SPATIOTEMPORAL MEASUREMENT OF FREEZING-INDUCED DEFORMATION OF ENGINEERED TISSUES

by

KA YAW TEO

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN BIOMEDICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2009

ACKNOWLEDGEMENTS

This work is supported by Grant R01 EB008388 from NIH/NIBIB and Grant CBET-0747631 from NSF. My heartfelt gratitude goes to:

Dr. Bumsoo Han, for his unflagging encouragement, wisdom, and generosity;

Dr. J. Craig Dutton, for sound advice and a ready ear;

Dr. Cheng-Jen Chuong, for his unfailing patience, understanding and support;

Dr. Hyejin Moon, for her help with the optical contact angle measuring system;

Jun-Kyu Jung, for kindly sharing his knowledge and experience;

My fellow lab mates, past and present, for their invaluable companionship;

Ka Swan and Ka Wee, for always being supportive of my academic quest;

My parents, for always believing in me.

November 19, 2009

ABSTRACT

SPATIOTEMPORAL MEASUREMENT OF FREEZING-INDUCED DEFORMATION OF ENGINEERED TISSUES

Ka Yaw Teo, M.S.

The University of Texas at Arlington, 2009

Supervising Professor: Bumsoo Han

In order to cryopreserve functional engineered tissues (ETs), the microstructure of extracellular matrix (ECM) as well as the cellular viability should be maintained given that the functionality of ETs is closely related to the ECM microstructure. Since the post-thaw ECM microstructure is determined by the deformation of ETs during cryopreservation, the freezing-induced deformation of ETs was measured in the present study with a newly developed quantum dot (QD)-mediated cell image deformetry technique using dermal equivalents as a model system. The dermal equivalents were constructed by seeding QD-labeled fibroblasts in type I collagen matrices. After 24 hour incubation, the ETs were directionally frozen by exposing them to a spatial temperature gradient (from 4 ℃ to -20 ℃ o ver a distance of 6 mm). The ETs were consecutively imaged while being frozen. Consecutive pairs of these images were two-dimensionally cross-correlated to determine the local deformation during freezing. The results showed that freezing induced the deformation of ET, and its magnitude varied with both time and location. The maximum local dilatation was 0.006 s^{-1} and was always observed at the phase change interface. Due to this local expansion, the unfrozen region in front of the freezing interface experienced compression. This expansion-compression pattern was observed

throughout the freezing process. In the unfrozen region, the deformation rate gradually decreased away from the freezing interface. After freezing/thawing, the ET experienced an approximately 28% decrease in thickness and 8% loss in weight. These results indicate that freezing-induced deformation caused the transport and extrusion of interstitial fluid. In summary, the results suggest that complex cell-fluid-matrix interactions occur within ETs during freezing, and these interactions determine the post-thaw ECM microstructure and eventual post-thaw tissue functionality.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

As tissue engineering technology develops, reliable long-term preservation technology is highly desired to provide "off-the-shelf" availability of various engineered tissues (ETs) [1,2]. Reliable preservation technology is also desired for more effective storage, banking as well as transportation of ETs. This is one of the critical elements for tissue engineering in order for it to be scaled up from the laboratory to the manufacturing scale [3]. In order to address this need, various preservation techniques have been proposed and studied for a wide variety of biomaterials [4,5]. These techniques include hypothermic preservation, cryopreservation, vitrification, freeze-drying, and desiccation. Although all of the above techniques show promise for tissue preservation, cryopreservation, which preserves biomaterials in the frozen state, is still the primary candidate for long-term storage of ETs considering the limitations of preservation time window, toxicity associated with high concentration of cryoprotective agents, and the feasibility of sublimation and desiccation [6,7].

Several critical challenges, however, should be addressed for successful cryopreservation of ETs. One of the most significant challenges is the lack of consistency in maintaining the functionality of ETs [5]. The functionality is associated with mechanical, optical and transport properties of ETs, and is crucial to the physiological function of ETs. These functional properties are closely related to, or are often determined by, the microstructural integrity of the extracellular matrix (ECM). Therefore, the functional tissue's ECM microstructure as well as the cellular viability should be maintained during cryopreservation [8-10].

Freezing/thawing (F/T) of tissue during cryopreservation causes multi-scale biophysical phenomena at the molecular-, cellular- and tissue-levels. The molecular and cellular level freezing-induced biophysical phenomena include: 1) intracellular ice formation (IIF), which is

spontaneous ice crystallization of intracellular water during rapid freezing [11,12]; 2) extracellular ice formation (EIF) and consequent osmotic pressure-driven cellular dehydration during slow freezing [13,14]; and 3) phase change and transition of lipids and proteins [15-18]. Although F/T-induced cellular damage has been explained with the 'two-factor' hypothesis [19], all of these biophysical events are thought to affect the post-thaw cellular viability.

Among the tissue-level phenomena, osmotic pressure-driven water transport between intracellular and extracellular spaces, and the formation of extracellular ice, which is thought to alter the ECM microstructure, have been investigated. In these studies, water transport and morphology of the extracellular ice were investigated using small tissue samples or compartment-based models [20-23]. Although these experimental studies provided physical insights on the temporal nature of EIF, the spatial aspects of EIF and subsequent damage to the ECM were neglected. Moreover, any interactive roles in the F/T-induced biophysical phenomena were not considered because the ECM was recognized as a passive scaffold.

A few recent studies suggested that freezing of tissue induces the spatial and temporal redistribution of interstitial fluid, subsequent spatiotemporal ECM swelling, and post-thaw microstructural changes of the ECM [24,25]. These studies are based on the hypothesis that spatiotemporal fluid-matrix interaction is induced during freezing of tissues. This interaction includes: 1) volumetric expansion during water-ice phase change; 2) interstitial fluid transport from the freezing interface; 3) swelling of the unfrozen ECM to accommodate the interstitial fluid; and 4) reciprocal action of the ECM through the interstitial fluid pressure (IFP)-stress balance. The results suggested that freezing might induce complex spatiotemporal interactions between the interstitial fluid and the ECM, and these interactions might cause post-thaw ECM structural alteration. However, these studies were based on observation of the post-thaw ECM microstructure, and the role of cells in this interaction was not considered. Thus, a real-time understanding of F/T-induced cell-fluid-matrix interaction is currently lacking in order to model the spatiotemporal deformation of the ECM, which is a direct measure of the post-thaw ECM

microstructure. However, quantitative spatiotemporal measurement of this F/T-induced interaction is extremely challenging due to the lack of reliable non-invasive, dynamic, and multi-scale measurement techniques, especially for micro-/meso-scale tissue deformation.

In the present study, a new experimental technique was developed to measure spatiotemporal deformation of an engineered tissue during freezing in order to provide quantitative information on F/T-induced cell-fluid-matrix interaction. As a model ET, dermal equivalent was prepared by seeding quantum dot (QD)-labeled human fibroblast in type I collagen matrix. The ET with QD-labeled cells was imaged during freezing with a fluorescence microscope and further analyzed for micro-/meso-scale tissue deformation. The results are discussed in the context of freezing-induced cell-fluid-matrix interaction relevant to the post-thaw ECM structural changes of tissues. These results are expected to provide insights to explain the effects of freezing on the functionality of tissues.

CHAPTER 2

MATERIALS AND METHODS

2.1 Cell Culture and Reagents

Early-passage human foreskin fibroblasts, obtained from Dr. Frederick Grinnell (Department of Cell Biology, University of Texas Southwestern Medical Center), were maintained in culture medium (DMEM/F12, Invitrogen, Grand Island, NY) supplemented with 10% fetal bovine serum, 2 mM L-glutamine, and 100 µg/mL penicillin/streptomycin. The fibroblasts were cultured up to the 15th passage in 75 cm^2 T-flasks at 37 ℃ and 5% CO_2. The cells were consistently harvested at 80% confluency by using 0.05% trypsin and 0.53 mM EDTA.

The collected cells were labeled with quantum dots (Qtracker 655, Invitrogen, Carlsbad, CA) according to the protocol suggested by the manufacturer. Briefly, a 10 nM labeling solution was prepared by mixing 1 µL of Qtracker Component A with 1 µL of Qtracker Component B in a 1.5 mL centrifuge tube. The solution was incubated at 37 ℃ for 5 minutes. 200 µL of complete culture medium was then added to the tube, followed by vortexing for 30 seconds. Subsequently, the cells were added to the tube containing the labeling solution and incubated at 37 ℃. The cellular uptake of quantum dots (QDs) was characterized and is shown in Figure 2.1. The amount of intracellular QDs was measured by quantifying the average fluorescence intensity per cell using image-processing software (ImageJ, NIH). Figure 2.1 shows that the cellular uptake generally increased as the incubation time increased. The most rapid uptake was observed in the first 30 minutes, and then the uptake increased linearly with time at a somewhat slower rate. In the present study, the fibroblasts were labeled for 30 minutes. After incubation the cells were washed twice with complete culture medium to remove excess QDs.

