UVBL[1] * (UVTM[1] * (UVMX[2] - UVBL[2] * UVMX[1] / UVBL[1]) -
UVMX[1] * (UVTM[2] - UVBL[2] * UVTM[1] / UVBL[1])) - UVTM[0] * (UVMX[2] - UVBL[2] *
UVMX[1] / UVBL[1]) + UVMX[0] * (UVTM[2] - UVBL[2] * UVTM[1] / UVBL[1]));
```

C++ Program (Continued)

```cpp
            UVTMMX1[2] = (0.7071 - UVTMMX1[0] * UVMX[0] + UVTMMX1[0] * UVBL[0] *
UVMX[1] / UVBL[1]) / (UVMX[2] - UVBL[2] * UVMX[1] / UVBL[1]);
            UVTMMX1[1] = (0 - UVTMMX1[0] * UVBL[0] - UVTMMX1[2] * UVBL[2]) /
UVBL[1];

            UVTMMX2[0] = (UVTMMX1[1] * UVBL[2]) - (UVTMMX1[2] * UVBL[1]);
            UVTMMX2[1] = (UVTMMX1[2] * UVBL[0]) - (UVTMMX1[0] * UVBL[2]);
            UVTMMX2[2] = (UVTMMX1[0] * UVBL[1]) - (UVTMMX1[1] * UVBL[0]);

            OUVTMMX1[0] = 0.7071 * (OUVTM[2] - OUVMX[2] - OUVBL[2] * (OUVTM[1] -
OUVMX[1]) / OUVBL[1]) / (OUVBL[0] / OUVBL[1] * (OUVTM[1] * (OUVMX[2] - OUVBL[2] *
OUVMX[1] / OUVBL[1]) - OUVMX[1] * (OUVTM[2] - OUVBL[2] * OUVTM[1] / OUVBL[1])) -
OUVTM[0] * (OUVMX[2] - OUVBL[2] * OUVMX[1] / OUVBL[1]) + OUVMX[0] * (OUVTM[2] -
OUVBL[2] * OUVTM[1] / OUVBL[1]));
            OUVTMMX1[2] = (0.7071 - OUVTMMX1[0] * OUVMX[0] + OUVTMMX1[0] *
OUVBL[0] * OUVMX[1] / OUVBL[1]) / (OUVMX[2] - OUVBL[2] * OUVMX[1] / OUVBL[1]);
            OUVTMMX1[1] = (0 - OUVTMMX1[0] * OUVBL[0] - OUVTMMX1[2] * OUVBL[2]) /
OUVBL[1];

            OUVTMMX2[0] = (OUVTMMX1[1] * OUVBL[2]) - (OUVTMMX1[2] * OUVBL[1]);
            OUVTMMX2[1] = (OUVTMMX1[2] * OUVBL[0]) - (OUVTMMX1[0] * OUVBL[2]);
            OUVTMMX2[2] = (OUVTMMX1[0] * OUVBL[1]) - (OUVTMMX1[1] * OUVBL[0]);

            UVMXBL1[0] = 0.7071 * (UVMX[2] - UVBL[2] - UVTM[2] * (UVMX[1] - UVBL[1]) /
UVTM[1]) / (UVTM[0] / UVTM[1] * (UVMX[1] * (UVBL[2] - UVTM[2] * UVBL[1] / UVTM[1]) -
UVBL[1] * (UVMX[2] - UVTM[2] * UVMX[1] / UVTM[1])) - UVMX[0] * (UVBL[2] - UVTM[2] *
UVBL[1] / UVTM[1]) + UVBL[0] * (UVMX[2] - UVTM[2] * UVMX[1] / UVTM[1]));
            UVMXBL1[2] = (0.7071 - UVMXBL1[0] * UVBL[0] + UVMXBL1[0] * UVTM[0] *
UVBL[1] / UVTM[1]) / (UVBL[2] - UVTM[2] * UVBL[1] / UVTM[1]);
            UVMXBL1[1] = (0 - UVMXBL1[0] * UVTM[0] - UVMXBL1[2] * UVTM[2]) /
UVTM[1];

            UVMXBL2[0] = (UVMXBL1[1] * UVTM[2]) - (UVMXBL1[2] * UVTM[1]);
            UVMXBL2[1] = (UVMXBL1[2] * UVTM[0]) - (UVMXBL1[0] * UVTM[2]);
            UVMXBL2[2] = (UVMXBL1[0] * UVTM[1]) - (UVMXBL1[1] * UVTM[0]);

            OUVMXBL1[0] = 0.7071 * (OUVMX[2] - OUVBL[2] - OUVTM[2] * (OUVMX[1] -
OUVBL[1]) / OUVTM[1]) / (OUVTM[0] / OUVTM[1] * (OUVMX[1] * (OUVBL[2] - OUVTM[2] *
OUVBL[1] / OUVTM[1]) - OUVBL[1] * (OUVMX[2] - OUVTM[2] * OUVMX[1] / OUVTM[1])) -
OUVMX[0] * (OUVBL[2] - OUVTM[2] * OUVBL[1] / OUVTM[1]) + OUVBL[0] * (OUVMX[2] -
OUVTM[2] * OUVMX[1] / OUVTM[1]));
            OUVMXBL1[2] = (0.7071 - OUVMXBL1[0] * OUVBL[0] + OUVMXBL1[0] *
OUVTM[0] * OUVBL[1] / OUVTM[1]) / (OUVBL[2] - OUVTM[2] * OUVBL[1] / OUVTM[1]);
            OUVMXBL1[1] = (0 - OUVMXBL1[0] * OUVTM[0] - OUVMXBL1[2] * OUVTM[2]) /
OUVTM[1];

            OUVMXBL2[0] = (OUVMXBL1[1] * OUVTM[2]) - (OUVMXBL1[2] * OUVTM[1]);
            OUVMXBL2[1] = (OUVMXBL1[2] * OUVTM[0]) - (OUVMXBL1[0] * OUVTM[2]);
            OUVMXBL2[2] = (OUVMXBL1[0] * OUVTM[1]) - (OUVMXBL1[1] * OUVTM[0]);
```

C++ Program (Continued)

```
                TUVBLTM1[0] = UVBLTM1[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVBLTM1[1] = UVBLTM1[1];
                TUVBLTM1[2] = UVBLTM1[2];

                TUVBLTM2[0] = UVBLTM2[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVBLTM2[1] = UVBLTM2[1];
                TUVBLTM2[2] = UVBLTM2[2];

                TUVTMMX1[0] = UVTMMX1[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVTMMX1[1] = UVTMMX1[1];
                TUVTMMX1[2] = UVTMMX1[2];

                TUVTMMX2[0] = UVTMMX2[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVTMMX2[1] = UVTMMX2[1];
                TUVTMMX2[2] = UVTMMX2[2];

                TUVMXBL1[0] = UVMXBL1[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVMXBL1[1] = UVMXBL1[1];
                TUVMXBL1[2] = UVMXBL1[2];

                TUVMXBL2[0] = UVMXBL2[0]; // temporarily hold "UV" values for use in assuring UV
status in the next step
                TUVMXBL2[1] = UVMXBL2[1];
                TUVMXBL2[2] = UVMXBL2[2];

                TOUVBLTM1[0] = OUVBLTM1[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVBLTM1[1] = OUVBLTM1[1];
                TOUVBLTM1[2] = OUVBLTM1[2];

                TOUVBLTM2[0] = OUVBLTM2[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVBLTM2[1] = OUVBLTM2[1];
                TOUVBLTM2[2] = OUVBLTM2[2];

                TOUVTMMX1[0] = OUVTMMX1[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVTMMX1[1] = OUVTMMX1[1];
                TOUVTMMX1[2] = OUVTMMX1[2];

                TOUVTMMX2[0] = OUVTMMX2[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVTMMX2[1] = OUVTMMX2[1];
                TOUVTMMX2[2] = OUVTMMX2[2];
```

C++ Program (Continued)

```
                TOUVMXBL1[0] = OUVMXBL1[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVMXBL1[1] = OUVMXBL1[1];
                TOUVMXBL1[2] = OUVMXBL1[2];

                TOUVMXBL2[0] = OUVMXBL2[0]; // temporarily hold "UV" values for use in assuring
UV status in the next step
                TOUVMXBL2[1] = OUVMXBL2[1];
                TOUVMXBL2[2] = OUVMXBL2[2];


                UVBLTM1[0] = UVBLTM1[0] / sqrt(TUVBLTM1[0] * TUVBLTM1[0] + TUVBLTM1[1]
* TUVBLTM1[1] + TUVBLTM1[2] * TUVBLTM1[2]);  // Reassuring unit vector length
                UVBLTM1[1] = UVBLTM1[1] / sqrt(TUVBLTM1[0] * TUVBLTM1[0] + TUVBLTM1[1]
* TUVBLTM1[1] + TUVBLTM1[2] * TUVBLTM1[2]);
                UVBLTM1[2] = UVBLTM1[2] / sqrt(TUVBLTM1[0] * TUVBLTM1[0] + TUVBLTM1[1]
* TUVBLTM1[1] + TUVBLTM1[2] * TUVBLTM1[2]);


                UVBLTM2[0] = UVBLTM2[0] / sqrt(TUVBLTM2[0] * TUVBLTM2[0] + TUVBLTM2[1]
* TUVBLTM2[1] + TUVBLTM2[2] * TUVBLTM2[2]);  // Reassuring unit vector length
                UVBLTM2[1] = UVBLTM2[1] / sqrt(TUVBLTM2[0] * TUVBLTM2[0] + TUVBLTM2[1]
* TUVBLTM2[1] + TUVBLTM2[2] * TUVBLTM2[2]);
                UVBLTM2[2] = UVBLTM2[2] / sqrt(TUVBLTM2[0] * TUVBLTM2[0] + TUVBLTM2[1]
* TUVBLTM2[1] + TUVBLTM2[2] * TUVBLTM2[2]);


                UVTMMX1[0] = UVTMMX1[0] / sqrt(TUVTMMX1[0] * TUVTMMX1[0] +
TUVTMMX1[1] * TUVTMMX1[1] + TUVTMMX1[2] * TUVTMMX1[2]);  // Reassuring unit vector length
                UVTMMX1[1] = UVTMMX1[1] / sqrt(TUVTMMX1[0] * TUVTMMX1[0] +
TUVTMMX1[1] * TUVTMMX1[1] + TUVTMMX1[2] * TUVTMMX1[2]);
                UVTMMX1[2] = UVTMMX1[2] / sqrt(TUVTMMX1[0] * TUVTMMX1[0] +
TUVTMMX1[1] * TUVTMMX1[1] + TUVTMMX1[2] * TUVTMMX1[2]);


                UVTMMX2[0] = UVTMMX2[0] / sqrt(TUVTMMX2[0] * TUVTMMX2[0] +
TUVTMMX2[1] * TUVTMMX2[1] + TUVTMMX2[2] * TUVTMMX2[2]);  // Reassuring unit vector length
                UVTMMX2[1] = UVTMMX2[1] / sqrt(TUVTMMX2[0] * TUVTMMX2[0] +
TUVTMMX2[1] * TUVTMMX2[1] + TUVTMMX2[2] * TUVTMMX2[2]);
                UVTMMX2[2] = UVTMMX2[2] / sqrt(TUVTMMX2[0] * TUVTMMX2[0] +
TUVTMMX2[1] * TUVTMMX2[1] + TUVTMMX2[2] * TUVTMMX2[2]);


                UVMXBL1[0] = UVMXBL1[0] / sqrt(TUVMXBL1[0] * TUVMXBL1[0] +
TUVMXBL1[1] * TUVMXBL1[1] + TUVMXBL1[2] * TUVMXBL1[2]);  // Reassuring unit vector length
                UVMXBL1[1] = UVMXBL1[1] / sqrt(TUVMXBL1[0] * TUVMXBL1[0] +
TUVMXBL1[1] * TUVMXBL1[1] + TUVMXBL1[2] * TUVMXBL1[2]);
                UVMXBL1[2] = UVMXBL1[2] / sqrt(TUVMXBL1[0] * TUVMXBL1[0] +
TUVMXBL1[1] * TUVMXBL1[1] + TUVMXBL1[2] * TUVMXBL1[2]);


                UVMXBL2[0] = UVMXBL2[0] / sqrt(TUVMXBL2[0] * TUVMXBL2[0] +
TUVMXBL2[1] * TUVMXBL2[1] + TUVMXBL2[2] * TUVMXBL2[2]);  // Reassuring unit vector length
                UVMXBL2[1] = UVMXBL2[1] / sqrt(TUVMXBL2[0] * TUVMXBL2[0] +
TUVMXBL2[1] * TUVMXBL2[1] + TUVMXBL2[2] * TUVMXBL2[2]);
```

C++ Program (Continued)

```
        UVMXBL2[2] = UVMXBL2[2] / sqrt(TUVMXBL2[0] * TUVMXBL2[0] +
TUVMXBL2[1] * TUVMXBL2[1] + TUVMXBL2[2] * TUVMXBL2[2]);

        OUVBLTM1[0] = OUVBLTM1[0] / sqrt(TOUVBLTM1[0] * TOUVBLTM1[0] +
TOUVBLTM1[1] * TOUVBLTM1[1] + TOUVBLTM1[2] * TOUVBLTM1[2]);  // Reassuring unit vector
length
        OUVBLTM1[1] = OUVBLTM1[1] / sqrt(TOUVBLTM1[0] * TOUVBLTM1[0] +
TOUVBLTM1[1] * TOUVBLTM1[1] + TOUVBLTM1[2] * TOUVBLTM1[2]);
        OUVBLTM1[2] = OUVBLTM1[2] / sqrt(TOUVBLTM1[0] * TOUVBLTM1[0] +
TOUVBLTM1[1] * TOUVBLTM1[1] + TOUVBLTM1[2] * TOUVBLTM1[2]);