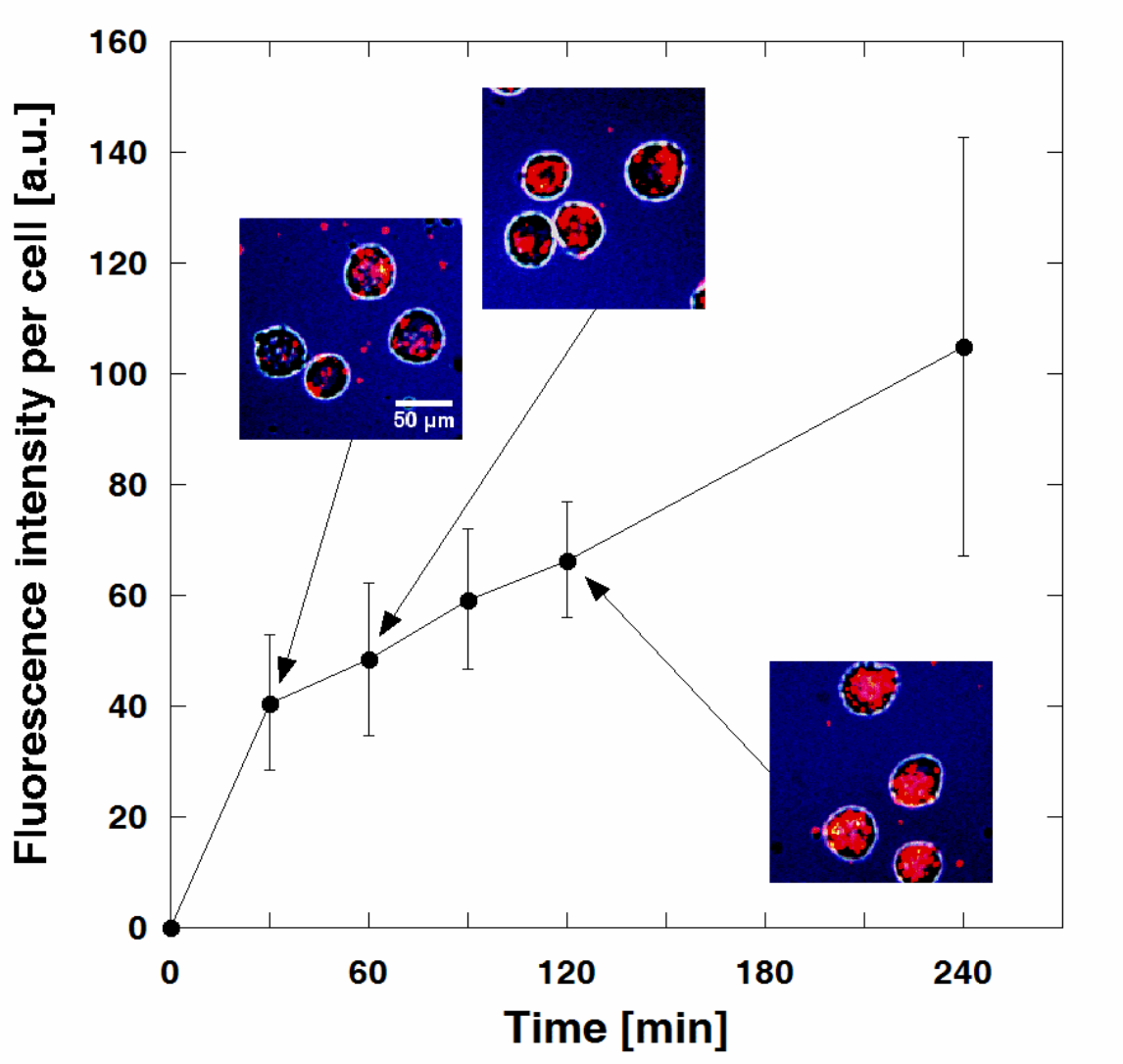
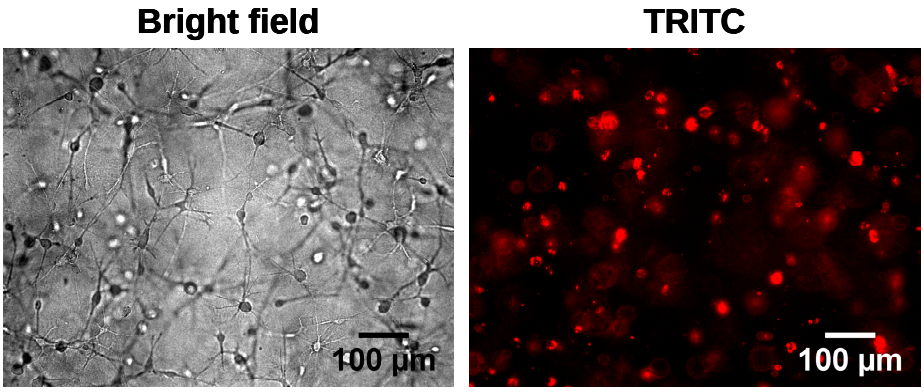


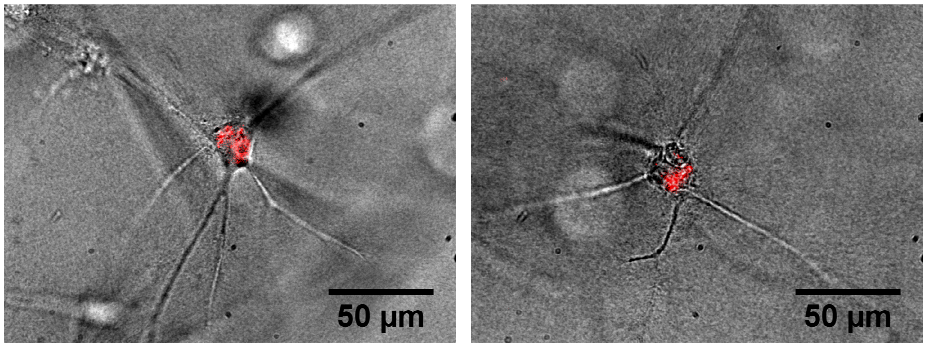
Figure 2.1 Quantum dot endocytosis characteristic of human foreskin fibroblasts (*n* = 3 for each time point). Intracellular quantum dot fluorescence intensity increases sharply with time in the first 30 minutes, and continues to increase almost linearly thereafter. Fluorescence images confirm the increase of quantum dot accumulation in fibroblasts over time.

2.2 Engineered Tissue with Quantum Dot-Labeled Cells

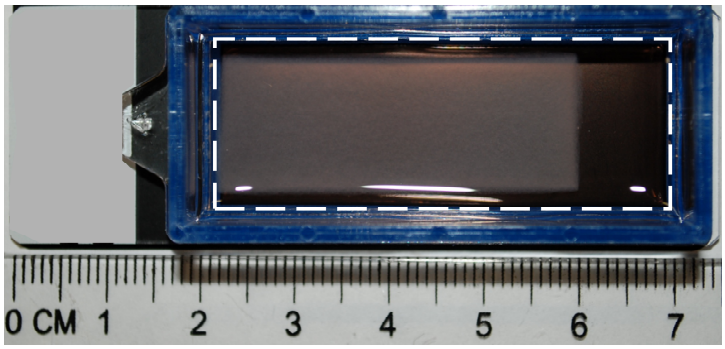
The QD-labeled cells were suspended in 2 mL of collagen solution prepared from high concentration type I rat tail collagen (BD Biosciences, Bedford, MA). The solution contained 3 mg/mL collagen, 10% 10X MEM (Minimum Essential Medium), 30 mM HEPES (4-(2-hydroxyethyl)-1-piperazineethanesulfonic acid), 10 μg/mL penicillin/streptomycin, 2 mM L-glutamine, 6% fetal bovine serum, and 2.3% (v/v of collagen added) 1N sodium hydroxide. Then, distilled water was added to make a total volume of 2 mL. The collagen solution was placed in a 48 × 18 mm chamber slide (Lab-Tek II, Nunc, Naperville, IL) and allowed to polymerize at 37 ℃ for 60 minutes. After polymerization 2 mL of complete medium was added and the ET was incubated for 24 hours before the freezing procedure. Figure 2.2 (a) shows that fibroblasts cultured in a collagen matrix develop a dendritic morphology after 24 hours of incubation. As shown in the corresponding fluorescence micrographs, a majority of the cells are still labeled with QDs after 24 hour incubation. Figure 2.2 (b) clearly shows that QDs accumulate in the intracellular space. Figure 2.2 (c) shows the typical fibroblast-driven compaction of the ETs. After incubation for 24 hours, the ETs contract from initial dimensions of 48.0 ± 0.1 × 18.0 ± 0.1 mm (dashed line) to 39.1 ± 0.6 × 14.8 ± 0.3 mm (light area) (n = 4). The morphology of the fibroblast and extent of collagen gel compaction observed are comparable to those reported elsewhere [26-29]. This implies that the effects of QD labeling are minimal on the behavior of fibroblasts in collagen matrices.



(*a*)



(*b*)



(*c*)

Figure 2.2 Bright field and fluorescence micrographs and top view macrograph of engineered tissue. (a) Fibroblasts, evenly embedded in collagen matrix, assume a dendritic morphology and are individually labeled with TRITC-fluorescence quantum dots. (b) Quantum dots specifically accumulate in the cytoplasm of fibroblasts. (c) Engineered tissue (light area) is compacted by its entrapped fibroblasts to a small fraction of its original size (dashed line).

2.3 Deformation Measurement during Freezing

An experimental method, named cell image deformetry (CID), was developed to measure the spatiotemporal deformation of ETs during freezing. The ET was frozen on a directional solidification stage as depicted in Figure 2.3. The stage consists of two independently controlled temperature reservoirs separated by a distance of 6 mm. By setting the temperatures of the reservoirs to 4 ℃ and -20 ℃ respectively, a spati al temperature gradient was imposed on the ET, causing it to freeze along the *x* direction. A fluorescence macro/microscope (MVX10, Olympus, Center Valley, PA) equipped with a long working distance objective lens and a TRITC (tetramethylrhodamine isothiocyanate) filter was used to visualize the QD-labeled cells of the ET during freezing. The ET was continuously imaged at 1 frame/second using a high sensitivity CCD camera (PIXIS 512, Princeton Instruments, Trenton, NJ). The acquired sequential images were cross-correlated at a 10 second interval to estimate the local deformation rates. For the cross-correlation, consecutive pairs of images at the 10 second interval were divided into 32 pixel $\times$ 32 pixel interrogation windows, and were cross-correlated using commercial software (DaVis 7.1, LaVision, Ypsilanti, MI) to determine the local deformation rates (μm/s) in each interrogation window. The experiments were repeated three times to determine the average deformation rates, *u* and *v*, along the *x* and *y* directions, respectively (n = 3). These deformation rates were analyzed to estimate strain rates (s^{-1}) as follows.

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} \approx \frac{\Delta u}{\Delta x} \text{ and } \varepsilon_{yy} = \frac{\partial v}{\partial y} \approx \frac{\Delta v}{\Delta y} \tag{1}$$

Dilatation (s^{-1}) defined as the rate of area increase per unit area, i.e., expansion, was also computed as follows.

$$e = \varepsilon_{xx} + \varepsilon_{yy} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \approx \frac{\Delta u}{\Delta x} + \frac{\Delta v}{\Delta y} \tag{2}$$

The central difference scheme was used to approximate the spatial derivatives needed to compute the strain rates.

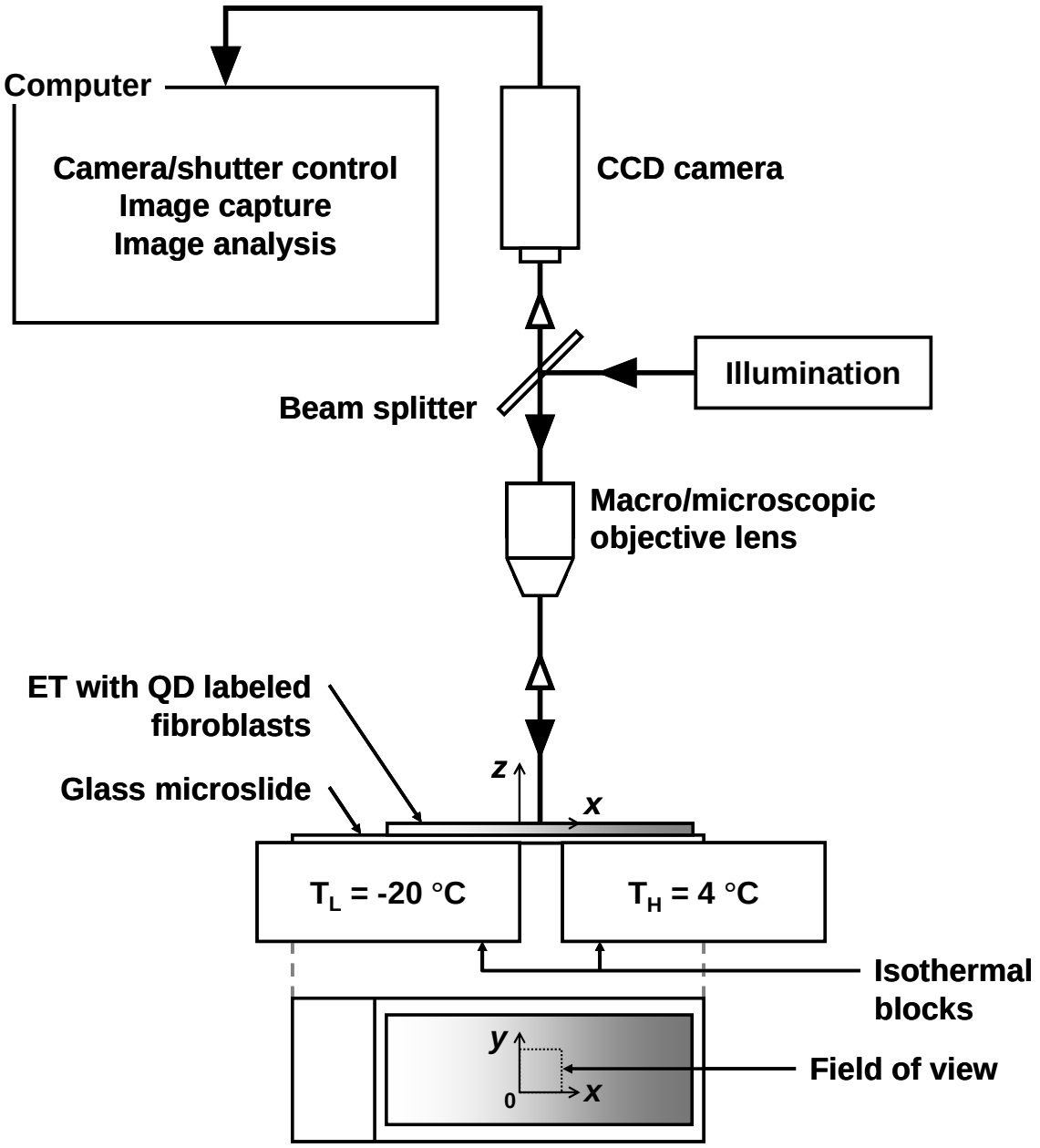


Figure 2.3 Schematic of cell image deformetry experimental setup. When engineered tissue is frozen on the directional solidification stage, it is continuously imaged while being illuminated with an excitation light, causing the quantum dots within the embedded fibroblasts to fluoresce.

2.4 Assessment of Post-Thaw Engineered Tissue Thickness and Weight

The thickness profile of the ET sample was imaged using an optical contact angle measuring system (CAM 101, KSV, Monroe, CT) after F/T. The image was further processed to quantify the ET thickness change. In addition, the weight of the ET was measured before and after F/T using a balance with a precision of 0.1 mg (AV114, Ohaus, Pine Brook, NJ).

2.5 Statistical Analysis

Each experiment was repeated at least 3 times ($n \geq 3$). The results were presented as mean ± standard error of the mean (SEM). Statistical analysis was performed using one-way ANOVA; $P < 0.05$ was considered statistically significant.

CHAPTER 3

RESULTS

Figure 3.1 (a) illustrates the procedures of the image processing to determine the deformation rates. A pair of consecutive fluorescence micrographs at a 10 second interval was obtained, and each image is divided into a grid of 32 × 32 pixel interrogation windows. The cross-correlation function of each interrogation window pair (original and delayed images) is computed as described in [30], and the location of the maximum peak of the cross-correlation function yields the average deformation for a given interrogation area. Dividing by the time interval between images then gives the deformation rate. Note that this process yields the deformation rates in both the *x* and *y* directions, i.e., *u* and *v*. Since each interrogation window contains multiple cells, the estimated deformation rate is a measure of local ET deformation averaged over the window rather than for individual cell movements. By performing the same process for each interrogation window, a vector field of the deformation rate is determined for the entire plane of the pair of fluorescence images. The deformation vector field overlapped with the corresponding fluorescence micrograph is showed in Figure 3.1 (b). This result demonstrates that the CID method can quantify the deformation rates of ETs during freezing. Although some scattering and reflection of the QD fluorescence occur in the frozen region, they do not affect the capability of the CID method if the camera is focused. Moreover, this overlapped image indicates that the maximum deformation occurs at the location of the freezing interface. This is consistently observed at different times during the freezing process. Thus, we use the following condition to determine the location of the phase change interface at time *t*, *X(t)*:

$$\frac{\partial u}{\partial x} = 0 \text{ at the freezing interface} \tag{3}$$

The average phase change interface location, *X(t)*, determined by the locus of $\partial u/\partial x = 0$ (*n* =

3) is shown in Figure 3.2. Since the freezing configuration of the present experimental setup can be approximated as a moving boundary problem with phase change [31], the interface location is curve-fitted using the analytical Neumann solution as given below:

$$X(t) = 2\lambda\sqrt{\alpha_S t} \tag{4}$$

where λ is a parameter of the freezing condition, and α_S is the thermal diffusivity of ice (1×10^6 $\mu m^2/s$). The resulting curve fit is shown as a solid line in Figure 3.2. The parameter λ is determined to be 0.114. The inset in Figure 3.2 shows *X(t)* plotted against $2\sqrt{\alpha_S t}$, where the slope represents λ, and the R^2 value is 0.978. Although some discrepancy between the analytical curve fit and the experimental data is observed at early times, the fit generally agrees well with the experimental results.

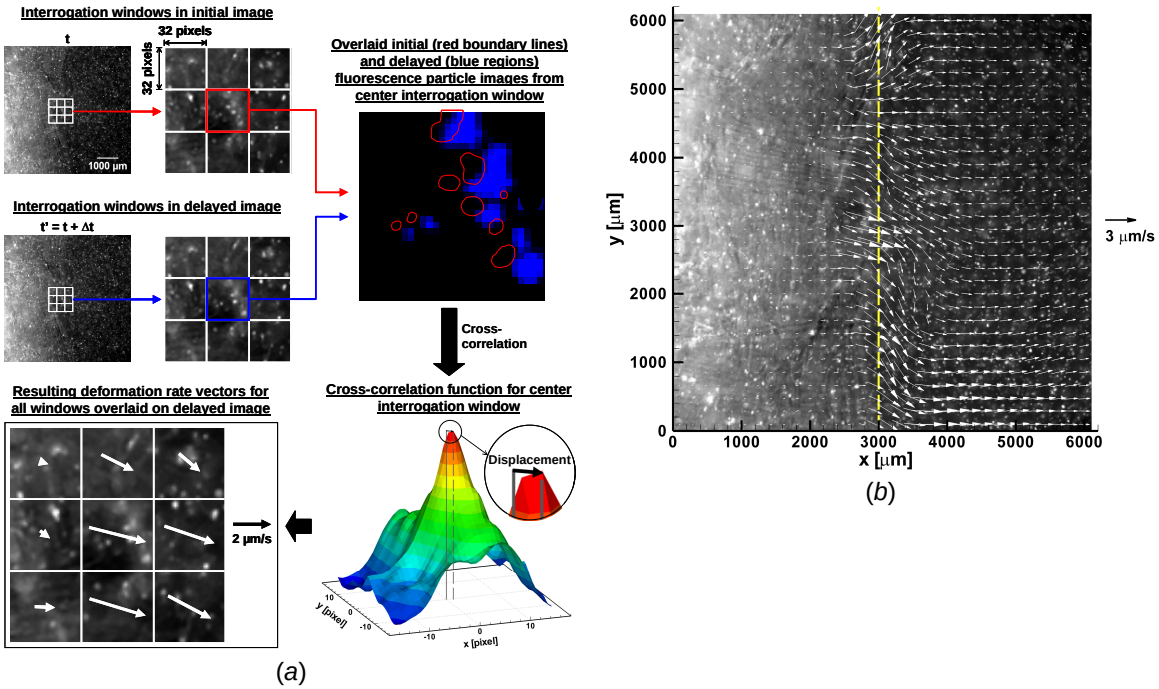


Figure 3.1 Cell image deformetry analysis flowchart. (a) Two fluorescence micrographs taken 10 seconds apart are divided into a series of 32 × 32 pixel interrogation windows. The interrogation windows in the initial and delayed images are cross-correlated to produce a map of correlation peaks. The location of the maximum correlation peak provides the two-dimensional displacement vector for the interrogation area. In order to improve the quality of cross-correlation, the interrogation windows are sometimes shifted toward estimated directions. (b) A deformation rate vector field is determined for the pair of fluorescence images shown in part (a) by using multi-pass processing with decreasing interrogation window size (1 iteration of 64 × 64 pixels followed by 2 iterations of 32 × 32 pixels) and 50% overlap. Overlaying the fluorescence image with the deformation rate vector field shows that the location of the freezing interface coincides with the location of maximum deformation rates.