        OUVBLTM2[0] = OUVBLTM2[0] / sqrt(TOUVBLTM2[0] * TOUVBLTM2[0] +
TOUVBLTM2[1] * TOUVBLTM2[1] + TOUVBLTM2[2] * TOUVBLTM2[2]);  // Reassuring unit vector
length
        OUVBLTM2[1] = OUVBLTM2[1] / sqrt(TOUVBLTM2[0] * TOUVBLTM2[0] +
TOUVBLTM2[1] * TOUVBLTM2[1] + TOUVBLTM2[2] * TOUVBLTM2[2]);
        OUVBLTM2[2] = OUVBLTM2[2] / sqrt(TOUVBLTM2[0] * TOUVBLTM2[0] +
TOUVBLTM2[1] * TOUVBLTM2[1] + TOUVBLTM2[2] * TOUVBLTM2[2]);

        OUVTMMX1[0] = OUVTMMX1[0] / sqrt(TOUVTMMX1[0] * TOUVTMMX1[0] +
TOUVTMMX1[1] * TOUVTMMX1[1] + TOUVTMMX1[2] * TOUVTMMX1[2]);  // Reassuring unit
vector length
        OUVTMMX1[1] = OUVTMMX1[1] / sqrt(TOUVTMMX1[0] * TOUVTMMX1[0] +
TOUVTMMX1[1] * TOUVTMMX1[1] + TOUVTMMX1[2] * TOUVTMMX1[2]);
        OUVTMMX1[2] = OUVTMMX1[2] / sqrt(TOUVTMMX1[0] * TOUVTMMX1[0] +
TOUVTMMX1[1] * TOUVTMMX1[1] + TOUVTMMX1[2] * TOUVTMMX1[2]);

        OUVTMMX2[0] = OUVTMMX2[0] / sqrt(TOUVTMMX2[0] * TOUVTMMX2[0] +
TOUVTMMX2[1] * TOUVTMMX2[1] + TOUVTMMX2[2] * TOUVTMMX2[2]);  // Reassuring unit
vector length
        OUVTMMX2[1] = OUVTMMX2[1] / sqrt(TOUVTMMX2[0] * TOUVTMMX2[0] +
TOUVTMMX2[1] * TOUVTMMX2[1] + TOUVTMMX2[2] * TOUVTMMX2[2]);
        OUVTMMX2[2] = OUVTMMX2[2] / sqrt(TOUVTMMX2[0] * TOUVTMMX2[0] +
TOUVTMMX2[1] * TOUVTMMX2[1] + TOUVTMMX2[2] * TOUVTMMX2[2]);

        OUVMXBL1[0] = OUVMXBL1[0] / sqrt(TOUVMXBL1[0] * TOUVMXBL1[0] +
TOUVMXBL1[1] * TOUVMXBL1[1] + TOUVMXBL1[2] * TOUVMXBL1[2]);  // Reassuring unit vector
length
        OUVMXBL1[1] = OUVMXBL1[1] / sqrt(TOUVMXBL1[0] * TOUVMXBL1[0] +
TOUVMXBL1[1] * TOUVMXBL1[1] + TOUVMXBL1[2] * TOUVMXBL1[2]);
        OUVMXBL1[2] = OUVMXBL1[2] / sqrt(TOUVMXBL1[0] * TOUVMXBL1[0] +
TOUVMXBL1[1] * TOUVMXBL1[1] + TOUVMXBL1[2] * TOUVMXBL1[2]);

        OUVMXBL2[0] = OUVMXBL2[0] / sqrt(TOUVMXBL2[0] * TOUVMXBL2[0] +
TOUVMXBL2[1] * TOUVMXBL2[1] + TOUVMXBL2[2] * TOUVMXBL2[2]);  // Reassuring unit vector
length
        OUVMXBL2[1] = OUVMXBL2[1] / sqrt(TOUVMXBL2[0] * TOUVMXBL2[0] +
TOUVMXBL2[1] * TOUVMXBL2[1] + TOUVMXBL2[2] * TOUVMXBL2[2]);
        OUVMXBL2[2] = OUVMXBL2[2] / sqrt(TOUVMXBL2[0] * TOUVMXBL2[0] +
TOUVMXBL2[1] * TOUVMXBL2[1] + TOUVMXBL2[2] * TOUVMXBL2[2]);
```

C++ Program (Continued)

```cpp
                    cout << "UVBL " << (UVBL[0]) << "    " << (UVBL[1]) << "    " << (UVBL[2]) << "
\n";
                    cout << "UVTM " << (UVTM[0]) << "    " << (UVTM[1]) << "    " << (UVTM[2]) << "
\n";
                    cout << "UVMX " << (UVMX[0]) << "    " << (UVMX[1]) << "    " << (UVMX[2]) << "
\n";
                    cout << "BLTM1 " << (UVBLTM1[0]) << "    " << (UVBLTM1[1]) << "    " <<
(UVBLTM1[2]) << "    \n";
                    cout << "BLTM2 " << (UVBLTM2[0]) << "    " << (UVBLTM2[1]) << "    " <<
(UVBLTM2[2]) << "    \n";
//                  cout << "TMMX1 " << (UVTMMX1[0]) << "    " << (UVTMMX1[1]) << "    " <<
(UVTMMX1[2]) << "    \n";
//                  cout << "TMMX2 " << (UVTMMX2[0]) << "    " << (UVTMMX2[1]) << "    " <<
(UVTMMX2[2]) << "    \n";
//                  cout << "MXBL1 " << (UVMXBL1[0]) << "    " << (UVMXBL1[1]) << "    " <<
(UVMXBL1[2]) << "    \n";
//                  cout << "MXBL2 " << (UVMXBL2[0]) << "    " << (UVMXBL2[1]) << "    " <<
(UVMXBL2[2]) << "    \n";

                    cout << "OUVBL " << (OUVBL[0]) << "    " << (OUVBL[1]) << "    " << (OUVBL[2])
<< "    \n";
                    cout << "OUVTM " << (OUVTM[0]) << "    " << (OUVTM[1]) << "    " << (OUVTM[2])
<< "    \n";
                    cout << "OUVMX " << (OUVMX[0]) << "    " << (OUVMX[1]) << "    " <<
(OUVMX[2]) << "    \n";
                    cout << "OBLTM1 " << (OUVBLTM1[0]) << "    " << (OUVBLTM1[1]) << "    " <<
(OUVBLTM1[2]) << "    \n";
                    cout << "OBLTM2 " << (OUVBLTM2[0]) << "    " << (OUVBLTM2[1]) << "    " <<
(OUVBLTM2[2]) << "    \n";
//                  cout << "OTMMX1 " << (OUVTMMX1[0]) << "    " << (OUVTMMX1[1]) << "    " <<
(OUVTMMX1[2]) << "    \n";
//                  cout << "OTMMX2 " << (OUVTMMX2[0]) << "    " << (OUVTMMX2[1]) << "    " <<
(OUVTMMX2[2]) << "    \n";
//                  cout << "OMXBL1 " << (OUVMXBL1[0]) << "    " << (OUVMXBL1[1]) << "    " <<
(OUVMXBL1[2]) << "    \n";
//                  cout << "OMXBL2 " << (OUVMXBL2[0]) << "    " << (OUVMXBL2[1]) << "    " <<
(OUVMXBL2[2]) << "    \n";

// ---------------------------- THETA Angle Change -----------------------------------------

                    thetadotBL = fabs(UVBL[0] * XcentralUV[0] + UVBL[1] * XcentralUV[1]); // dot product
of BL and Xcentral unit vectors
                    thetadotTM = fabs(UVTM[0] * XcentralUV[0] + UVTM[1] * XcentralUV[1]); // of TM
and
                    thetadotMX = fabs(UVMX[0] * XcentralUV[0] + UVMX[1] * XcentralUV[1]); // of MX,
ETC
                    thetadotBLTM1 = fabs(UVBLTM1[0] * XcentralUV[0] + UVBLTM1[1] * XcentralUV[1]);
                    thetadotBLTM2 = fabs(UVBLTM2[0] * XcentralUV[0] + UVBLTM2[1] * XcentralUV[1]);
```

C++ Program (Continued)

```cpp
                thetadotTMMX1 = fabs(UVTMMX1[0] * XcentralUV[0] + UVTMMX1[1] *
XcentralUV[1]);
                thetadotTMMX2 = fabs(UVTMMX2[0] * XcentralUV[0] + UVTMMX2[1] *
XcentralUV[1]);
                thetadotMXBL1 = fabs(UVMXBL1[0] * XcentralUV[0] + UVMXBL1[1] *
XcentralUV[1]);
                thetadotMXBL2 = fabs(UVMXBL2[0] * XcentralUV[0] + UVMXBL2[1] *
XcentralUV[1]);


                thetadotOBL = fabs(OUVBL[0] * XcentralUV[0] + OUVBL[1] * XcentralUV[1]); // dot
product of OBL and Xcentral unit vectors
                thetadotOTM = fabs(OUVTM[0] * XcentralUV[0] + OUVTM[1] * XcentralUV[1]); // of
OTM and
                thetadotOMX = fabs(OUVMX[0] * XcentralUV[0] + OUVMX[1] * XcentralUV[1]); // of
OMX and
                thetadotOBLTM1 = fabs(OUVBLTM1[0] * XcentralUV[0] + OUVBLTM1[1] *
XcentralUV[1]);
                thetadotOBLTM2 = fabs(OUVBLTM2[0] * XcentralUV[0] + OUVBLTM2[1] *
XcentralUV[1]);
                thetadotOTMMX1 = fabs(OUVTMMX1[0] * XcentralUV[0] + OUVTMMX1[1] *
XcentralUV[1]);
                thetadotOTMMX2 = fabs(OUVTMMX2[0] * XcentralUV[0] + OUVTMMX2[1] *
XcentralUV[1]);
                thetadotOMXBL1 = fabs(OUVMXBL1[0] * XcentralUV[0] + OUVMXBL1[1] *
XcentralUV[1]);
                thetadotOMXBL2 = fabs(OUVMXBL2[0] * XcentralUV[0] + OUVMXBL2[1] *
XcentralUV[1]);

                cout << "thetadotTMMX1 " << (thetadotTMMX1) << "     \n";
                cout << "thetadotOTMMX1 " << (thetadotOTMMX1) << "     \n";

                if (thetadotBL > thetadotOBL) // lets figure out if BL defines angle theta worse than OBL
                {
                        WthetadotBL = thetadotBL;
                }
                else
                {
                        WthetadotBL = thetadotOBL; // or if visa versa
                }


                if (thetadotTM > thetadotOTM) // same as above loop but for TM and OTM
                {
                        WthetadotTM = thetadotTM;
                }
                else
                {
                        WthetadotTM = thetadotOTM;
                }
```

C++ Program (Continued)

```cpp
if (thetadotMX > thetadotOMX) // same as above loop but for MX and OMX
{
        WthetadotMX = thetadotMX;
}
else
{
        WthetadotMX = thetadotOMX;
}


if (thetadotBLTM1 > thetadotOBLTM1) // same as above with new vectors
{
        WthetadotBLTM1 = thetadotBLTM1;
}
else
{
        WthetadotBLTM1 = thetadotOBLTM1;
}


if (thetadotBLTM2 > thetadotOBLTM2) // same as above with new vectors
{
        WthetadotBLTM2 = thetadotBLTM2;
}
else
{
        WthetadotBLTM2 = thetadotOBLTM2;
}


if (thetadotTMMX1 > thetadotOTMMX1) // same as above with new vectors
{
        WthetadotTMMX1 = thetadotTMMX1;
}
else
{
        WthetadotTMMX1 = thetadotOTMMX1;
}


if (thetadotTMMX2 > thetadotOTMMX2) // same as above with new vectors
{
        WthetadotTMMX2 = thetadotTMMX2;
}
else
{
        WthetadotTMMX2 = thetadotOTMMX2;
}


if (thetadotMXBL1 > thetadotOMXBL1) // same as above with new vectors
```

C++ Program (Continued)

```
            {
                    WthetadotMXBL1 = thetadotMXBL1;
            }
            else
            {
                    WthetadotMXBL1 = thetadotOMXBL1;
            }


            if (thetadotMXBL2 > thetadotOMXBL2) // same as above with new vectors
            {
                    WthetadotMXBL2 = thetadotMXBL2;
            }
            else
            {
                    WthetadotMXBL2 = thetadotOMXBL2;
            }


            if ( WthetadotBL <= WthetadotTM && WthetadotBL <= WthetadotMX && WthetadotBL
<= WthetadotBLTM1 && WthetadotBL <= WthetadotBLTM2 && WthetadotBL <= WthetadotTMMX1 &&
WthetadotBL <= WthetadotTMMX2 && WthetadotBL <= WthetadotMXBL1 && WthetadotBL <=
WthetadotMXBL2) // means BL gets to define theta angles for this phase transition
            {
                    RXangle = atan2(centralUV[1], centralUV[0]); // how many degrees around a
vertical axis is radial vector from X axis
                    CXangle = atan2(UVBL[1], UVBL[0]); // how many degrees around vertical axis
is BL vector from X axis
                    OXangle = atan2(OUVBL[1], OUVBL[0]); // how many degrees around vertical
axis is OBL vector from X axis

                    Cadjacent = cos(CXangle - RXangle) * sqrt(UVBL[0] * UVBL[0] + UVBL[1] *
UVBL[1]); // find adjacent piece of current angle
                    Oadjacent = cos(OXangle - RXangle) * sqrt(OUVBL[0] * OUVBL[0] +
OUVBL[1] * OUVBL[1]); // find adjacent piece of old angle

                    Copp = UVBL[2];
                    Oopp = OUVBL[2];

                    Ctheta = atan2(Copp,Cadjacent);
                    Otheta = atan2(Oopp,Oadjacent);

                    theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14; // finds the change in
theta in this phase transition. remember, phases starts

                            // out equal to 2 and count equal to 0 on the first loop.
            }
```

C++ Program (Continued)