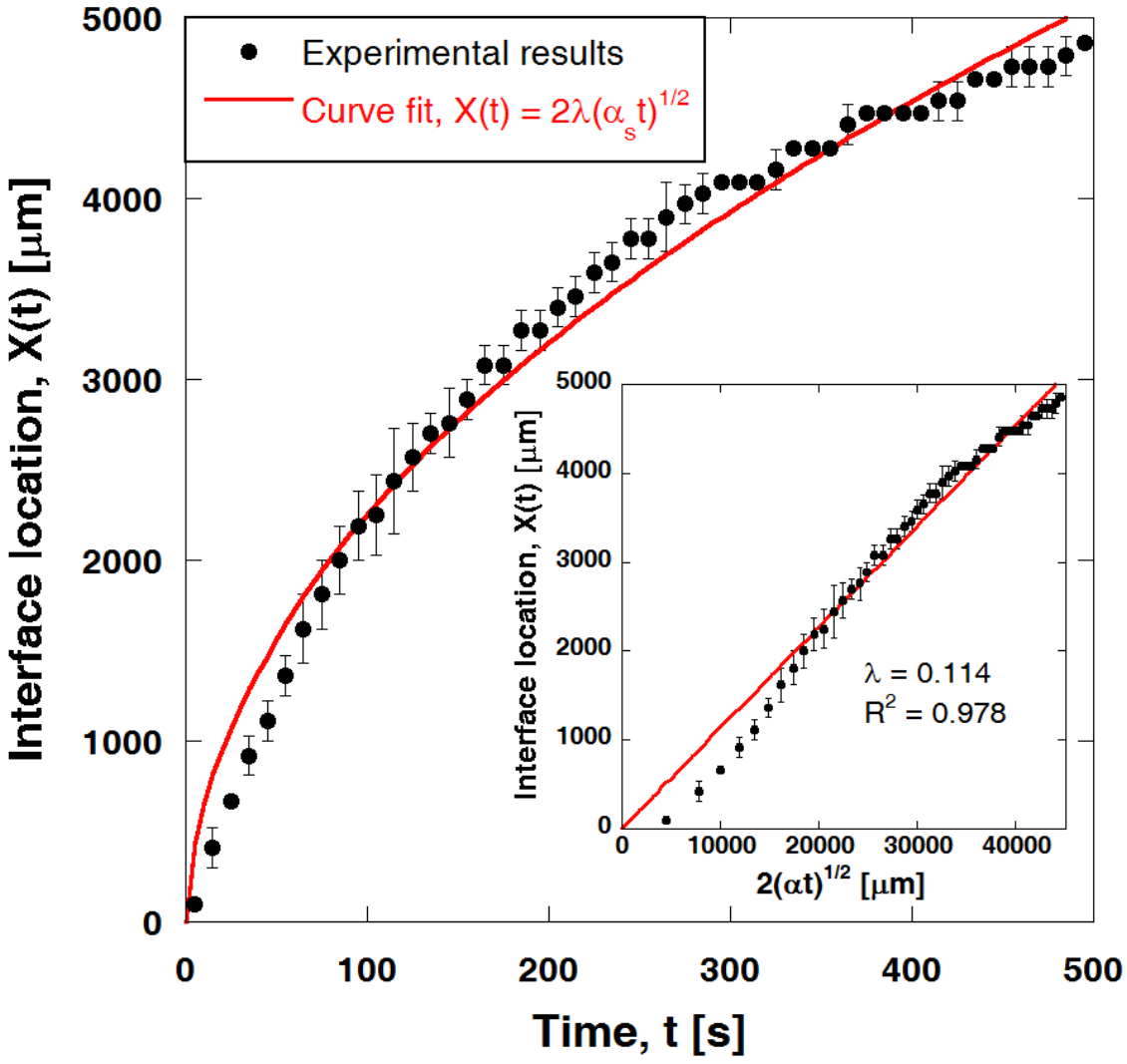
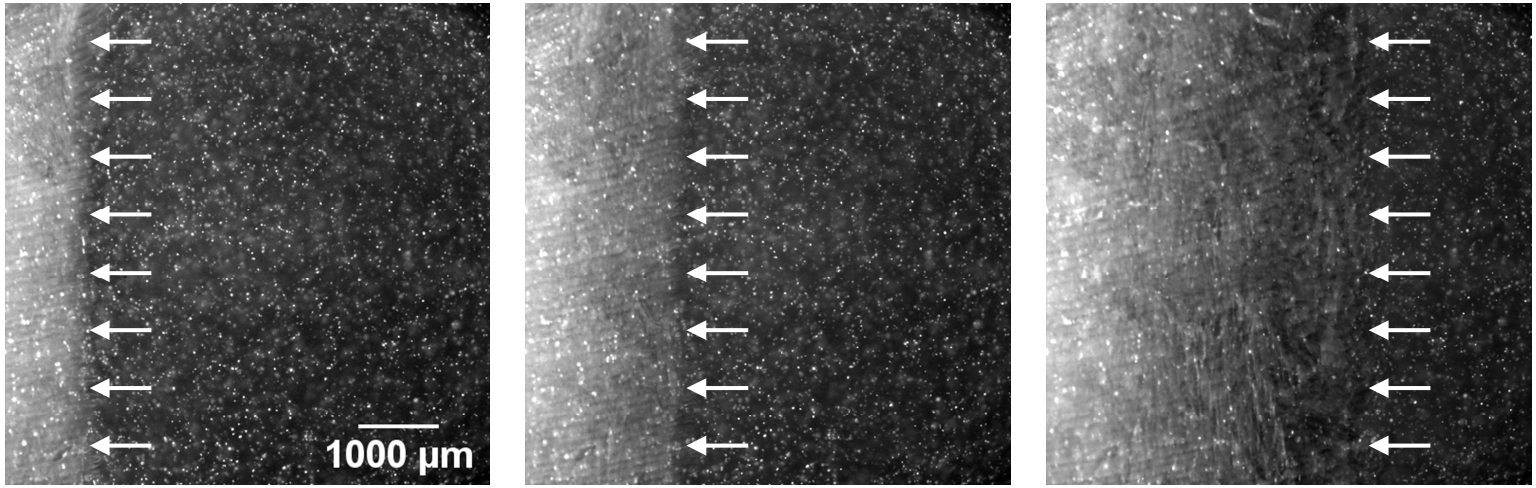


Figure 3.2 Phase change interface location during freezing and curve fit using analytical Neumann solution. Interface location is determined from displacement field using $\partial u / \partial x = 0$ or $u = u_{max}$ ($n = 3$). Interface location is subsequently curve-fitted to Neumann solution, $X(t) = 2\lambda(\alpha_S t)^{1/2}$ with $\alpha_S = 1 \times 10^6$ μm^2/s (thermal diffusivity of ice), yielding $\lambda = 0.114$. Inset shows $X(t)$ plotted against $2(\alpha_S t)^{1/2}$ along with a Neumann solution fitting curve ($R^2 = 0.978$).

Figures 3.3 (a)-(c) show fluorescence micrographs of an ET during directional freezing when $X(t) \approx 1000$, 2000 and 4000 μm. Since the cold base is located at $x = 0$, the freezing interface moves from left to right. The freezing interface is visible in the fluorescence micrographs and is noted with arrows. The interface initially appears to be a sharp front and gradually becomes "mushy" with an apparent thickness as it propagates. During freezing, the movement of the QD-labeled cells, which physically adhere to the ECM, reflects the average local deformation of the ECM. This yields the local details of the non-affine network deformation of the tissue [32].



(*a*) (*b*) (*c*)

Figure 3.3 Fluorescence micrographs of engineered tissue when (a) $X(t) \approx 1000$ μm, (b) $X(t) \approx 2000$ μm, and (c) $X(t) \approx 4000$ μm. Phase change interface, denoted by arrows, propagates from left to right while the tissue freezes one-dimensionally.

The corresponding deformation rates experienced by the ET (n = 3) are presented in Figures 3.4 (a)-(c). The ET undergoes local deformation while it is being frozen. The deformation measured is highly spatiotemporal in nature. As mentioned in regard to Figure 3.1, the maximum local deformation rate (≈ 2 μm/s) is observed at the freezing interface. In the unfrozen region, the displacement rate gradually decreases away from the interface. These results imply that freezing induces local deformation of the ET in both the frozen and unfrozen zones. The magnitude of the deformation rate varies with both time and location, and the maximum deformation rate is always observed at the phase change interface. Interestingly, even after freezing occurs, deformation is measured. This is probably due to the characteristics of the binary mixture phase change (i.e., water-NaCl) in which the unfrozen fraction of the water-NaCl solution still exists below the freezing temperature of water.

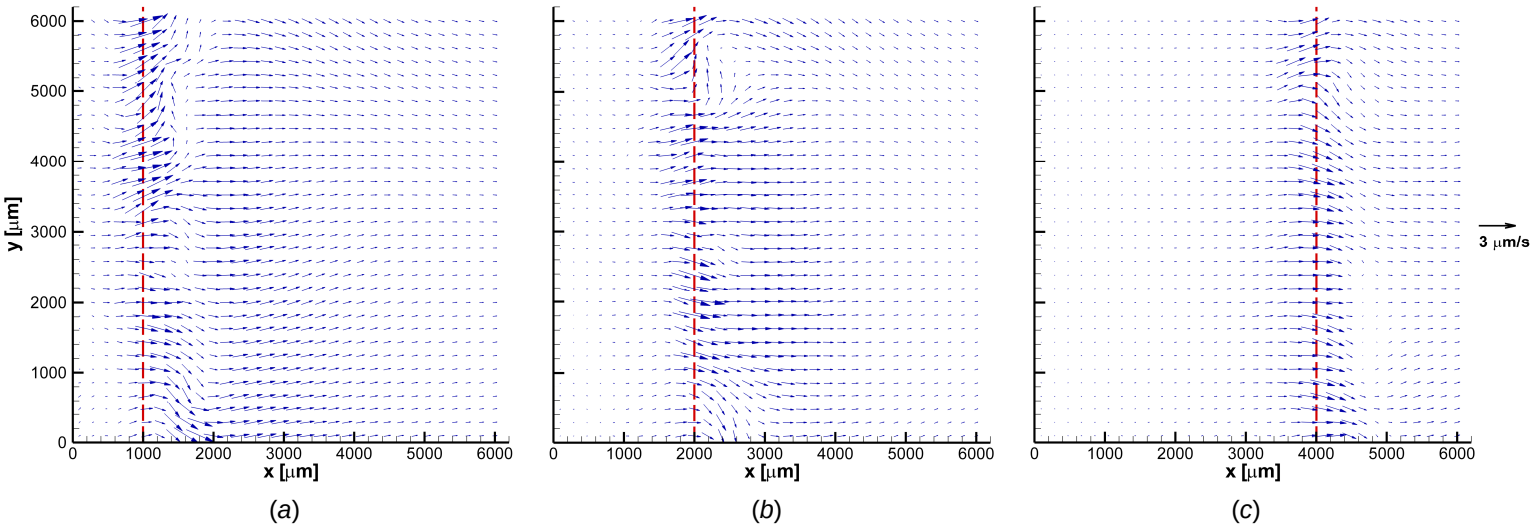


Figure 3.4 Deformation rate vector fields when (a) $X(t) \approx 1000$ μm, (b) $X(t) \approx 2000$ μm, and (c) $X(t) \approx 4000$ μm. Engineered tissue experiences highly spatiotemporal deformation during freezing. The maximum local deformation rate is observed at the freezing interface. Deformation rate progressively decreases away from the interface in the unfrozen region. A slight to no deformation rate is observed in the frozen region.

Figures 3.5 (a)-(c) show the corresponding dilatation of the ET (n = 3) when $X(t) \approx$ 1000, 2000 and 4000 μm. The ET is dilated and compressed during freezing in a highly spatiotemporal manner. Local expansion occurs just after the phase change interface, and the unfrozen region right ahead of the freezing interface shows a resulting compression. This expansion-compression pattern is observed throughout the freezing process. Since the maximum and minimum dilatations are observed adjacent to the freezing interface, these deformations are thought to be induced by freezing. However, the extent of this local deformation varies in time such that the maximum dilatation rate (maximum expansion rate) when $X(t) \approx$ 1000 μm is approximately 0.006 s^{-1} and decreases to approximately 0.002 s^{-1} when $X(t) \approx$ 4000 μm. Similarly, the minimum dilatation rate (maximum compression rate) varies from about -0.006 to -0.002 s^{-1} as the interface propagates from $X(t) \approx$ 1000 to 4000 μm. This deformation with expansion-compression provides new insight into freezing-induced tissue deformation. In addition to the deformation in the *xy*-plane, z-directional deformation has also been noticed, and it is confirmed by cells moving out of the focal plane (data not shown).

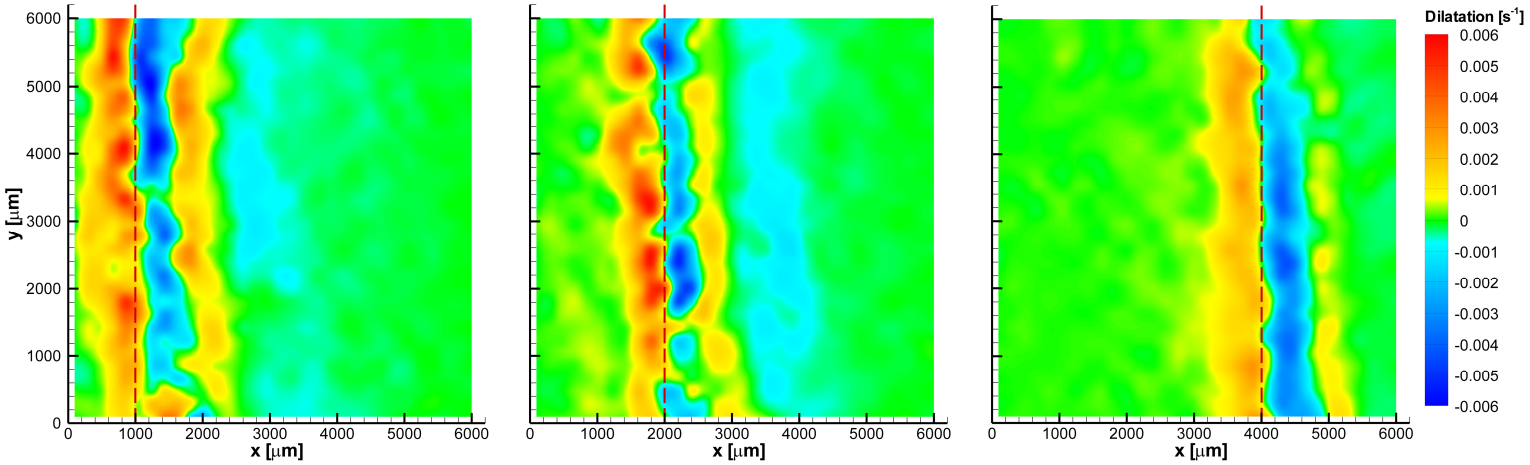


Figure 3.5 Dilatation contours when (a) $X(t) \approx 1000$ μm, (b) $X(t) \approx 2000$ μm, and (c) $X(t) \approx 4000$ μm. Engineered tissue is dilated and compressed in a highly spatiotemporal fashion during freezing. Engineered tissue experiences local expansion immediately following phase change, and a corresponding local compression is observed concurrently in the unfrozen region. Dilatation is approximately zero at the freezing interface.

Figure 3.6 shows the deformation rates and dilatation averaged along the y-axis (n = 3) for this nominally one-dimensional process. At the freezing interface, the ET experiences the maximum average horizontal displacement rate. The maximum average deformation rate in the x direction is about 1.8 μm/s and slightly decreases as the interface propagates. The error bars in Figure 3.6 represent the extent of variation along the y-axis. This variation decreases as the interface continues to advance, implying that the deformation rate becomes more uniform along the y direction (i.e., more one-dimensional) as the propagating interface slows. In addition, a second peak in the average x-deformation rate is measured at a distance of 1000 to 1500 μm ahead of the interface, and this second peak value decreases as the interface continues to propagate with decreasing velocity. As shown in Figure 3.6 (b), the y-deformation rates also have maxima at the interface. However, the magnitude is significantly smaller than for the x-deformations (< ± 0.5 μm/s) throughout the freezing process. The y-averaged dilatation (n = 3) is shown in Figure 3.6 (c). Since the major deformation occurs along the x-direction at the freezing interface, approximately zero dilatation is observed at the interface. The maximum average expansion rate (positive dilatation) occurs in the frozen region immediately behind the freezing interface. This peak of average dilatation starts at approximately 0.003 s^{-1} and decreases to slightly above 0.002 s^{-1} as the interface moves from $X(t) \approx$ 1000 to 4000 μm. The maximum average compression rate is measured as approximately -0.002 s^{-1} in the unfrozen region just ahead of the interface. It is also important to note that the standard variation in the y-averaged dilatation decreases as the interface advances further into the unfrozen region. Moreover, a second peak of average dilatation, which is comparably smaller in magnitude than the peak at the freezing interface, takes place in front of the compression in the unfrozen region. This second peak gradually disappears as the interface continues to propagate and slows.