```
                if ( WthetadotTM <= WthetadotMX && WthetadotTM <= WthetadotBL &&
WthetadotTM <= WthetadotBLTM1 && WthetadotTM <= WthetadotBLTM2 && WthetadotTM <=
WthetadotTMMX1 && WthetadotTM <= WthetadotTMMX2 && WthetadotTM <= WthetadotMXBL1 &&
WthetadotTM <= WthetadotMXBL2) // Same as above loop but TM defines theta, not BL
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVTM[1], UVTM[0]);
                        OXangle = atan2(OUVTM[1], OUVTM[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVTM[0] * UVTM[0] + UVTM[1] *
UVTM[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVTM[0] * OUVTM[0] +
OUVTM[1] * OUVTM[1]);

                        Copp = UVTM[2];
                        Oopp = OUVTM[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }

                if ( WthetadotMX < WthetadotBL && WthetadotMX <= WthetadotTM && WthetadotMX
<= WthetadotBLTM1 && WthetadotMX <= WthetadotBLTM2 && WthetadotMX <= WthetadotTMMX1
&& WthetadotMX <= WthetadotTMMX2 && WthetadotMX <= WthetadotMXBL1 && WthetadotMX <=
WthetadotMXBL2) // Same as above loop but MX defines theta, not TM
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVMX[1], UVMX[0]);
                        OXangle = atan2(OUVMX[1], OUVMX[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVMX[0] * UVMX[0] + UVMX[1]
* UVMX[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVMX[0] * OUVMX[0] +
OUVMX[1] * OUVMX[1]);

                        Copp = UVMX[2];
                        Oopp = OUVMX[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }
```

C++ Program (Continued)

```
if ( WthetadotBLTM1 < WthetadotBL && WthetadotBLTM1 <= WthetadotTM &&
WthetadotBLTM1 <= WthetadotMX && WthetadotBLTM1 <= WthetadotBLTM2 && WthetadotBLTM1
<= WthetadotTMMX1 && WthetadotBLTM1 <= WthetadotTMMX2 && WthetadotBLTM1 <=
WthetadotMXBL1 && WthetadotBLTM1 <= WthetadotMXBL2) // Same as above loop
        {
                RXangle = atan2(centralUV[1], centralUV[0]);
                CXangle = atan2(UVBLTM1[1], UVBLTM1[0]);
                OXangle = atan2(OUVBLTM1[1], OUVBLTM1[0]);

                Cadjacent = cos(CXangle - RXangle) * sqrt(UVBLTM1[0] * UVBLTM1[0] +
UVBLTM1[1] * UVBLTM1[1]);
                Oadjacent = cos(OXangle - RXangle) * sqrt(OUVBLTM1[0] * OUVBLTM1[0] +
OUVBLTM1[1] * OUVBLTM1[1]);

                Copp = UVBLTM1[2];
                Oopp = OUVBLTM1[2];

                Ctheta = atan2(Copp,Cadjacent);
                Otheta = atan2(Oopp,Oadjacent);

                theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

        }

        if ( WthetadotBLTM2 < WthetadotBL && WthetadotBLTM2 <= WthetadotTM &&
WthetadotBLTM2 <= WthetadotBLTM1 && WthetadotBLTM2 <= WthetadotMX && WthetadotBLTM2
<= WthetadotTMMX1 && WthetadotBLTM2 <= WthetadotTMMX2 && WthetadotBLTM2 <=
WthetadotMXBL1 && WthetadotBLTM2 <= WthetadotMXBL2) // Same as above loop
        {
                RXangle = atan2(centralUV[1], centralUV[0]);
                CXangle = atan2(UVBLTM2[1], UVBLTM2[0]);
                OXangle = atan2(OUVBLTM2[1], OUVBLTM2[0]);

                Cadjacent = cos(CXangle - RXangle) * sqrt(UVBLTM2[0] * UVBLTM2[0] +
UVBLTM2[1] * UVBLTM2[1]);
                Oadjacent = cos(OXangle - RXangle) * sqrt(OUVBLTM2[0] * OUVBLTM2[0] +
OUVBLTM2[1] * OUVBLTM2[1]);

                Copp = UVBLTM2[2];
                Oopp = OUVBLTM2[2];

                Ctheta = atan2(Copp,Cadjacent);
                Otheta = atan2(Oopp,Oadjacent);

                theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

        }
```

C++ Program (Continued)

```cpp
                if ( WthetadotTMMX1 < WthetadotBL && WthetadotTMMX1 <= WthetadotTM &&
WthetadotTMMX1 <= WthetadotBLTM1 && WthetadotTMMX1 <= WthetadotBLTM2 &&
WthetadotTMMX1 <= WthetadotMX && WthetadotTMMX1 <= WthetadotTMMX2 &&
WthetadotTMMX1 <= WthetadotMXBL1 && WthetadotTMMX1 <= WthetadotMXBL2) // Same as above
loop
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVTMMX1[1], UVTMMX1[0]);
                        OXangle = atan2(OUVTMMX1[1], OUVTMMX1[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVTMMX1[0] * UVTMMX1[0] +
UVTMMX1[1] * UVTMMX1[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVTMMX1[0] * OUVTMMX1[0]
+ OUVTMMX1[1] * OUVTMMX1[1]);

                        Copp = UVTMMX1[2];
                        Oopp = OUVTMMX1[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }


                if ( WthetadotTMMX2 < WthetadotBL && WthetadotTMMX2 <= WthetadotTM &&
WthetadotTMMX2 <= WthetadotBLTM1 && WthetadotTMMX2 <= WthetadotBLTM2 &&
WthetadotTMMX2 <= WthetadotTMMX1 && WthetadotTMMX2 <= WthetadotMX &&
WthetadotTMMX2 <= WthetadotMXBL1 && WthetadotTMMX2 <= WthetadotMXBL2) // Same as above
loop
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVTMMX2[1], UVTMMX2[0]);
                        OXangle = atan2(OUVTMMX2[1], OUVTMMX2[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVTMMX2[0] * UVTMMX2[0] +
UVTMMX2[1] * UVTMMX2[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVTMMX2[0] * OUVTMMX2[0]
+ OUVTMMX2[1] * OUVTMMX2[1]);

                        Copp = UVTMMX2[2];
                        Oopp = OUVTMMX2[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }
```

C++ Program (Continued)

```cpp
                if ( WthetadotMXBL1 < WthetadotBL && WthetadotMXBL1 <= WthetadotTM &&
WthetadotMXBL1 <= WthetadotBLTM1 && WthetadotMXBL1 <= WthetadotBLTM2 &&
WthetadotMXBL1 <= WthetadotTMMX1 && WthetadotMXBL1 <= WthetadotTMMX2 &&
WthetadotMXBL1 <= WthetadotMX && WthetadotMXBL1 <= WthetadotMXBL2) // Same as above loop
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVMXBL1[1], UVMXBL1[0]);
                        OXangle = atan2(OUVMXBL1[1], OUVMXBL1[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVMXBL1[0] * UVMXBL1[0] +
UVMXBL1[1] * UVMXBL1[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVMXBL1[0] * OUVMXBL1[0] +
OUVMXBL1[1] * OUVMXBL1[1]);

                        Copp = UVMXBL1[2];
                        Oopp = OUVMXBL1[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }


                if ( WthetadotMXBL2 < WthetadotBL && WthetadotMXBL2 <= WthetadotTM &&
WthetadotMXBL2 <= WthetadotBLTM1 && WthetadotMXBL2 <= WthetadotBLTM2 &&
WthetadotMXBL2 <= WthetadotTMMX1 && WthetadotMXBL2 <= WthetadotTMMX2 &&
WthetadotMXBL2 <= WthetadotMXBL1 && WthetadotMXBL2 <= WthetadotMX) // Same as above loop
                {
                        RXangle = atan2(centralUV[1], centralUV[0]);
                        CXangle = atan2(UVMXBL2[1], UVMXBL2[0]);
                        OXangle = atan2(OUVMXBL2[1], OUVMXBL2[0]);

                        Cadjacent = cos(CXangle - RXangle) * sqrt(UVMXBL2[0] * UVMXBL2[0] +
UVMXBL2[1] * UVMXBL2[1]);
                        Oadjacent = cos(OXangle - RXangle) * sqrt(OUVMXBL2[0] * OUVMXBL2[0] +
OUVMXBL2[1] * OUVMXBL2[1]);

                        Copp = UVMXBL2[2];
                        Oopp = OUVMXBL2[2];

                        Ctheta = atan2(Copp,Cadjacent);
                        Otheta = atan2(Oopp,Oadjacent);

                        theta[phase - 1][countb] = (Ctheta - Otheta) * 180 / 3.14;

                }


                if (fabs(theta[phase - 1][countb]) > 90 && fabs(theta[phase - 1][countb]) <= 270)
```

C++ Program (Continued)

```
                {
                        theta[phase - 1][countb] = fabs(theta[phase - 1][countb]) - 180;
                }  // This loop ensures that a change in angle of 90 to 270 degrees is not allowed but is
instead understood as a reverse in direction (180 rotation) of the reference vector

                if (fabs(theta[phase - 1][countb]) > 270)
                {
                        theta[phase - 1][countb] = (fabs(theta[phase - 1][countb]) * fabs(theta[phase -
1][countb]) / theta[phase - 1][countb]) - (360 * fabs(theta[phase - 1][countb]) / theta[phase - 1][countb]);
                }  // This loop ensures that a change in angle of 270+ degrees is not allowed but is instead
taken as rotating the opposite direction


// ----------------------------------- ALPHA Angle Change -----------------------------------------------------------

                alphadotBL = fabs(UVBL[2]); // dot product of BL and vertical unit vectors
                alphadotTM = fabs(UVTM[2]); // of TM and
                alphadotMX = fabs(UVMX[2]); // of MX, etc
                alphadotBLTM1 = fabs(UVBLTM1[2]);
                alphadotBLTM2 = fabs(UVBLTM2[2]);
                alphadotTMMX1 = fabs(UVTMMX1[2]);
                alphadotTMMX2 = fabs(UVTMMX2[2]);
                alphadotMXBL1 = fabs(UVMXBL1[2]);
                alphadotMXBL2 = fabs(UVMXBL2[2]);

                alphadotOBL = fabs(OUVBL[2]); // dot product of OBL and vertical unit vectors
                alphadotOTM = fabs(OUVTM[2]); // of OTM and
                alphadotOMX = fabs(OUVMX[2]); // of OMX, etc
                alphadotOBLTM1 = fabs(OUVBLTM1[2]);
                alphadotOBLTM2 = fabs(OUVBLTM2[2]);
                alphadotOTMMX1 = fabs(OUVTMMX1[2]);
                alphadotOTMMX2 = fabs(OUVTMMX2[2]);
                alphadotOMXBL1 = fabs(OUVMXBL1[2]);
                alphadotOMXBL2 = fabs(OUVMXBL2[2]);

                cout << "alphadotTMMX1 " << (alphadotTMMX1) << "     \n";
                cout << "alphadotOTMMX1 " << (alphadotOTMMX1) << "     \n";


                if (alphadotBL > alphadotOBL) // lets figure out if BL defines angle alpha worse than OBL
                {
                        WalphadotBL = alphadotBL;
                }
                else
                {
                        WalphadotBL = alphadotOBL; // or if visa versa
                }


                if (alphadotTM > alphadotOTM) // same as above loop but for TM and OTM
                {
```

C++ Program (Continued)

```
                WalphadotTM = alphadotTM;
        }
        else
        {
                WalphadotTM = alphadotOTM;
        }


        if (alphadotMX > alphadotOMX) // same as above loop but for MX and OMX
        {
                WalphadotMX = alphadotMX;
        }
        else
        {
                WalphadotMX = alphadotOMX;
        }


        if (alphadotBLTM1 > alphadotOBLTM1) // same as above loop
        {
                WalphadotBLTM1 = alphadotBLTM1;
        }
        else
        {
                WalphadotBLTM1 = alphadotOBLTM1;
        }


        if (alphadotBLTM2 > alphadotOBLTM2) // same as above loop
        {
                WalphadotBLTM2 = alphadotBLTM2;
        }
        else
        {
                WalphadotBLTM2 = alphadotOBLTM2;
        }


        if (alphadotTMMX1 > alphadotOTMMX1) // same as above loop
        {
                WalphadotTMMX1 = alphadotTMMX1;
        }
        else
        {
                WalphadotTMMX1 = alphadotOTMMX1;
        }


        if (alphadotTMMX2 > alphadotOTMMX2) // same as above loop
        {
                WalphadotTMMX2 = alphadotTMMX2;
```

C++ Program (Continued)

```
        }
        else
        {
                WalphadotTMMX2 = alphadotOTMMX2;
        }


        if (alphadotMXBL1 > alphadotOMXBL1) // same as above loop
        {
                WalphadotMXBL1 = alphadotMXBL1;
        }
        else
        {
                WalphadotMXBL1 = alphadotOMXBL1;
        }


        if (alphadotMXBL2 > alphadotOMXBL2) // same as above loop
        {
                WalphadotMXBL2 = alphadotMXBL2;
        }
        else
        {
                WalphadotMXBL2 = alphadotOMXBL2;
        }


        if ( WalphadotBL <= WalphadotTM && WalphadotBL <= WalphadotMX &&
WalphadotBL <= WalphadotBLTM1 && WalphadotBL <= WalphadotBLTM2 && WalphadotBL <=
WalphadotTMMX1 && WalphadotBL <= WalphadotTMMX2 && WalphadotBL <= WalphadotMXBL1
&& WalphadotBL <= WalphadotMXBL2) // means BL gets to define alpha angles for this phase transition
        {

                Calpha = atan2(UVBL[1],UVBL[0]); // when rotating around a fixed z axis, angles
are easy to compute, just like 2d geometry
                Oalpha = atan2(OUVBL[1],OUVBL[0]);

                alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                        // out equal to 2 and count equal to 0 on the first loop.
        }

        if ( WalphadotTM <= WalphadotMX && WalphadotTM <= WalphadotBL &&
WalphadotTM <= WalphadotBLTM1 && WalphadotTM <= WalphadotBLTM2 && WalphadotTM <=
WalphadotTMMX1 && WalphadotTM <= WalphadotTMMX2 && WalphadotTM <= WalphadotMXBL1
&& WalphadotTM <= WalphadotMXBL2) // Same as above loop but TM defines alpha, not BL
        {

                Calpha = atan2(UVTM[1],UVTM[0]);
                Oalpha = atan2(OUVTM[1],OUVTM[0]);
```

C++ Program (Continued)

```cpp
                        alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                        // out equal to 2 and count equal to 0 on the first loop.