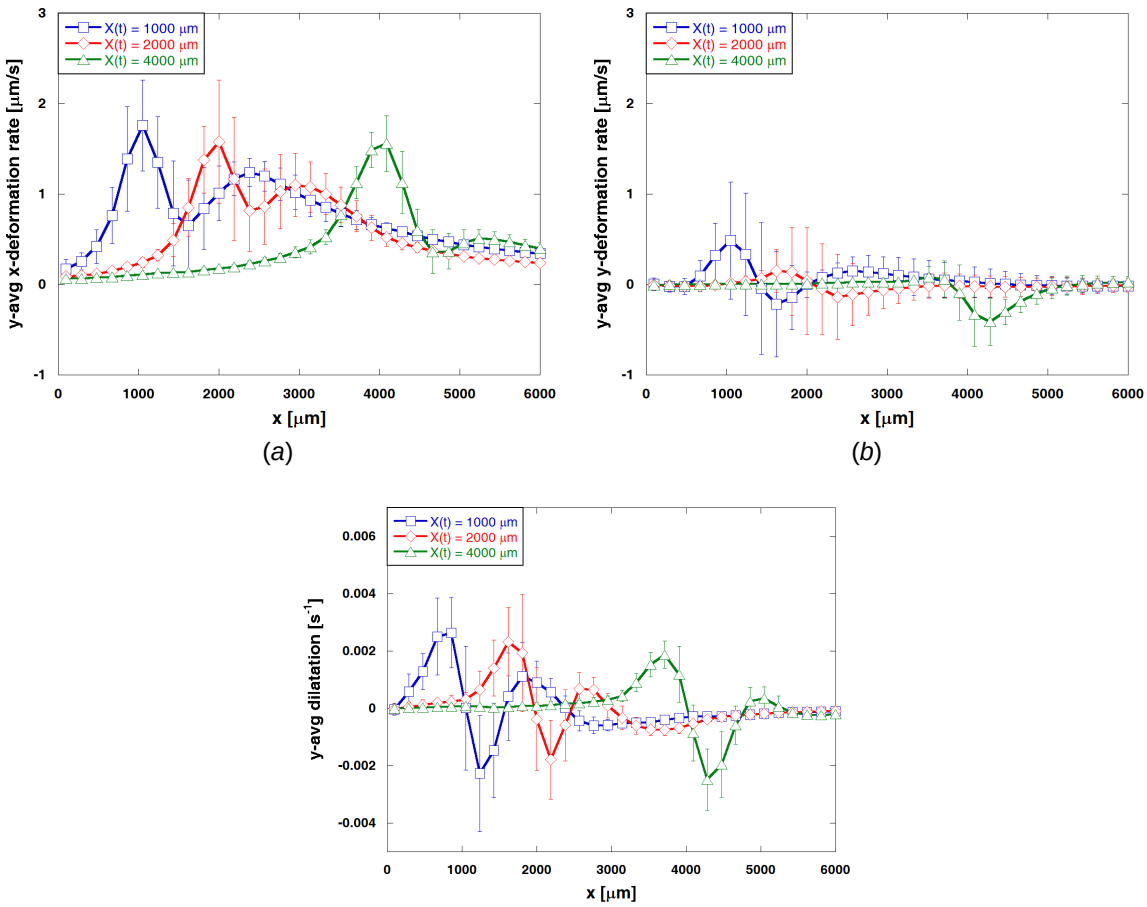
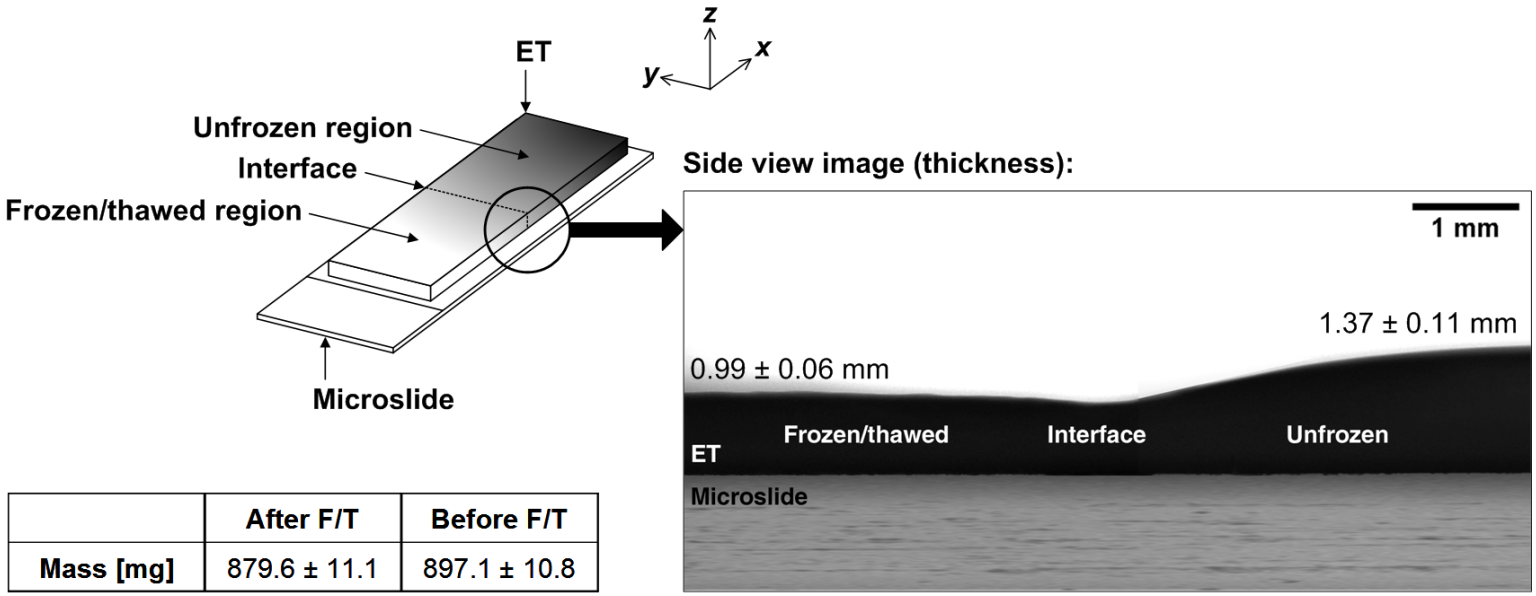


Figure 3.6 Averaged deformation rates and dilatation profiles. (a) *y*-averaged *x*-deformation rate, (b) *y*-averaged y-deformation rate, and (c) *y*-averaged dilatation when X(t) ≈ 1000, 2000, and 4000 μm. One-dimensional freezing induces deformation of engineered tissue primarily in the *x* direction with an average magnitude ranging from 0 to 1.8 μm/s. Average deformation rates in the *y* direction are much smaller than those in the *x* direction.

The post-thaw thickness profile of a typical ET sample is shown in Figure 3.7. The frozen/thawed region is significantly thinner than the unfrozen region ($P < 0.005$). The unfrozen region has an average thickness of 1.37 ± 0.11 mm ($n = 3$). That of the frozen region is 0.99 ± 0.06 mm ($n = 3$), thus representing a 27.7% reduction in thickness after F/T. In addition, the weight of the ET decreases significantly following F/T ($P << 0.001$). In particular, the ET initially weighs an average of 897.1 ± 10.8 mg ($n = 3$) and loses 7.8% of its weight after F/T (879.6 ± 11.1 mg; $n = 3$). A similar thickness change and weight loss after F/T were reported in [33,34]. The thickness change and weight loss are thought to be caused by the efflux of the interstitial fluid during freezing. While ET locally deforms during freezing, the interstitial fluid may flow away from the interface and be extruded from the ET. Based on our observations, the extruded fluid does not re-hydrate the ET after thawing.



	After F/T	Before F/T
Mass [mg]	879.6 ± 11.1	897.1 ± 10.8

Figure 3.7. Post-thaw profile of engineered tissue sample. Engineered tissue becomes significantly thinner after freezing and thawing. In addition, engineered tissue experiences a significant lost in mass after freezing and thawing. These changes in thickness and mass are thought to be caused by the efflux of interstitial fluid during freezing and thawing.

CHAPTER 4

DISCUSSION

4.1 Freezing-Induced Deformation of Engineered Tissue

The present results show that freezing induces the deformation of tissues and the deformation is highly spatial and temporal. Although the majority of previous studies on the effects of freezing on biological tissues [20-23] have focused on addressing its temporal aspects, the present results illustrate that freezing could induce both spatial and temporal deformation of an ET. The maximum local expansion occurs immediately following the freezing interface indicating that expansion is induced by freezing. Since the ET can be approximated as a poroelastic material saturated with interstitial fluid, mainly isotonic saline, its freezing is mainly attributed to the water/ice phase change. Since the expansion associated with ice formation is caused by the water/ice density change, the maximum possible expansion should be approximately 9% (i.e., the ratio of specific volume of ice to that of water = $\upsilon_{ice}/\upsilon_{water} = \rho_{water/}\ \rho_{ice}$ = (1000 kg/m^3)/(920 kg/m^3) = 1.09). The maximum freezing-induced tissue expansion measured in the present study (~ 6%) is less than that of ice formation. Even considering that the current methodology is a two-dimensional measurement of three-dimensional phenomena, this value is smaller than the volume expansion of water/ice phase change. This may be explained by the following reasons - 1) the interstitial fluid is a mixture of water and other solutes, and its phase change is different from that of pure water; and 2) complex cell-fluid-matrix interactions are associated with local tissue expansion. Even though water is the major component of the interstitial fluid, various solutes are dissolved in the fluid including salts (or electrolytes), carbohydrates, amino acids, and proteins. During freezing, the presence of these solutes affects the phase change behavior, including freezing point depression and eutectic phase change as

described in [35,36]. Furthermore, this could lead to osmotic pressure-driven water transport across the cellular membrane.

In addition, complex interactions among cells, interstitial fluid and ECM could affect the volumetric expansion. When tissue undergoes F/T, as proposed in [25], freezing induces interstitial fluid movement. This excess fluid transport may interact with the ECM as follows: 1) volumetric expansion during water-ice phase change; 2) interstitial fluid transport from the freezing interface; 3) local swelling of the ECM to accommodate the interstitial fluid transport; and 4) reciprocal action of the ECM through the interstitial fluid pressure (IFP)-stress balance. In addition to this fluid-matrix interaction, cells may play various roles in these freezing-induced phenomena which include - 1) cell-fluid interaction: both osmotic and interstitial fluid pressure-driven water transport across the cellular membrane; and 2) cell-matrix interaction: mechanical stress interaction induced by F/T-induced deformation. If the deformation caused by these interactions is beyond the tolerable range, this could lead to a post-thaw change of tissue functionality.