                }

                if ( WalphadotMX <= WalphadotBL && WalphadotMX <= WalphadotTM &&
WalphadotMX <= WalphadotBLTM1 && WalphadotMX <= WalphadotBLTM2 && WalphadotMX <=
WalphadotTMMX1 && WalphadotMX <= WalphadotTMMX2 && WalphadotMX <= WalphadotMXBL1
&& WalphadotMX <= WalphadotMXBL2) // Same as above loop but MX defines alpha, not TM
                {

                        Calpha = atan2(UVMX[1],UVMX[0]);
                        Oalpha = atan2(OUVMX[1],OUVMX[0]);

                        alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                        // out equal to 2 and count equal to 0 on the first loop.

                }

                if ( WalphadotBLTM1 <= WalphadotBL && WalphadotBLTM1 <= WalphadotTM &&
WalphadotBLTM1 <= WalphadotMX && WalphadotBLTM1 <= WalphadotBLTM2 && WalphadotBLTM1
<= WalphadotTMMX1 && WalphadotBLTM1 <= WalphadotTMMX2 && WalphadotBLTM1 <=
WalphadotMXBL1 && WalphadotBLTM1 <= WalphadotMXBL2) // Same as above loop but MX defines
alpha, not TM
                {

                        Calpha = atan2(UVBLTM1[1],UVBLTM1[0]);
                        Oalpha = atan2(OUVBLTM1[1],OUVBLTM1[0]);

                        alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                        // out equal to 2 and count equal to 0 on the first loop.

                }

                if ( WalphadotBLTM2 <= WalphadotBL && WalphadotBLTM2 <= WalphadotTM &&
WalphadotBLTM2 <= WalphadotBLTM1 && WalphadotBLTM2 <= WalphadotMX && WalphadotBLTM2
<= WalphadotTMMX1 && WalphadotBLTM2 <= WalphadotTMMX2 && WalphadotBLTM2 <=
WalphadotMXBL1 && WalphadotBLTM2 <= WalphadotMXBL2) // Same as above loop but MX defines
alpha, not TM
                {

                        Calpha = atan2(UVBLTM2[1],UVBLTM2[0]);
                        Oalpha = atan2(OUVBLTM2[1],OUVBLTM2[0]);
```

C++ Program (Continued)

```
                    alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                // out equal to 2 and count equal to 0 on the first loop.
            }

            if ( WalphadotTMMX1 <= WalphadotBL && WalphadotTMMX1 <= WalphadotTM &&
WalphadotTMMX1 <= WalphadotBLTM1 && WalphadotTMMX1 <= WalphadotBLTM2 &&
WalphadotTMMX1 <= WalphadotMX && WalphadotTMMX1 <= WalphadotTMMX2 &&
WalphadotTMMX1 <= WalphadotMXBL1 && WalphadotTMMX1 <= WalphadotMXBL2) // Same as
above loop but MX defines alpha, not TM
            {

                Calpha = atan2(UVTMMX1[1],UVTMMX1[0]);
                Oalpha = atan2(OUVTMMX1[1],OUVTMMX1[0]);

                    alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                // out equal to 2 and count equal to 0 on the first loop.
            }

            if ( WalphadotTMMX2 <= WalphadotBL && WalphadotTMMX2 <= WalphadotTM &&
WalphadotTMMX2 <= WalphadotBLTM1 && WalphadotTMMX2 <= WalphadotBLTM2 &&
WalphadotTMMX2 <= WalphadotTMMX1 && WalphadotTMMX2 <= WalphadotMX &&
WalphadotTMMX2 <= WalphadotMXBL1 && WalphadotTMMX2 <= WalphadotMXBL2) // Same as
above loop but MX defines alpha, not TM
            {

                Calpha = atan2(UVTMMX2[1],UVTMMX2[0]);
                Oalpha = atan2(OUVTMMX2[1],OUVTMMX2[0]);

                    alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in
alpha in this phase transition. remember, phases starts

                                // out equal to 2 and count equal to 0 on the first loop.
            }

            if ( WalphadotMXBL1 <= WalphadotBL && WalphadotMXBL1 <= WalphadotTM &&
WalphadotMXBL1 <= WalphadotBLTM1 && WalphadotMXBL1 <= WalphadotBLTM2 &&
WalphadotMXBL1 <= WalphadotTMMX1 && WalphadotMXBL1 <= WalphadotTMMX2 &&
WalphadotMXBL1 <= WalphadotMX && WalphadotMXBL1 <= WalphadotMXBL2) // Same as above loop
but MX defines alpha, not TM
            {

                Calpha = atan2(UVMXBL1[1],UVMXBL1[0]);
                Oalpha = atan2(OUVMXBL1[1],OUVMXBL1[0]);
```

C++ Program (Continued)

              alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in alpha in this phase transition. remember, phases starts

                        // out equal to 2 and count equal to 0 on the first loop.
        }

        if (WalphadotMXBL2 <= WalphadotBL && WalphadotMXBL2 <= WalphadotTM && WalphadotMXBL2 <= WalphadotBLTM1 && WalphadotMXBL2 <= WalphadotBLTM2 && WalphadotMXBL2 <= WalphadotTMMX1 && WalphadotMXBL2 <= WalphadotTMMX2 && WalphadotMXBL2 <= WalphadotMXBL1 && WalphadotMXBL2 <= WalphadotMX) // Same as above loop but MX defines alpha, not TM
        {

              Calpha = atan2(UVMXBL2[1],UVMXBL2[0]);
              Oalpha = atan2(OUVMXBL2[1],OUVMXBL2[0]);

              alpha[phase - 1][countb] = (Calpha - Oalpha) * 180 / 3.14; // finds the change in alpha in this phase transition. remember, phases starts

                        // out equal to 2 and count equal to 0 on the first loop.
        }

        if (fabs(alpha[phase - 1][countb]) > 90 && fabs(alpha[phase - 1][countb]) <= 270)
        {
              alpha[phase - 1][countb] = fabs(alpha[phase - 1][countb]) - 180;
        } // This loop ensures that a change in angle of 90 to 270 degrees is not allowed but is instead understood as a reverse in direction (180 rotation) of the reference vector

        if (fabs(alpha[phase - 1][countb]) > 270)
        {
              alpha[phase - 1][countb] = (fabs(alpha[phase - 1][countb]) * fabs(alpha[phase - 1][countb]) / alpha[phase - 1][countb]) - (360 * fabs(alpha[phase - 1][countb]) / alpha[phase - 1][countb]);
        } // This loop ensures that a change in angle of 270+ degrees is not allowed but is instead takes as rotating the opposite direction

// --- BETA Angle Change --

        betadotBL = fabs(UVBL[0] * centralUV[0] + UVBL[1] * centralUV[1]); // dot product of BL and central unit vectors
        betadotTM = fabs(UVTM[0] * centralUV[0] + UVTM[1] * centralUV[1]); // of TM and
        betadotMX = fabs(UVMX[0] * centralUV[0] + UVMX[1] * centralUV[1]); // of MX, etc
        betadotBLTM1 = fabs(UVBLTM1[0] * centralUV[0] + UVBLTM1[1] * centralUV[1]);
        betadotBLTM2 = fabs(UVBLTM2[0] * centralUV[0] + UVBLTM2[1] * centralUV[1]);
        betadotTMMX1 = fabs(UVTMMX1[0] * centralUV[0] + UVTMMX1[1] * centralUV[1]);

C++ Program (Continued)

```cpp
                betadotTMMX2 = fabs(UVTMMX2[0] * centralUV[0] + UVTMMX2[1] * centralUV[1]);
                betadotMXBL1 = fabs(UVMXBL1[0] * centralUV[0] + UVMXBL1[1] * centralUV[1]);
                betadotMXBL2 = fabs(UVMXBL2[0] * centralUV[0] + UVMXBL2[1] * centralUV[1]);

                betadotOBL = fabs(OUVBL[0] * centralUV[0] + OUVBL[1] * centralUV[1]); // dot
product of OBL and central unit vectors
                betadotOTM = fabs(OUVTM[0] * centralUV[0] + OUVTM[1] * centralUV[1]); // of OTM
and
                betadotOMX = fabs(OUVMX[0] * centralUV[0] + OUVMX[1] * centralUV[1]); // of
OMX, etc
                betadotOBLTM1 = fabs(OUVBLTM1[0] * centralUV[0] + OUVBLTM1[1] *
centralUV[1]);
                betadotOBLTM2 = fabs(OUVBLTM2[0] * centralUV[0] + OUVBLTM2[1] *
centralUV[1]);
                betadotOTMMX1 = fabs(OUVTMMX1[0] * centralUV[0] + OUVTMMX1[1] *
centralUV[1]);
                betadotOTMMX2 = fabs(OUVTMMX2[0] * centralUV[0] + OUVTMMX2[1] *
centralUV[1]);
                betadotOMXBL1 = fabs(OUVMXBL1[0] * centralUV[0] + OUVMXBL1[1] *
centralUV[1]);
                betadotOMXBL2 = fabs(OUVMXBL2[0] * centralUV[0] + OUVMXBL2[1] *
centralUV[1]);


                if (betadotBL > betadotOBL) // lets figure out if BL defines angle beta worse than OBL
                {
                        WbetadotBL = betadotBL;
                }
                else
                {
                        WbetadotBL = betadotOBL; // or if visa versa
                }


                if (betadotTM > betadotOTM) // same as above loop but for TM and OTM
                {
                        WbetadotTM = betadotTM;
                }
                else
                {
                        WbetadotTM = betadotOTM;
                }

                if (betadotMX > betadotOMX) // same as above loop but for MX and OMX
                {
                        WbetadotMX = betadotMX;
                }
                else
                {
                        WbetadotMX = betadotOMX;
```

C++ Program (Continued)

```
}

if (betadotBLTM1 > betadotOBLTM1) // same as above loop
{
        WbetadotBLTM1 = betadotBLTM1;
}
else
{
        WbetadotBLTM1 = betadotOBLTM1;
}


if (betadotBLTM2 > betadotOBLTM2) // same as above loop
{
        WbetadotBLTM2 = betadotBLTM2;
}
else
{
        WbetadotBLTM2 = betadotOBLTM2;
}


if (betadotTMMX1 > betadotOTMMX1) // same as above loop
{
        WbetadotTMMX1 = betadotTMMX1;
}
else
{
        WbetadotTMMX1 = betadotOTMMX1;
}


if (betadotTMMX2 > betadotOTMMX2) // same as above loop
{
        WbetadotTMMX2 = betadotTMMX2;
}
else
{
        WbetadotTMMX2 = betadotOTMMX2;
}


if (betadotMXBL1 > betadotOMXBL1) // same as above loop
{
        WbetadotMXBL1 = betadotMXBL1;
}
else
{
        WbetadotMXBL1 = betadotOMXBL1;
}
```

C++ Program (Continued)

```
                if (betadotMXBL2 > betadotOMXBL2) // same as above loop
                {
                        WbetadotMXBL2 = betadotMXBL2;
                }
                else
                {
                        WbetadotMXBL2 = betadotOMXBL2;
                }

                if ( WbetadotBL <= WbetadotTM && WbetadotBL <= WbetadotMX && WbetadotBL <=
        WbetadotBLTM1 && WbetadotBL <= WbetadotBLTM2 && WbetadotBL <= WbetadotTMMX1 &&
        WbetadotBL <= WbetadotTMMX2 && WbetadotBL <= WbetadotMXBL1 && WbetadotBL <=
        WbetadotMXBL2) // means BL gets to define beta angles for this phase transition
                {
                        XRXangle = atan2(XcentralUV[1], XcentralUV[0]); // how many degrees around
        a vertical axis is the cross radial vector from X axis
                        CXangle = atan2(UVBL[1], UVBL[0]); // how many degrees around vertical axis
        is BL vector from X axis
                        OXangle = atan2(OUVBL[1], OUVBL[0]); // how many degrees around vertical
        axis is OBL vector from X axis

                        Cadjacent = cos(CXangle - XRXangle) * sqrt(UVBL[0] * UVBL[0] + UVBL[1] *
        UVBL[1]); // find adjacent piece of current angle
                        Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVBL[0] * OUVBL[0] +
        OUVBL[1] * OUVBL[1]); // find adjacent piece of old angle

                        Copp = UVBL[2];
                        Oopp = OUVBL[2];

                        Cbeta = atan2(Copp,Cadjacent);
                        Obeta = atan2(Oopp,Oadjacent);

                        beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14; // finds the change in beta
        in this phase transition. remember, phases starts

                                        // out equal to 2 and count equal to 0 on the first loop.

                }

                if ( WbetadotTM <= WbetadotMX && WbetadotTM < WbetadotBL && WbetadotTM <=
        WbetadotBLTM1 && WbetadotTM <= WbetadotBLTM2 && WbetadotTM <= WbetadotTMMX1 &&
        WbetadotTM <= WbetadotTMMX2 && WbetadotTM <= WbetadotMXBL1 && WbetadotTM <=
        WbetadotMXBL2) // Same as above loop but TM defines theta, not BL
                {
                        XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                        CXangle = atan2(UVTM[1], UVTM[0]);
                        OXangle = atan2(OUVTM[1], OUVTM[0]);

                        Cadjacent = cos(CXangle - XRXangle) * sqrt(UVTM[0] * UVTM[0] + UVTM[1]
        * UVTM[1]);
```

C++ Program (Continued)

```cpp
                    Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVTM[0] * OUVTM[0] +
OUVTM[1] * OUVTM[1]);

                    Copp = UVTM[2];
                    Oopp = OUVTM[2];

                    Cbeta = atan2(Copp,Cadjacent);
                    Obeta = atan2(Oopp,Oadjacent);

                    beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

            }

            if ( WbetadotMX < WbetadotBL && WbetadotMX < WbetadotTM && WbetadotMX <=
WbetadotBLTM1 && WbetadotMX <= WbetadotBLTM2 && WbetadotMX <= WbetadotTMMX1 &&
WbetadotMX <= WbetadotTMMX2 && WbetadotMX <= WbetadotMXBL1 && WbetadotMX <=
WbetadotMXBL2) // Same as above loop but MX defines theta, not TM
            {
                    XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                    CXangle = atan2(UVMX[1], UVMX[0]);
                    OXangle = atan2(OUVMX[1], OUVMX[0]);

                    Cadjacent = cos(CXangle - XRXangle) * sqrt(UVMX[0] * UVMX[0] + UVMX[1]
* UVMX[1]);
                    Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVMX[0] * OUVMX[0] +
OUVMX[1] * OUVMX[1]);

                    Copp = UVMX[2];
                    Oopp = OUVMX[2];

                    Cbeta = atan2(Copp,Cadjacent);
                    Obeta = atan2(Oopp,Oadjacent);

                    beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

            }

            if ( WbetadotBLTM1 < WbetadotBL && WbetadotBLTM1 < WbetadotTM &&
WbetadotBLTM1 <= WbetadotMX && WbetadotBLTM1 <= WbetadotBLTM2 && WbetadotBLTM1 <=
WbetadotTMMX1 && WbetadotBLTM1 <= WbetadotTMMX2 && WbetadotBLTM1 <= WbetadotMXBL1
&& WbetadotBLTM1 <= WbetadotMXBL2) // Same as above loop but MX defines theta, not TM
            {
                    XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                    CXangle = atan2(UVBLTM1[1], UVBLTM1[0]);
                    OXangle = atan2(OUVBLTM1[1], OUVBLTM1[0]);

                    Cadjacent = cos(CXangle - XRXangle) * sqrt(UVBLTM1[0] * UVBLTM1[0] +
UVBLTM1[1] * UVBLTM1[1]);
                    Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVBLTM1[0] * OUVBLTM1[0]
+ OUVBLTM1[1] * OUVBLTM1[1]);
```

C++ Program (Continued)

```cpp
                        Copp = UVBLTM1[2];
                        Oopp = OUVBLTM1[2];

                        Cbeta = atan2(Copp,Cadjacent);
                        Obeta = atan2(Oopp,Oadjacent);

                        beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                }

                if ( WbetadotBLTM2 < WbetadotBL && WbetadotBLTM2 < WbetadotTM &&
WbetadotBLTM2 <= WbetadotBLTM1 && WbetadotBLTM2 <= WbetadotMX && WbetadotBLTM2 <=
WbetadotTMMX1 && WbetadotBLTM2 <= WbetadotTMMX2 && WbetadotBLTM2 <= WbetadotMXBL1
&& WbetadotBLTM2 <= WbetadotMXBL2) // Same as above loop but MX defines theta, not TM
                {
                        XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                        CXangle = atan2(UVBLTM2[1], UVBLTM2[0]);
                        OXangle = atan2(OUVBLTM2[1], OUVBLTM2[0]);

                        Cadjacent = cos(CXangle - XRXangle) * sqrt(UVBLTM2[0] * UVBLTM2[0] +
UVBLTM2[1] * UVBLTM2[1]);
                        Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVBLTM2[0] * OUVBLTM2[0]
+ OUVBLTM2[1] * OUVBLTM2[1]);

                        Copp = UVBLTM2[2];
                        Oopp = OUVBLTM2[2];

                        Cbeta = atan2(Copp,Cadjacent);
                        Obeta = atan2(Oopp,Oadjacent);

                        beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                }

                if ( WbetadotTMMX1 < WbetadotBL && WbetadotTMMX1 < WbetadotTM &&
WbetadotTMMX1 <= WbetadotBLTM1 && WbetadotTMMX1 <= WbetadotBLTM2 &&
WbetadotTMMX1 <= WbetadotMX && WbetadotTMMX1 <= WbetadotTMMX2 && WbetadotTMMX1
<= WbetadotMXBL1 && WbetadotTMMX1 <= WbetadotMXBL2) // Same as above loop but MX defines
theta, not TM
                {
                        XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                        CXangle = atan2(UVTMMX1[1], UVTMMX1[0]);
                        OXangle = atan2(OUVTMMX1[1], OUVTMMX1[0]);

                        Cadjacent = cos(CXangle - XRXangle) * sqrt(UVTMMX1[0] * UVTMMX1[0] +
UVTMMX1[1] * UVTMMX1[1]);
                        Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVTMMX1[0] * OUVTMMX1[0]
+ OUVTMMX1[1] * OUVTMMX1[1]);

                        Copp = UVTMMX1[2];
```

C++ Program (Continued)

```cpp
                          Oopp = OUVTMMX1[2];

                          Cbeta = atan2(Copp,Cadjacent);
                          Obeta = atan2(Oopp,Oadjacent);

                          beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                  }

                  if ( WbetadotTMMX2 < WbetadotBL && WbetadotTMMX2 < WbetadotTM &&
         WbetadotTMMX2 <= WbetadotBLTM1 && WbetadotTMMX2 <= WbetadotBLTM2 &&
         WbetadotTMMX2 <= WbetadotTMMX1 && WbetadotTMMX2 <= WbetadotMX && WbetadotTMMX2
         <= WbetadotMXBL1 && WbetadotTMMX2 <= WbetadotMXBL2) // Same as above loop but MX defines
         theta, not TM
                  {
                          XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                          CXangle = atan2(UVTMMX2[1], UVTMMX2[0]);
                          OXangle = atan2(OUVTMMX2[1], OUVTMMX2[0]);

                          Cadjacent = cos(CXangle - XRXangle) * sqrt(UVTMMX2[0] * UVTMMX2[0] +
         UVTMMX2[1] * UVTMMX2[1]);
                          Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVTMMX2[0] * OUVTMMX2[0]
         + OUVTMMX2[1] * OUVTMMX2[1]);

                          Copp = UVTMMX2[2];
                          Oopp = OUVTMMX2[2];

                          Cbeta = atan2(Copp,Cadjacent);
                          Obeta = atan2(Oopp,Oadjacent);

                          beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                  }

                  if ( WbetadotMXBL1 < WbetadotBL && WbetadotMXBL1 < WbetadotTM &&
         WbetadotMXBL1 <= WbetadotBLTM1 && WbetadotMXBL1 <= WbetadotBLTM2 && WbetadotMXBL1
         <= WbetadotTMMX1 && WbetadotMXBL1 <= WbetadotTMMX2 && WbetadotMXBL1 <= WbetadotMX
         && WbetadotMXBL1 <= WbetadotMXBL2) // Same as above loop but MX defines theta, not TM
                  {
                          XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                          CXangle = atan2(UVMXBL1[1], UVMXBL1[0]);
                          OXangle = atan2(OUVMXBL1[1], OUVMXBL1[0]);

                          Cadjacent = cos(CXangle - XRXangle) * sqrt(UVMXBL1[0] * UVMXBL1[0] +
         UVMXBL1[1] * UVMXBL1[1]);
                          Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVMXBL1[0] * OUVMXBL1[0]
         + OUVMXBL1[1] * OUVMXBL1[1]);

                          Copp = UVMXBL1[2];
                          Oopp = OUVMXBL1[2];
```

C++ Program (Continued)

```
                            Cbeta = atan2(Copp,Cadjacent);
                            Obeta = atan2(Oopp,Oadjacent);

                            beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                    }

            if ( WbetadotMXBL2 < WbetadotBL && WbetadotMXBL2 < WbetadotTM &&
    WbetadotMXBL2 <= WbetadotBLTM1 && WbetadotMXBL2 <= WbetadotBLTM2 && WbetadotMXBL2
    <= WbetadotTMMX1 && WbetadotMXBL2 <= WbetadotTMMX2 && WbetadotMXBL2 <=
    WbetadotMXBL1 && WbetadotMXBL2 <= WbetadotMX) // Same as above loop but MX defines theta, not
    TM
                    {
                            XRXangle = atan2(XcentralUV[1], XcentralUV[0]);
                            CXangle = atan2(UVMXBL2[1], UVMXBL2[0]);
                            OXangle = atan2(OUVMXBL2[1], OUVMXBL2[0]);

                            Cadjacent = cos(CXangle - XRXangle) * sqrt(UVMXBL2[0] * UVMXBL2[0] +
    UVMXBL2[1] * UVMXBL2[1]);
                            Oadjacent = cos(OXangle - XRXangle) * sqrt(OUVMXBL2[0] * OUVMXBL2[0]
    + OUVMXBL2[1] * OUVMXBL2[1]);

                            Copp = UVMXBL2[2];
                            Oopp = OUVMXBL2[2];

                            Cbeta = atan2(Copp,Cadjacent);
                            Obeta = atan2(Oopp,Oadjacent);

                            beta[phase - 1][countb] = (Cbeta - Obeta) * 180 / 3.14;

                    }


            if (fabs(beta[phase - 1][countb]) > 90 && fabs(beta[phase - 1][countb]) <= 270)
            {
                    beta[phase - 1][countb] = fabs(beta[phase - 1][countb]) - 180;
            }  // This loop ensures that a change in angle of 90 to 270 degrees is not allowed but is
    instead understood as a reverse in direction (180 rotation) of the reference vector


            if (fabs(beta[phase - 1][countb]) > 270)
            {
                    beta[phase - 1][countb] = (fabs(beta[phase - 1][countb]) * fabs(beta[phase -
    1][countb]) / beta[phase - 1][countb]) - (360 * fabs(beta[phase - 1][countb]) / beta[phase - 1][countb]);
            }  // This loop ensures that a change in angle of 270+ degrees is not allowed but is instead
    takes as rotating the opposite direction


    //--------------------------------- Total Rotations --------------------------------------------
```

C++ Program (Continued)

```
                TR1[phase - 1][countb] = acos(UVBL[0] * OUVBL[0] + UVBL[1] * OUVBL[1] +
UVBL[2] * OUVBL[2] - 0.00001) * 180 / 3.14;
                TR2[phase - 1][countb] = acos(UVTM[0] * OUVTM[0] + UVTM[1] * OUVTM[1] +
UVTM[2] * OUVTM[2] - 0.00001) * 180 / 3.14;
                TR3[phase - 1][countb] = acos(UVMX[0] * OUVMX[0] + UVMX[1] * OUVMX[1] +
UVMX[2] * OUVMX[2] - 0.00001) * 180 / 3.14;
                TR4[phase - 1][countb] = acos(UVBLTM1[0] * OUVBLTM1[0] + UVBLTM1[1] *
OUVBLTM1[1] + UVBLTM1[2] * OUVBLTM1[2] - 0.00001) * 180 / 3.14;
                TR5[phase - 1][countb] = acos(UVBLTM2[0] * OUVBLTM2[0] + UVBLTM2[1] *
OUVBLTM2[1] + UVBLTM2[2] * OUVBLTM2[2] - 0.00001) * 180 / 3.14;
                TR6[phase - 1][countb] = acos(UVTMMX1[0] * OUVTMMX1[0] + UVTMMX1[1] *
OUVTMMX1[1] + UVTMMX1[2] * OUVTMMX1[2] - 0.00001) * 180 / 3.14;
                TR7[phase - 1][countb] = acos(UVTMMX2[0] * OUVTMMX2[0] + UVTMMX2[1] *
OUVTMMX2[1] + UVTMMX2[2] * OUVTMMX2[2] - 0.00001) * 180 / 3.14;
                TR8[phase - 1][countb] = acos(UVMXBL1[0] * OUVMXBL1[0] + UVMXBL1[1] *
OUVMXBL1[1] + UVMXBL1[2] * OUVMXBL1[2] - 0.00001) * 180 / 3.14;
                TR9[phase - 1][countb] = acos(UVMXBL2[0] * OUVMXBL2[0] + UVMXBL2[1] *
OUVMXBL2[1] + UVMXBL2[2] * OUVMXBL2[2] - 0.00001) * 180 / 3.14;

                cout << "TR1 " << (TR1[phase - 1][countb]) << "     \n";
                cout << "TR2 " << (TR2[phase - 1][countb]) << "     \n";
                cout << "TR3 " << (TR3[phase - 1][countb]) << "     \n";
                cout << "TR4 " << (TR4[phase - 1][countb]) << "     \n";
                cout << "TR5 " << (TR5[phase - 1][countb]) << "     \n";
                cout << "TR6 " << (TR6[phase - 1][countb]) << "     \n";
                cout << "TR7 " << (TR7[phase - 1][countb]) << "     \n";
                cout << "TR8 " << (TR8[phase - 1][countb]) << "     \n";
                cout << "TR9 " << (TR9[phase - 1][countb]) << "     \n";

// --------------------------------- Input Reset -----------------------------------------------

                cout << "Are these last inputs correct? y or n.";
                cin >> retry;
//              cout << "countb equals " << countb << endl;
                countb = countb + 1;
//              cout << "countb now equals " << countb << endl;
                if (retry == 'n')  // loop that lets you redo if you entered somthing wrong
                {
                        countb = countb - 1;
                }
         }
        phase = phase + 1;
        countb = 0;
        }

// ---------------------------------- Output -----------------------------------------------
        phase = 0;
        countb = 0;