In addition to the expansion due to freezing (water/ice phase change) as mentioned earlier, local compression was observed in the unfrozen region, and was particularly significant in front of the freezing interface. This may be caused by the freezing-induced expansion in the adjacent frozen region. The compression in the unfrozen region may lead to the extrusion of the interstitial fluid out of the ET, which is confirmed by the following observations: 1) The frozen/thawed region is thinner than the unfrozen region; 2) the ET weighs less after freezing than that before freezing; 3) the frozen/thawed ET appears to be more translucent as compared to the unfrozen ET; (4) water accumulates on the outside surface of the ET after F/T. Controlling these freezing-induced spatiotemporal deformation and dehydration phenomena is thought to be critical in maintaining the functionality of ETs throughout cryopreservation protocols.

4.2 Measurements by Cell Image Deformetry

Various experimental techniques have been developed to measure micro-scale tissue deformation using various microscopy techniques including differential interference contrast microscopy [37], laser confocal microscopy [38-40], and multiphoton fluorescence microscopy [41]. In addition, tracking the movement of micron-sized fluorescent beads was also employed [42]. Cross-correlation of digital video microscopy was developed in [43] based on bright field or conventional fluorescence microscopy. Other recent studies [44,45] applied the micro-particle image velocimetry (PIV) technique to hydrogel deformation (i.e., displacement) measurements, but these studies need a relatively large amount of micron-sized seed particles (~ 1% v/v), which may alter the rheological and mechanical properties of the hydrogels. Many of these techniques are useful to investigate subcellular and cellular level deformation, but their effectiveness is limited - 1) when ice crystals are formed within the tissues; and 2) by photobleaching of conventional fluorescence dyes.

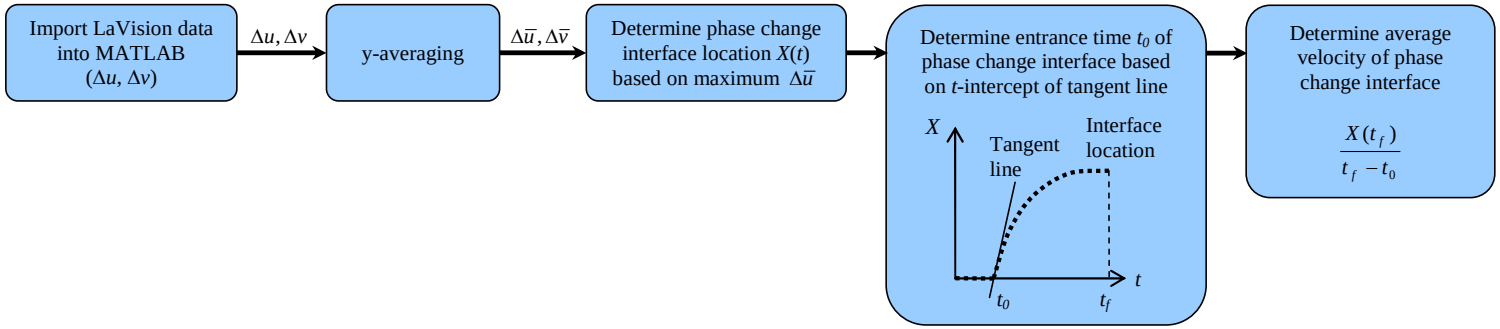
Cell image deformetry has been shown in our study to be a feasible method for measuring freezing-induced spatiotemporal deformation in ETs. It combines a high-resolution time-lapse digital microscopy system with PIV, a well-established digital image correlation technique for measuring continuum deformations (or velocities). The major advantage of the method lies in the fact that it is a non-intrusive optical visualization technique. No physical probe is used for measurement purposes, and the sample is not perturbed by the measuring instrument. The seed particles used are QD-labeled fibroblasts physically coupled to the collagenous ECM, thus introducing no foreign tracking particles that may undesirably alter the mechanical properties of the sample examined. As the tissue deforms on the microscale level during freezing, the embedded cells respond and move correspondingly, yielding information about the degree of deformation of their local environment. The capability of the method to continuously monitor the movement of the cells embedded in the collagen matrix during freezing makes it possible to visualize the phenomenon of freezing-induced tissue deformation in real time. Digital image

cross-correlation is then used to determine the extent of local tissue deformation quantitatively. In addition, the fluorescent labels used (i.e., QDs) are specific (i.e., cell-targeting) and robust for imaging purposes. The fluorescence signal of the QDs, unlike other fluorophores, is known to be remarkably stable and resistant to photobleaching.