        Outputfile.open("c:\\triaxdata.txt", ios::out);
```

C++ Program (Continued)

```
        Outputfile.precision(5);

        Outputfile << "     Particle #    Phase #   R dist(vox)   Z pos(vox)   R move(vox)  V move(vox)
XR move(vox)  R dist(in)   Z pos(in)    R move(in)   V move(in)   XR move(in)   Theta Change   Alpha
Change    Beta Change \n";

        while (phase <= 0)
        {
                while (countb < count)
                {
                        Outputfile << setw(14) << (countb + 1) << setw(14) << (phase + 1) << setw(14)
<< rdis[phase][countb] << setw(14) << Zpos[phase][countb] << setw(14) << Rmove[phase][countb] <<
setw(14) << Vmove[phase][countb] << setw(14) << XRmove[phase][countb];
                        rdis[phase][countb] = rdis[phase][countb] / resolution;
                        Zpos[phase][countb] = Zpos[phase][countb] / resolution;
                        Rmove[phase][countb] = Rmove[phase][countb] / resolution;
                        Vmove[phase][countb] = Vmove[phase][countb] / resolution;
                        XRmove[phase][countb] = XRmove[phase][countb] / resolution;
                        Outputfile << setw(14) << rdis[phase][countb] << setw(14) <<
Zpos[phase][countb] << setw(14) << Rmove[phase][countb] << setw(14) << Vmove[phase][countb] <<
setw(14) << XRmove[phase][countb];
                        Outputfile << setw(17) << theta[phase][countb] << setw(17) <<
alpha[phase][countb] << setw(17) << beta[phase][countb] << endl;
                        countb = countb + 1;
                }
        countb = 0;
        phase = phase + 1;
        }

        phase = 0;

        countb = 0;

        Outputfile << " \n";
        Outputfile << "     Particle #    Phase #     TR1                TR2            TR3            TR4
TR5       TR6        TR7        TR8        TR9 \n";

        while (phase <= 0)
        {
                while (countb < count)
                {
                        Outputfile << setw(14) << (countb + 1) << setw(14) << (phase + 1) << setw(14)
<< TR1[phase][countb] << setw(14) << TR2[phase][countb] << setw(14) << TR3[phase][countb] << setw(14)
<< TR4[phase][countb];
                        Outputfile << setw(14) << TR5[phase][countb] << setw(14) <<
TR6[phase][countb] << setw(14) << TR7[phase][countb] << setw(14) << TR8[phase][countb] << setw(14)
<< TR9[phase][countb] <<endl;
                        countb = countb + 1;
                }
        countb = 0;
        phase = phase + 1;
```

C++ Program (Continued)

```
        }

        phase = 0;
        countb = 0;

        Outputfile << " \n";
        Outputfile << "       INPUT      INPUT      INPUT      INPUT       EFFECTIVE      INPUT
INPUT      EFFECTIVE      INPUT       INPUT       EFFECTIVE \n";
        Outputfile << "     Particle #  Strain inc.    Bx          By          Bz          Lx          Ly          Lz
Tx          Ty          Tz   \n";

        while (phase <= 1)
        {
                while (countb < count)
                {
                        Outputfile << setw(14) << (countb + 1) << setw(14) << (phase + 1) << setw(14)
<< XBcoord[phase][countb] << setw(14) << YBcoord[phase][countb] << setw(14) <<
ZBcoord[phase][countb] << setw(14) << XLcoord[phase][countb] << setw(14) << YLcoord[phase][countb];
                        Outputfile << setw(14) << ZLcoord[phase][countb] << setw(14) <<
XTcoord[phase][countb] << setw(14) << YTcoord[phase][countb] << setw(14) << ZTcoord[phase][countb]
<< endl;
                        countb = countb + 1;
                }
        countb = 0;
        phase = phase + 1;
        }



        phase = 0;

        Outputfile << " \n";
        Outputfile << "       INPUT      INPUT      INPUT      INPUT       INPUT       INPUT
INPUT \n";
        Outputfile << "     Strain inc.    Xtop         Ytop          Ztop         Xbottom       Ybottom
Zbottom \n";

        while (phase <= 1)
        {
                Outputfile << setw(14) << (phase + 1) << setw(14) << Xcentertop[phase] << setw(14) <<
Ycentertop[phase] << setw(14) << Zcentertop[phase] << setw(14) << Xcenterbottom[phase] << setw(14) <<
Ycenterbottom[phase] << setw(14) << Zcenterbottom[phase] << endl;

                phase = phase + 1;
        }
```

C++ Program (Continued)

```
        Outputfile << " \n";
        Outputfile << "      INPUT       INPUT   \n";
        Outputfile << "    Hor. Res.    V. Ratio  \n";
        Outputfile << setw(14) << resolution << setw(14) << ratio << endl;

}
```

APPENDIX C: ILLUSTRATED GRANULAR MOVEMENTS OF INTEREST

Overview

The following figures display X-ray CT cross sections which, when viewed alone or in series from one strain increment to another, portray qualities thought to be of interest to a reader of this thesis. They also provide a feeling of what cross sectional images look like when viewing volumetric CT-scan files of granular material with solder particle markings.

Reference Solder Particles

When collecting the postion of solder particles, the three solder pieces attached to each grain are distinguishable from on another in terms of size and shape. The three solder pieces are refered to as B (for big), L (for long) and T (for tiny). In Figure C.1 a particle is aligned so-that all three pieces are viewable and identifiable in a single cross section.

Figure C.1. Figure illustrating solder particle pieces B, L and T.

Large Particle Movement, Sample Spreading

Figures C.2 through C.6 follow the movement of sand grain number 28 within sample 3. The sand grain is located in the bottom left corner of the cross sections. For reference, each cross sectional image is taken at the mid-point of particle L (the furthest left of the two solder pieces). The particle L is inclined relatively vertically so it appears more as a dot than a rod. The particle to the right of L is marker B which begins below the center of marker L and results above the center of L (as can be seen by B becoming more and then less distant). The large rotation along the alpha plane (the plane which lies along the cross section) is very visable between 15% and 20% strain. The rotations of sand grain 28 along the 3 orthogonal reference planes between the five strain increments are provided in Table C.1.

Table C.1. Rotations, in degrees, of sand grain number 28 during incremental strain.

Strain Increment	Theta	alpha	beta
0% to 5% strain	13°	10°	-4°
5% to 10% strain	6°	6°	4°
10% to 15% strain	21°	-1°	-3°
15% to 20% strain	5°	-28°	-7°

Possibly more interesting than watching the large rotations of an individual particle is the development of movements of particles in groups. It can be seen that as the sample strains, large voids begin to open as the sample appears to split and spread.

Figure C.2. Initial position.

Figure C.3. Position at 5% strain.

Figure C.4. Position at 10% strain.

Figure C.5. Position at 15% strain.

Figure C.6. Position at 20% strain.

Large Particle Movement, Sample Ends

Common place of large particle movement is at the sample ends near the radial extent of the sample. This is exemplified by sand grain number 2 of sample 2 as shown in Figures C.7 through C.10). This sand grain, located in the lower left of the sample cross section, rotates significantly as it moves laterally outside of the general sample area and around the porous stone which caps the sample. Sand grain number 2 rotates a cumulative 87 degrees during 20% sample strain.

The center of solder marker T is used as a reference, keeping it level with each cross section. Nearly all of the rotation occurs in reference angle theta so T and B, located close to the axis of theta, don't move much relative to each other. It can be seen, however, that the reference solder marker T does move vertically enough from 10% to 20% strain that the entire cross section looks different.

Figure C.7. Initial position.

Figure C.8. Position at 5% strain.

178

Figure C.9. Position at 10% strain.

Figure C.10. Position at 20% strain.

X-ray CT Produced Sample Profiles

The following Figures. C.11 through C.19 illustrate the initial and final strain increment profiles of all samples analyzed in this research, both from a perspective of the sample as a whole and with the entirety of the sample removed with the exception of the solder markers.

Figure C.11. Initial sample 1 profile.   Figure C.12. Sample 1 at 20% strain.

Figure C.13. Initial sample 1 profile.

Figure C.14. Sample 1 at 20% strain.

Figure C.15. Initial sample 2 profile.

Figure C.16. Sample 2 at 20% strain.

182

Figure C.17. Initial sample 2 profile.

Figure C.18. Sample 2 at 20% strain.

Figure C.19. Initial sample 3 profile.

Figure C.20. Sample 3 at 8% strain.

Figure C.21. Initial sample 3 profile.　　Figure C.22. Sample 3 at 8% strain.

APPENDIX D: THREE DIMENSIONAL COORDINATES OF REFERENCE DATA USED IN GRANULAR MOVEMENT CHARACTERIZATION

The following Tables D.1 through D.6 list the input coordinates of each solder particle types B, L and T and the input coordinates of sample ends for each sample tested. These are the required inputs for the C++ program, provided in Appendix B, to compute particle movement. All coordinates are in units of voxels. The voxels are 0.12 mm (0.0047 in) cubes.

Table D.1. Solder particle input coordinates of sample 1.

Particle	Strain%	Bx	By	Bz	Lx	Ly	Lz	Tx	Ty	Tz
1	0	483	356	8	473	340	33	457	314	18
2	0	539	341	98	518	342	125	499	353	87
3	0	487	331	145	465	361	156	500	395	130
4	0	235	360	144	247	342	170	225	306	154
5	0	485	529	179	464	556	168	439	540	189
6	0	186	269	179	197	264	213	235	286	213
7	0	304	408	200	269	405	221	255	370	209
8	0	545	273	214	553	295	235	537	281	260
9	0	458	291	263	441	328	232	418	333	266
10	0	344	180	260	384	196	273	372	151	259
11	0	534	357	288	508	323	277	533	355	255
12	0	258	330	263	258	329	227	205	309	235
13	0	228	522	293	239	515	268	271	555	286
14	0	333	513	340	362	494	321	383	531	337
15	0	332	200	379	317	230	349	277	219	380
16	0	578	355	365	539	344	349	566	320	372
17	0	314	449	383	358	440	398	344	475	363
18	0	312	286	395	291	291	407	276	285	374

Table D.1 (Continued)

19	0	530	293	417	515	275	444	511	246	409
20	0	517	413	444	543	399	411	563	365	442
21	0	228	309	500	203	267	506	240	247	502
22	0	285	439	497	298	429	529	267	394	510
23	0	362	181	514	360	162	534	389	184	551
24	0	367	260	570	356	249	529	304	259	567
25	0	382	285	556	416	299	575	380	328	567
26	0	430	231	560	456	227	569	440	186	569
27	0	515	262	612	492	283	594	486	327	623
28	0	302	225	639	279	210	627	281	241	593
29	0	293	381	666	328	380	652	313	343	636
30	0	540	503	657	519	461	640	549	453	616
31	0	374	213	674	387	186	653	387	243	640
32	0	276	241	704	305	258	682	298	278	717
33	0	366	170	741	404	177	734	393	201	710
34	0	319	218	783	345	185	771	324	205	734
35	0	414	206	762	381	219	775	379	211	746
36	0	463	194	723	424	201	724	472	210	760
37	0	336	177	842	323	169	788	295	187	817
38	0	217	349	760	242	346	733	254	377	760
39	0	215	369	816	195	386	844	170	335	819
40	0	203	305	816	224	292	849	176	275	853
1	5	485	357	8	475	342	29	456	314	18
2	5	540	340	97	518	342	124	500	352	87
3	5	486	329	144	466	360	154	501	394	129
4	5	233	361	142	244	343	167	222	307	152
5	5	486	532	173	470	559	164	442	547	185
6	5	182	270	172	195	264	208	231	284	210
7	5	300	408	199	266	408	218	249	374	207

Table D.1 (Continued)

8	5	553	272	208	562	294	226	545	283	254
9	5	457	288	254	444	326	227	421	331	264
10	5	348	172	249	390	185	264	375	142	251
11	5	540	358	282	517	322	269	539	356	249
12	5	252	330	258	251	329	223	200	310	227
13	5	220	537	278	232	526	252	264	566	274
14	5	332	523	329	363	504	311	384	541	325
15	5	333	188	365	318	224	340	277	207	367
16	5	595	352	352	555	344	338	582	319	361
17	5	308	450	367	353	448	374	336	483	347
18	5	304	282	375	284	288	389	265	284	358
19	5	541	292	400	526	270	426	521	240	393
20	5	531	414	428	556	403	394	577	369	425
21	5	212	304	472	190	263	481	230	244	483
22	5	266	443	473	285	434	504	252	398	491
23	5	360	167	490	361	149	510	388	174	524
24	5	368	254	543	359	242	501	305	254	539
25	5	381	283	525	417	293	548	382	326	537
26	5	434	223	534	458	223	544	443	179	542
27	5	526	262	590	503	278	570	495	324	595
28	5	296	214	609	275	194	602	272	221	562
29	5	284	377	632	316	382	618	302	342	601
30	5	545	515	629	526	471	615	558	464	590
31	5	374	203	643	388	179	620	386	236	611
32	5	268	234	668	296	250	645	292	273	681
33	5	362	165	705	399	167	705	394	191	677
34	5	314	212	745	339	179	732	318	198	697
35	5	412	201	728	379	214	738	375	204	710
36	5	463	186	690	425	192	694	473	205	725

Table D.1 (Continued)

37	5	330	172	802	314	161	750	287	182	779
38	5	210	349	723	234	345	695	247	376	720
39	5	209	368	778	186	384	806	164	332	782
40	5	195	300	778	219	288	810	172	273	816
1	10	483	356	8	477	343	32	457	314	18
2	10	541	340	98	519	339	124	498	352	88
3	10	487	330	143	466	361	154	502	394	129
4	10	232	362	140	242	344	167	233	306	148
5	10	486	531	172	469	561	164	442	549	182
6	10	180	268	170	193	261	205	228	282	209
7	10	298	408	199	263	410	218	245	377	207
8	10	559	272	203	567	294	221	553	280	247
9	10	457	286	251	446	325	224	423	331	261
10	10	350	167	245	394	177	259	375	134	246
11	10	543	359	278	524	321	266	540	358	245
12	10	245	331	256	248	329	222	197	307	223
13	10	206	552	254	222	538	235	254	575	258
14	10	330	534	319	360	512	306	382	552	314
15	10	330	178	354	320	217	333	274	198	357
16	10	604	352	347	564	343	333	593	316	349
17	10	303	464	344	348	452	358	334	492	338
18	10	284	295	358	265	293	375	245	289	344
19	10	549	282	388	535	261	414	529	234	381
20	10	540	417	422	564	405	386	585	372	417
21	10	184	304	442	165	258	451	203	244	455
22	10	234	450	441	260	441	467	227	405	463
23	10	349	153	465	349	135	487	375	161	502
24	10	356	251	511	344	234	473	293	256	505
25	10	370	288	502	405	287	529	383	326	516

189

Table D.1 (Continued)

26	10	431	219	519	455	211	530	434	173	520
27	10	540	259	576	512	275	553	504	319	576
28	10	277	211	574	258	189	569	246	213	529
29	10	264	382	597	296	387	579	281	349	561
30	10	539	537	599	528	490	585	562	489	565
31	10	372	195	612	377	170	585	379	230	581
32	10	251	234	629	281	254	604	275	275	640
33	10	355	159	664	392	158	672	396	177	641
34	10	303	214	701	328	182	689	304	200	654
35	10	402	196	683	370	212	696	367	204	667
36	10	462	173	657	424	183	660	475	196	686
37	10	321	173	757	302	162	706	276	182	736
38	10	192	349	685	218	348	657	229	380	682
39	10	195	370	741	173	385	771	151	334	746
40	10	181	302	741	206	285	770	161	273	780
1	15	484	358	10	473	343	36	456	315	20
2	15	539	341	102	518	339	129	498	353	91
3	15	484	330	147	463	361	157	494	394	131
4	15	230	360	142	239	345	167	220	306	148
5	15	484	532	176	466	563	168	440	548	186
6	15	176	267	170	188	260	204	224	281	209
7	15	292	406	198	259	408	219	240	376	205
8	15	558	272	207	567	295	224	553	284	251
9	15	453	285	252	438	330	228	419	329	264
10	15	350	164	245	391	175	259	373	135	247
11	15	540	360	282	523	321	269	535	362	250
12	15	240	328	255	244	327	222	193	305	221
13	15	186	561	237	203	545	221	233	585	242
14	15	321	541	312	352	520	303	373	558	309

Table D.1 (Continued)

15	15	325	173	351	311	214	334	268	191	353
16	15	602	353	349	562	345	335	592	318	351
17	15	283	467	334	325	454	352	313	496	334
18	15	260	300	343	242	294	358	228	285	326
19	15	546	280	389	533	260	416	527	231	383
20	15	539	417	425	562	406	390	582	374	419
21	15	143	303	414	129	257	426	169	247	423
22	15	199	457	409	226	442	428	186	412	428
23	15	338	137	453	338	119	474	369	144	486
24	15	342	247	483	328	224	449	280	255	483
25	15	362	285	485	383	283	523	376	323	499
26	15	423	214	514	447	203	525	427	169	506
27	15	537	262	576	511	276	555	497	318	575
28	15	245	204	539	226	183	544	210	195	501
29	15	232	386	561	262	394	543	249	354	524
30	15	517	558	579	514	507	566	549	507	550
31	15	365	187	587	369	162	561	369	221	553
32	15	223	238	589	250	257	564	247	278	599
33	15	338	153	632	375	143	640	383	166	615
34	15	279	217	658	303	182	645	279	201	612
35	15	390	191	643	361	214	650	360	206	621
36	15	464	151	631	428	166	635	480	180	654
37	15	298	174	711	277	162	662	253	183	692
38	15	166	351	652	188	351	621	200	382	646
39	15	169	369	706	151	384	737	127	332	711
40	15	157	302	704	184	288	733	140	272	743
1	20	484	357	9	475	344	34	455	313	19
2	20	540	341	98	519	340	125	498	353	88
3	20	485	330	144	466	363	154	501	393	129

Table D.1 (Continued)

4	20	233	360	140	243	344	166	222	306	147
5	20	485	531	173	469	562	166	442	549	184
6	20	179	268	168	189	262	203	227	282	207
7	20	293	406	193	262	408	217	241	375	205
8	20	562	272	201	571	295	219	558	283	247
9	20	456	285	250	444	326	226	422	330	261
10	20	352	162	242	393	171	255	376	130	244
11	20	544	361	278	526	322	266	539	361	245
12	20	238	330	254	244	328	221	194	304	215
13	20	182	572	221	202	554	212	229	592	236
14	20	322	543	307	353	520	300	374	560	303
15	20	323	167	343	308	205	325	266	183	342
16	20	607	353	346	567	343	330	597	317	344
17	20	281	470	328	322	452	349	310	501	333
18	20	244	298	332	227	290	348	209	280	319
19	20	551	280	384	539	256	411	533	228	376
20	20	546	422	422	567	407	385	589	377	416
21	20	118	305	384	106	262	397	147	250	393
22	20	177	462	375	202	444	393	158	420	391
23	20	331	127	440	334	106	458	366	125	469
24	20	330	239	463	318	218	428	270	247	462
25	20	354	276	465	369	270	506	374	311	483
26	20	422	205	499	441	188	513	428	160	479
27	20	551	258	560	531	274	534	511	317	558
28	20	234	200	506	207	190	516	196	194	469
29	20	214	394	521	243	402	501	230	363	483
30	20	526	575	556	521	525	551	558	525	533
31	20	363	173	562	357	146	539	374	201	524
32	20	206	251	546	231	269	520	231	285	558

Table D.1 (Continued)

33	20	330	133	601	364	121	612	379	143	587
34	20	270	223	618	287	188	598	262	212	569
35	20	382	178	603	353	200	612	355	198	583
36	20	466	132	603	429	145	597	480	161	626
37	20	287	175	668	264	165	620	243	190	652
38	20	152	358	616	173	358	584	187	388	609
39	20	158	375	671	141	389	703	116	337	678
40	20	145	308	668	174	292	696	130	276	708

Table D.2. Central top and bottom input coordinates of sample 1.

Strain (%)	Xtop	Ytop	Ztop	Xbottom	Ybottom	Zbottom
0	360.0	377.0	3.5	368.0	365.0	351.0
5	356.0	376.0	6.0	368.0	364.0	336.5
10	347.0	379.0	8.0	367.0	365.0	321.5
15	326.0	380.0	12.0	367.0	365.0	311.0
20	319.0	386.0	16.0	368.0	365.0	296.5

Table D.3. Solder particle input coordinates of sample 2.

Particle	Strain%	Bx	By	Bz	Lx	Ly	Lz	Tx	Ty	Tz
1	0	372	355	56	387	387	71	415	381	50
2	0	183	344	70	162	352	56	176	388	70
3	0	553	360	79	514	361	70	556	400	84
4	0	432	165	93	402	179	124	405	173	78
5	0	464	396	88	457	359	92	418	371	99
6	0	337	563	102	319	527	96	288	551	110
7	0	285	256	133	285	283	99	322	266	120
8	0	381	327	162	369	329	138	373	283	147
9	0	545	420	196	526	407	213	494	422	203
10	0	460	296	199	473	305	237	511	309	213
11	0	434	395	240	407	393	213	403	422	232
12	0	463	177	228	456	209	245	489	214	219

Table D.3 (Continued)

13	0	305	465	268	254	464	242	289	489	227
14	0	578	304	240	552	321	224	570	365	249
15	0	449	369	275	461	394	241	408	418	270
16	0	214	238	279	244	281	279	274	256	284
17	0	524	463	293	523	466	328	556	486	310
18	0	441	466	320	459	425	321	502	447	328
19	0	399	196	326	439	185	349	398	159	347
20	0	450	252	440	434	264	408	483	273	408
21	0	390	401	496	419	376	477	374	359	491
22	0	386	453	523	379	446	500	380	413	515
23	0	443	538	521	444	522	546	473	508	536
24	0	393	565	553	420	542	551	401	494	564
25	0	268	424	649	264	429	685	317	455	657
26	0	291	526	658	330	530	686	354	528	650
27	0	314	364	668	274	334	666	259	367	686
28	0	552	423	674	527	389	681	551	412	709
29	0	490	483	686	520	507	711	531	458	719
30	0	454	216	710	457	212	682	446	262	686
31	0	237	399	695	272	407	694	278	374	694
32	0	437	440	736	458	437	704	474	407	731
33	0	271	413	728	290	392	756	258	411	794
34	0	497	394	764	491	388	733	524	355	737
35	0	519	350	766	493	374	771	457	354	760
36	0	390	487	849	351	488	858	355	464	835
37	0	428	295	843	416	299	875	456	333	849
38	0	376	165	889	378	203	872	348	175	845
39	0	546	305	848	557	308	890	534	269	878
40	0	325	181	895	336	187	868	291	174	857
1	5	372	356	57	387	388	73	415	382	50

194

Table D.3 (Continued)

2	5	164	341	65	159	359	45	164	385	71
3	5	556	361	79	517	361	73	560	401	83
4	5	432	163	92	400	174	124	409	171	78
5	5	465	399	85	460	363	88	420	373	99
6	5	313	573	94	317	535	97	287	563	102
7	5	283	256	133	284	283	98	321	266	120
8	5	382	328	159	369	327	131	373	284	144
9	5	553	424	190	534	412	208	504	426	199
10	5	466	296	194	479	307	231	518	309	207
11	5	436	402	233	412	409	205	406	427	226
12	5	467	173	219	460	202	242	489	212	218
13	5	308	479	251	255	472	236	282	502	215
14	5	587	302	235	560	319	218	581	364	242
15	5	451	375	266	463	401	233	412	423	261
16	5	204	239	261	232	281	266	265	260	273
17	5	528	472	283	529	471	317	564	491	302
18	5	445	475	310	460	435	309	504	453	319
19	5	395	192	314	439	182	337	395	156	337
20	5	446	258	426	432	265	393	481	276	395
21	5	383	407	477	412	381	459	369	365	472
22	5	380	462	501	376	454	478	373	422	495
23	5	443	552	499	443	535	524	470	520	516
24	5	387	576	529	416	555	527	399	507	538
25	5	257	436	618	252	443	654	304	469	630
26	5	282	544	629	323	546	657	344	545	620
27	5	299	368	640	258	342	636	244	376	655
28	5	558	430	654	534	395	654	556	413	686
29	5	492	503	655	522	524	681	531	474	689
30	5	455	204	682	461	197	655	449	250	657

Table D.3 (Continued)

31	5	223	412	663	258	417	662	262	384	662
32	5	435	450	700	458	448	672	474	419	703
33	5	260	422	695	276	401	723	245	421	761
34	5	496	407	729	498	397	697	526	364	708
35	5	518	357	731	493	380	737	456	361	725
36	5	383	495	811	345	499	820	326	473	798
37	5	424	298	799	410	303	831	451	336	807
38	5	365	168	848	368	201	822	334	169	805
39	5	546	305	810	552	307	852	528	272	838
40	5	315	185	853	325	185	825	278	171	820
1	10	373	356	56	386	388	71	416	382	50
2	10	153	343	61	154	360	41	152	387	68
3	10	557	361	79	517	362	73	560	401	83
4	10	431	163	92	403	175	125	403	172	77
5	10	466	399	85	458	362	88	420	373	99
6	10	333	576	93	314	538	98	284	566	98
7	10	282	256	134	284	283	98	320	266	120
8	10	381	328	158	369	326	131	372	284	144
9	10	555	426	189	548	411	203	505	428	198
10	10	467	296	193	480	306	230	518	308	205
11	10	437	403	231	412	413	203	408	432	223
12	10	465	173	219	458	199	242	489	212	217
13	10	299	490	233	242	482	228	266	515	207
14	10	587	304	235	561	319	221	582	366	240
15	10	452	378	264	469	400	230	411	428	258
16	10	199	238	255	224	281	261	257	259	270
17	10	529	479	281	531	478	314	566	495	297
18	10	443	482	302	459	441	302	501	458	315
19	10	395	193	312	434	184	337	391	158	337

Table D.3 (Continued)

20	10	441	266	422	430	270	388	474	282	389
21	10	371	421	464	402	394	453	356	380	460
22	10	369	484	485	366	474	462	358	445	482
23	10	449	589	478	440	576	505	468	559	507
24	10	376	615	516	404	601	507	395	547	520
25	10	238	462	593	231	471	627	285	494	602
26	10	264	577	604	305	577	631	326	578	594
27	10	272	383	607	227	364	613	222	401	629
28	10	564	466	638	546	430	622	565	437	659
29	10	496	539	629	523	558	659	534	507	660
30	10	450	208	646	455	202	618	444	254	620
31	10	201	440	635	236	442	629	239	410	631
32	10	433	482	664	465	478	644	470	451	674
33	10	242	449	664	260	422	692	229	444	732
34	10	492	433	691	493	429	659	518	391	667
35	10	509	383	690	483	406	694	447	388	680
36	10	371	515	776	334	519	789	313	494	766
37	10	413	319	751	400	322	784	440	357	761
38	10	358	184	796	355	213	766	326	176	755
39	10	537	328	757	543	324	801	519	291	783
40	10	306	199	802	309	191	774	264	176	785
1	20	377	354	54	391	387	69	419	381	47
2	20	144	342	50	153	356	28	134	385	50
3	20	562	359	77	522	359	71	565	399	82
4	20	432	157	88	404	169	121	406	168	74
5	20	468	398	82	461	360	85	424	372	97
6	20	333	580	84	314	547	99	285	571	84
7	20	287	253	128	285	284	94	323	266	116
8	20	385	327	152	373	326	126	374	282	139

Table D.3 (Continued)

9	20	570	432	173	561	420	191	522	437	189
10	20	474	296	185	487	303	223	526	306	199
11	20	447	410	215	417	420	194	418	438	217
12	20	476	162	209	468	189	235	498	204	210
13	20	306	526	196	257	500	205	262	534	176
14	20	606	302	223	579	319	210	605	365	226
15	20	457	383	252	477	403	221	424	436	245
16	20	186	239	221	213	277	241	241	251	252
17	20	546	496	260	547	495	296	583	511	279
18	20	455	501	283	466	458	281	509	470	298
19	20	393	177	293	431	168	319	390	142	319
20	20	431	271	395	430	263	358	473	281	372
21	20	356	446	427	385	417	414	336	406	418
22	20	353	515	445	352	505	422	338	477	441
23	20	444	648	414	436	635	441	465	617	444
24	20	363	668	454	393	653	444	394	604	465
25	20	212	516	529	205	527	564	261	546	539
26	20	247	637	548	288	636	575	309	637	538
27	20	231	413	546	187	407	567	190	450	565
28	20	562	514	562	545	479	543	565	483	581
29	20	493	598	558	522	611	592	536	564	588
30	20	447	214	581	453	210	553	438	262	556
31	20	175	488	572	203	496	556	211	461	558
32	20	426	537	590	460	532	571	461	501	597
33	20	225	503	595	242	477	620	207	488	660
34	20	484	483	607	492	481	574	513	444	578
35	20	497	432	594	473	460	604	438	449	593
36	20	361	555	705	325	558	717	303	535	692
37	20	395	363	652	384	359	686	426	394	666

Table D.3 (Continued)

38	20	341	216	686	326	247	660	308	204	644
39	20	524	367	653	531	355	695	505	327	672
40	20	285	236	694	289	213	674	248	202	696

Table D.4. Central top and bottom input coordinates of sample 2.

Strain (%)	Xtop	Ytop	Ztop	Xbottom	Ybottom	Zbottom
0	361.0	370.0	7.5	369.0	366.0	362.0
5	354.0	376.0	14.0	369.0	366.0	352.5
10	344.0	398.0	15.0	369.0	366.0	337.5
20	346.0	416.0	27.5	371.0	352.0	316.0

Table D.5. Solder particle input coordinates of sample 3.

Particle	Strain%	Bx	By	Bz	Lx	Ly	Lz	Tx	Ty	Tz
1	0	483	356	8	473	340	33	457	314	18
2	0	539	341	98	518	342	125	499	353	87
3	0	487	331	145	465	361	156	500	395	130
4	0	235	360	144	247	342	170	225	306	154
5	0	485	529	179	464	556	168	439	540	189
6	0	186	269	179	197	264	213	235	286	213
7	0	304	408	200	269	405	221	255	370	209
8	0	545	273	214	553	295	235	537	281	260
9	0	458	291	263	441	328	232	418	333	266
10	0	344	180	260	384	196	273	372	151	259
11	0	534	357	288	508	323	277	533	355	255
12	0	258	330	263	258	329	227	205	309	235
13	0	228	522	293	239	515	268	271	555	286
14	0	333	513	340	362	494	321	383	531	337
15	0	332	200	379	317	230	349	277	219	380
16	0	578	355	365	539	344	349	566	320	372
17	0	314	449	383	358	440	398	344	475	363

Table D.5 (Continued)

18	0	312	286	395	291	291	407	276	285	374
19	0	530	293	417	515	275	444	511	246	409
20	0	517	413	444	543	399	411	563	365	442
21	0	228	309	500	203	267	506	240	247	502
22	0	285	439	497	298	429	529	267	394	510
23	0	362	181	514	360	162	534	389	184	551
24	0	367	260	570	356	249	529	304	259	567
25	0	382	285	556	416	299	575	380	328	567
26	0	430	231	560	456	227	569	440	186	569
27	0	515	262	612	492	283	594	486	327	623
28	0	302	225	639	279	210	627	281	241	593
29	0	293	381	666	328	380	652	313	343	636
30	0	540	503	657	519	461	640	549	453	616
31	0	374	213	674	387	186	653	387	243	640
32	0	276	241	704	305	258	682	298	278	717
33	0	366	170	741	404	177	734	393	201	710
34	0	319	218	783	345	185	771	324	205	734
35	0	414	206	762	381	219	775	379	211	746
36	0	463	194	723	424	201	724	472	210	760
37	0	336	177	842	323	169	788	295	187	817
38	0	217	349	760	242	346	733	254	377	760
39	0	215	369	816	195	386	844	170	335	819
40	0	203	305	816	224	292	849	176	275	853
1	2	485	357	8	475	342	29	456	314	18
2	2	540	340	97	518	342	124	500	352	87
3	2	486	329	144	466	360	154	501	394	129
4	2	233	361	142	244	343	167	222	307	152
5	2	486	532	173	470	559	164	442	547	185
6	2	182	270	172	195	264	208	231	284	210

Table D.5 (Continued)

7	2	300	408	199	266	408	218	249	374	207
8	2	553	272	208	562	294	226	545	283	254
9	2	457	288	254	444	326	227	421	331	264
10	2	348	172	249	390	185	264	375	142	251
11	2	540	358	282	517	322	269	539	356	249
12	2	252	330	258	251	329	223	200	310	227
13	2	220	537	278	232	526	252	264	566	274
14	2	332	523	329	363	504	311	384	541	325
15	2	333	188	365	318	224	340	277	207	367
16	2	595	352	352	555	344	338	582	319	361
17	2	308	450	367	353	448	374	336	483	347
18	2	304	282	375	284	288	389	265	284	358
19	2	541	292	400	526	270	426	521	240	393
20	2	531	414	428	556	403	394	577	369	425
21	2	212	304	472	190	263	481	230	244	483
22	2	266	443	473	285	434	504	252	398	491
23	2	360	167	490	361	149	510	388	174	524
24	2	368	254	543	359	242	501	305	254	539
25	2	381	283	525	417	293	548	382	326	537
26	2	434	223	534	458	223	544	443	179	542
27	2	526	262	590	503	278	570	495	324	595
28	2	296	214	609	275	194	602	272	221	562
29	2	284	377	632	316	382	618	302	342	601
30	2	545	515	629	526	471	615	558	464	590
31	2	374	203	643	388	179	620	386	236	611
32	2	268	234	668	296	250	645	292	273	681
33	2	362	165	705	399	167	705	394	191	677
34	2	314	212	745	339	179	732	318	198	697
35	2	412	201	728	379	214	738	375	204	710

Table D.5 (Continued)

36	2	463	186	690	425	192	694	473	205	725
37	2	330	172	802	314	161	750	287	182	779
38	2	210	349	723	234	345	695	247	376	720
39	2	209	368	778	186	384	806	164	332	782
40	2	195	300	778	219	288	810	172	273	816
1	4	483	356	8	477	343	32	457	314	18
2	4	541	340	98	519	339	124	498	352	88
3	4	487	330	143	466	361	154	502	394	129
4	4	232	362	140	242	344	167	233	306	148
5	4	486	531	172	469	561	164	442	549	182
6	4	180	268	170	193	261	205	228	282	209
7	4	298	408	199	263	410	218	245	377	207
8	4	559	272	203	567	294	221	553	280	247
9	4	457	286	251	446	325	224	423	331	261
10	4	350	167	245	394	177	259	375	134	246
11	4	543	359	278	524	321	266	540	358	245
12	4	245	331	256	248	329	222	197	307	223
13	4	206	552	254	222	538	235	254	575	258
14	4	330	534	319	360	512	306	382	552	314
15	4	330	178	354	320	217	333	274	198	357
16	4	604	352	347	564	343	333	593	316	349
17	4	303	464	344	348	452	358	334	492	338
18	4	284	295	358	265	293	375	245	289	344
19	4	549	282	388	535	261	414	529	234	381
20	4	540	417	422	564	405	386	585	372	417
21	4	184	304	442	165	258	451	203	244	455
22	4	234	450	441	260	441	467	227	405	463
23	4	349	153	465	349	135	487	375	161	502
24	4	356	251	511	344	234	473	293	256	505

Table D.5 (Continued)

25	4	370	288	502	405	287	529	383	326	516
26	4	431	219	519	455	211	530	434	173	520
27	4	540	259	576	512	275	553	504	319	576
28	4	277	211	574	258	189	569	246	213	529
29	4	264	382	597	296	387	579	281	349	561
30	4	539	537	599	528	490	585	562	489	565
31	4	372	195	612	377	170	585	379	230	581
32	4	251	234	629	281	254	604	275	275	640
33	4	355	159	664	392	158	672	396	177	641
34	4	303	214	701	328	182	689	304	200	654
35	4	402	196	683	370	212	696	367	204	667
36	4	462	173	657	424	183	660	475	196	686
37	4	321	173	757	302	162	706	276	182	736
38	4	192	349	685	218	348	657	229	380	682
39	4	195	370	741	173	385	771	151	334	746
40	4	181	302	741	206	285	770	161	273	780
1	6	484	358	10	473	343	36	456	315	20
2	6	539	341	102	518	339	129	498	353	91
3	6	484	330	147	463	361	157	494	394	131
4	6	230	360	142	239	345	167	220	306	148
5	6	484	532	176	466	563	168	440	548	186
6	6	176	267	170	188	260	204	224	281	209
7	6	292	406	198	259	408	219	240	376	205
8	6	558	272	207	567	295	224	553	284	251
9	6	453	285	252	438	330	228	419	329	264
10	6	350	164	245	391	175	259	373	135	247
11	6	540	360	282	523	321	269	535	362	250
12	6	240	328	255	244	327	222	193	305	221
13	6	186	561	237	203	545	221	233	585	242

Table D.5 (Continued)

14	6	321	541	312	352	520	303	373	558	309
15	6	325	173	351	311	214	334	268	191	353
16	6	602	353	349	562	345	335	592	318	351
17	6	283	467	334	325	454	352	313	496	334
18	6	260	300	343	242	294	358	228	285	326
19	6	546	280	389	533	260	416	527	231	383
20	6	539	417	425	562	406	390	582	374	419
21	6	143	303	414	129	257	426	169	247	423
22	6	199	457	409	226	442	428	186	412	428
23	6	338	137	453	338	119	474	369	144	486
24	6	342	247	483	328	224	449	280	255	483
25	6	362	285	485	383	283	523	376	323	499
26	6	423	214	514	447	203	525	427	169	506
27	6	537	262	576	511	276	555	497	318	575
28	6	245	204	539	226	183	544	210	195	501
29	6	232	386	561	262	394	543	249	354	524
30	6	517	558	579	514	507	566	549	507	550
31	6	365	187	587	369	162	561	369	221	553
32	6	223	238	589	250	257	564	247	278	599
33	6	338	153	632	375	143	640	383	166	615
34	6	279	217	658	303	182	645	279	201	612
35	6	390	191	643	361	214	650	360	206	621
36	6	464	151	631	428	166	635	480	180	654
37	6	298	174	711	277	162	662	253	183	692
38	6	166	351	652	188	351	621	200	382	646
39	6	169	369	706	151	384	737	127	332	711
40	6	157	302	704	184	288	733	140	272	743
1	8	484	357	9	475	344	34	455	313	19
2	8	540	341	98	519	340	125	498	353	88

Table D.5 (Continued)

3	8	485	330	144	466	363	154	501	393	129
4	8	233	360	140	243	344	166	222	306	147
5	8	485	531	173	469	562	166	442	549	184
6	8	179	268	168	189	262	203	227	282	207
7	8	293	406	193	262	408	217	241	375	205
8	8	562	272	201	571	295	219	558	283	247
9	8	456	285	250	444	326	226	422	330	261
10	8	352	162	242	393	171	255	376	130	244
11	8	544	361	278	526	322	266	539	361	245
12	8	238	330	254	244	328	221	194	304	215
13	8	182	572	221	202	554	212	229	592	236
14	8	322	543	307	353	520	300	374	560	303
15	8	323	167	343	308	205	325	266	183	342
16	8	607	353	346	567	343	330	597	317	344
17	8	281	470	328	322	452	349	310	501	333
18	8	244	298	332	227	290	348	209	280	319
19	8	551	280	384	539	256	411	533	228	376
20	8	546	422	422	567	407	385	589	377	416
21	8	118	305	384	106	262	397	147	250	393
22	8	177	462	375	202	444	393	158	420	391
23	8	331	127	440	334	106	458	366	125	469
24	8	330	239	463	318	218	428	270	247	462
25	8	354	276	465	369	270	506	374	311	483
26	8	422	205	499	441	188	513	428	160	479
27	8	551	258	560	531	274	534	511	317	558
28	8	234	200	506	207	190	516	196	194	469
29	8	214	394	521	243	402	501	230	363	483
30	8	526	575	556	521	525	551	558	525	533
31	8	363	173	562	357	146	539	374	201	524

Table D.5 (Continued)

32	8	206	251	546	231	269	520	231	285	558
33	8	330	133	601	364	121	612	379	143	587
34	8	270	223	618	287	188	598	262	212	569
35	8	382	178	603	353	200	612	355	198	583
36	8	466	132	603	429	145	597	480	161	626
37	8	287	175	668	264	165	620	243	190	652
38	8	152	358	616	173	358	584	187	388	609
39	8	158	375	671	141	389	703	116	337	678
40	8	145	308	668	174	292	696	130	276	708

Table D.6. Central top and bottom input coordinates of sample 3.

Strain (%)	Xtop	Ytop	Ztop	Xbottom	Ybottom	Zbottom
0	376.0	368.0	14.5	371.0	356.0	366.5
2	376.0	368.0	16.0	371.0	356.0	362.0
4	375.0	367.0	18.0	371.0	356.0	357.0
6	374.0	367.0	15.0	371.0	357.0	347.0
8	366.0	361.0	12.0	370.0	354.0	341.0