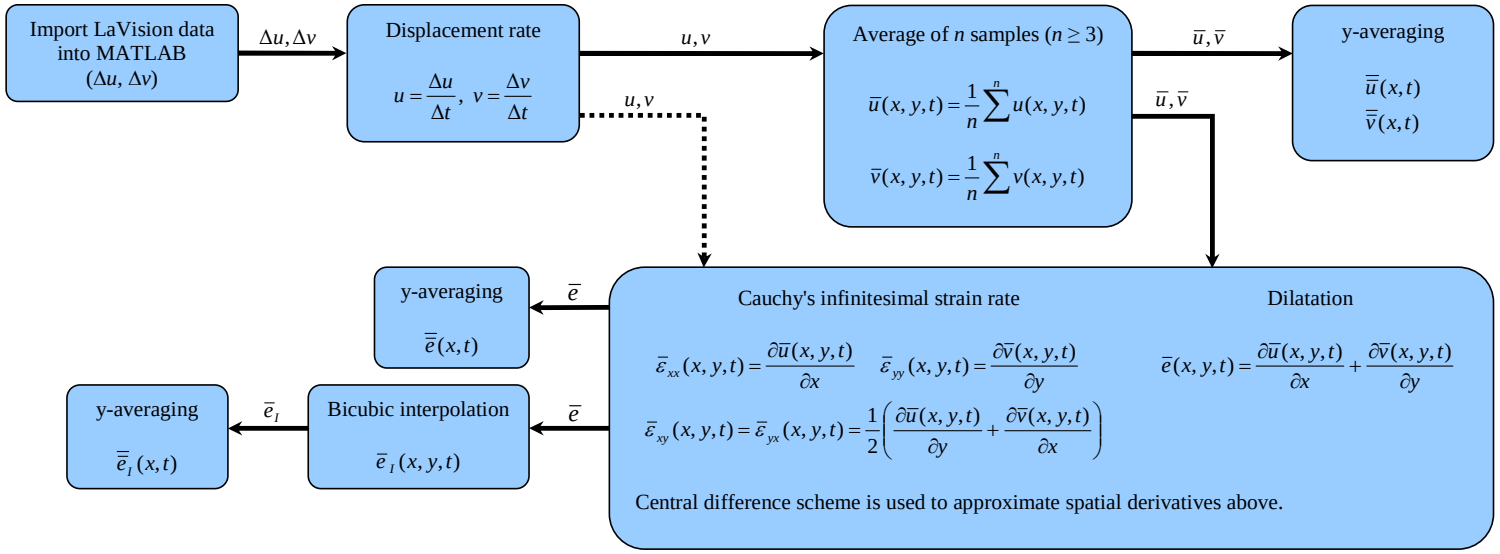
APPENDIX A

CID DATA ANALYSIS

Part 1: Determine phase change interface location and entrance time t_0



Part 2: Calculate strain rates and dilatation (after re-performing cross-correlation based on t_0)



```
% ExeSeq.m
% Ka Yaw Teo
% Last updated: 1/3/2009

%% SEQUENCE 1: Determine Time-Zero

% The following codes execute three .m files to
% (1) import delta u and delta v,
% (2) average delta u and delta v in y-direction, and
% (3) determine interface location, time-zero, and average interface velocity.

ImportDelDisp;
YAvgDelDisp;
InterLoc;

% NOTE: The sequences below should be executed after re-performing cross-correlation using time-zero.

%% SEQUENCE 2: Single-Set Post-Processing (Only At Specific Interface Locations)

% The following codes execute five .m files to
% (1) import delta_u and delta_v,
% (2) compute delta_u_avg and delta_v_avg by averaging delta_u and delta_v in y-direction,
% (3) compute delta_Exx, delta_Eyy and delta_e from delta_u and delta_v,
% (4) compute delta_e_avg by averaging delta_e in y-direction, and
% (5) determine interface location, and find delta_u, delta_v, delta_u_avg, delta_e, and delta_e_avg at specific
%     interface locations.

ImportDelDisp;
YAvgDelDisp;
CalcDelStrnDltn;
YAvgDelDltn;
SpecInterLocDispDltn;

%% SEQUENCE 3: Multi-Set Post-Processing (Only At Specific Interface Locations)

% The following codes execute five .m files to
% (1) import multiple sets of delta_u and delta_v,
% (2) compute delta_u_avg and delta_v_avg by averaging delta_u and delta_v in y-direction,
% (3) Determine specific interface locations for each data set,
% (4) Compute delta_u_sAvg and delta_v_sAvg by averaging delta_u and delta_v from multiple sets of data at pre-
%     determined specific interface locations,
% (5) Compute delta_u_sAvg_yAvg and delta_v_sAvg_yAvg by averaging delta_u_sAvg and delta_v_sAvg in y-direction,
% (6) Compute delta_Exx_sAvg, delta Eyy_sAvg and delta_e_sAvg from delta_u_sAvg and delta_v_sAvg, and
% (7) compute delta_e_sAvg_yAvg by averaging delta_e_sAvg in y-direction.

ImportDelDisp_MultiSets;
YAvgDelDisp_MultiSets;
```

```
SpecInterLoc_MultiSets;
SampAvgSpecInterLocDelDisp_MultiSets;
YAvgSampAvgSpecInterLocDelDisp_MultiSets;
CalSampAvgSpecInterLocDelStrnDltn_MultiSets;
InterpSampAvgSpecInterLocDelDltn_MultiSets;
YAvgSampAvgSpecInterLocDelDltn_MultiSets;

%% SEQUENCE 4: Determine and Curve-Fit Interface Location from Multiple Sets of Data

% The following codes execute four .m files to
% (1) import multiple sets of delta_u and delta_v,
% (2) compute delta_u_avg and delta_v_avg by averaging delta_u and delta_v in y-direction,
% (3) determine interface location over time for each data set, and
% (4) compute avgInterLoc by averaging interface locations from multiple sets of data,
%     and curve-fit sample-averaged interface location (by using power curve fitting).

ImportDelDisp_MultiSets;
YAvgDelDisp_MultiSets;
InterLoc_MultiSets;
AvgInterLocFit_MultiSets_New;
```

```
% ImportDelDisp.m
% Ka Yaw Teo
% Last updated: 7/15/2008

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% delta_u(i,j,k) where i = 1:1:xDispCoorNum, j = 1:1:yDispCoorNum, and k = 1:1:tStepNum
% delta_v(i,j,k)
% xDispCoorNum
% yDispCoorNum
% xDisp(i)
% yDisp(j)
% tStep
% tStepNum
% t(k)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Importing data files (SPECIFY dataNum)
```

```
% Specify directories for retrieving data and saving results
datDir = uigetdir('C:\','Select a directory to retrieve data:');
savDir = uigetdir(datDir,'Select a directory to save results:');

% Specify number of sets of data files
dataNum = 50;

% Preallocate cell arrays for storing data files
dispData = cell(dataNum,1);

% Open and import data files for displacement measurements (delta u & delta v)
for count = 1:1:dataNum
    fid=fopen(strcat(datDir,'\(',num2str(count),').dat'),'r');
    dispData{count} = textscan(fid,'%f %f %f %f',-1,'headerlines',3);
    fclose(fid);
end

fprintf('ImportDisp: Importing data files has finished.\n');

%% Setting up parameters and variables for displacement measurements (SPECIFY convFac and tStep)

% Define convertion factor between image space and object space
convFac = 11.9;              % um/pixel (2X magnification)

% Define variables for xy-coordinates
% Note: x and y have the same dimensions (m x m)
dispCoorNum = size(dispData{1}{1},1);   % number of xy-coordinates for u & v
xDispCoorNum = sqrt(dispCoorNum);       % number of x-coordinates for u & v
yDispCoorNum = sqrt(dispCoorNum);       % number of y-coordinates for u & v
xDisp = zeros(xDispCoorNum,1);          % x-coordinate for u & v
yDisp = zeros(yDispCoorNum,1);          % y-coordinate for u & v

% Difine time step, time variable, and input values into time variable
tStepNum = dataNum;             % number of time steps
tStep = 10;                     % time step in seconds
t = zeros(tStepNum,1);          % time

for k = 1:1:tStepNum
    t(k) = (k*tStep-tStep/2);
end

% Define variables for displacement measurements
delta_u = zeros(xDispCoorNum,yDispCoorNum,tStepNum);       % delta x-displacement
delta_v = zeros(xDispCoorNum,yDispCoorNum,tStepNum);       % delta y-displacement

fprintf('ImportDisp: Setting up parameters and variables has finished.\n');
```

```
%% Inputting data into variables

% Input displacement corrdinates and convert them into object length
for i = 1:1:xDispCoorNum
    xDisp(i) = dispData{1}{1}(i)*convFac;
end

for j = 1:1:yDispCoorNum
    yDisp(j) = dispData{1}{2}(1+(j-1)*xDispCoorNum)*convFac;
end

% Input displacement measurements (u & v) and convert them into object length
for k = 1:1:tStepNum
    count = 1;

    for j = 1:1:yDispCoorNum
        for i = 1:1:xDispCoorNum
            delta_u(i,j,k) = dispData{k}{3}(count)*convFac;
            delta_v(i,j,k) = dispData{k}{4}(count)*convFac;
            count = count+1;
        end
    end
end

clear dispData

fprintf('ImportDisp: Inputting data has finished.\n');
```

```
% YAvgDelDisp.m
% Ka Yaw Teo
% Last updated: 7/22/2008

% DESCRIPTION: -
% This file consists of two sections of codes (variable set-up and
% displacement averaging). A message will be displayed when each section finishes running.
% The resulting variables are:

% delta_u_avg(i,k) where i = 1:1:xDispCoorNum, and k = 1:1:tStepNum
% delta_v_avg(i,k)
% std_delta_u_avg(i,k)
% std_delta_v_avg(i,k)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.
```

```
%% Define variables for y-averaged delta u and delta v

delta_u_avg = zeros(xDispCoorNum,tStepNum);
delta_v_avg = zeros(xDispCoorNum,tStepNum);
std_delta_u_avg = zeros(xDispCoorNum,tStepNum);
std_delta_v_avg = zeros(xDispCoorNum,tStepNum);

fprintf('YAvgDelDisp: Defining variables has finished.\n');

%% Compute y-averaged delta u and delta v

for k = 1:1:tStepNum
    for i = 1:1:xDispCoorNum
        delta_u_nonZero = nonzeros(delta_u(i,:,k));
        delta_v_nonZero = nonzeros(delta_v(i,:,k));

        delta_u_avg(i,k) = mean2(delta_u_nonZero);
        delta_v_avg(i,k) = mean2(delta_v_nonZero);
        std_delta_u_avg(i,k) = std(delta_u_nonZero);
        std_delta_v_avg(i,k) = std(delta_v_nonZero);
    end
end

fprintf('YAvgDelDisp: Computing y-averaged delta u and delta v has finished.\n');
```

```
% InterLoc.m
% Ka Yaw Teo
% Last updated: 7/22/2008

% DESCRIPTION: -
% This file consists of four sections of codes (variable set-up, determining
% interface location, determining time-zero and average interface velocity,
% and plotting interface location with time-zero and average interface
% velocity). A message will be displayed when each section finishes running.

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables for determining interface location

max_delta_u_avg_mag = zeros(tStepNum,1);
max_delta_u_avg_xCoor = zeros(tStepNum,1);
interLoc = zeros(tStepNum,1);

fprintf('InterLoc: Defining variables has finished.\n');
```

```
%% Determine interface location

for k = 1:1:tStepNum
    [max_delta_u_avg_mag(k),max_delta_u_avg_xCoor(k)] = max(delta_u_avg(:,k));
    interLoc(k) = xDisp(max_delta_u_avg_xCoor(k));
end

fprintf('InterLoc: Determining interface location has finished.\n');

%% Determine time-zero and average interface velocity

interLocTemp = zeros(5,1);
tTemp = zeros(5,1);

count = 1;
for k = 1:1:5                          % Designate a range (5 points) to be fit
    interLocTemp(count) = interLoc(k);
    tTemp(count) = t(k);
    count = count+1;
end

% Linear regression
x = [ones(size(tTemp,1),1) tTemp];
y = interLocTemp;
[b,bint,r,rint,stats] = regress(y,x);

% Compute time-zero
tZero = -b(1)/b(2);

% Determine average interface velocity (X(tf)/tf-t0)
avgInterVel = interLoc(tStepNum)/(t(tStepNum)-tZero);

fprintf('InterLoc: Determining time-zero and average interface velocity has finished.\n');

%% Plot interface location, show time-zero and average interface velocity

fig = figure('Name','Interface location with time-zero and average interface velocity','NumberTitle','off');
set(fig,'Visible','off');
plot(t,t*b(2)+b(1),'r-');
hold on;
plot(t,interLoc,'b.','MarkerSize',15);
text(50,3000,texlabel(strcat('t_0 = ',num2str(tZero),' s')));
text(50,2400,{'Average interface velocity';strcat('= ',num2str(avgInterVel),' um/s')});
axis([0 max(t) 0 6000]);
axis square;
axis manual;
```

```
xlabel('Time, t [s]');
ylabel('Phase change interface location, X [um]');
title('Phase Change Interface Location Vs. Time');
saveas(fig,strcat(savDir,'\Interface location.png'));
close(fig);

fprintf('InterLoc: Plotting interface location with time-zero and average interface velocity has finished.\n');
```

```
% CalcDelStrnDltn.m
% Ka Yaw Teo
% Last updated: 12/16/2008

% DESCRIPTION: -
% This file consists of four sections of codes (variable set-up,
% dilatation computation, plotting contour, and printing results).
% A message will be displayed when each section finishes running.
% The resulting variable is:

% Cauchy's infinitesimal strains/dilation:
% delta_Exx(i,j,k) where i = 1:1:xDispCoorNum, j = 1:1:yDispCoorNum, and k = 1:1:tStepNum
% delta_Eyy(i,j,k)
% delta_e(i,j,k)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables for calculated strains and dilatation

% Strains
delta_Exx = zeros(xDispCoorNum,yDispCoorNum,tStepNum);
delta_Eyy = zeros(xDispCoorNum,yDispCoorNum,tStepNum);

% Dilatation
delta_e = zeros(xDispCoorNum,yDispCoorNum,tStepNum);

fprintf('CalcStrnDltn: Defining variables has finished.\n');

%% Compute dilatation (SPECIFY SCHEME for difference approximation)

% Cauchy's infinitesimal strains (squares and products of the partial
% derivatives of displacement are negligible compared with the first order
% terms)

SCHEME = 'CD';    % UD, CD, LUD, QUD, ELUD, EQUD, or CUD
alpha = 1/3;      % DESIGNATE alpha = 1/3 or 1/6 for CUD
```

```
% NOTE: In the following codes, only CD scheme includes difference
% approximation for boundaries.  Codes for approximating spatial
% derivatives at boundaries in other schemes have yet to be added.

if strcmp(SCHEME,'UD')
    alpha = 0.5;
    beta = 0;
    gamma = 0;
    xFP = 2;
    yFP = 2;
    xLP = xDispCoorNum;
    yLP = yDispCoorNum;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*((2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-
2*beta+gamma)*delta_u(i-1,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*((2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-
2*beta+gamma)*delta_v(i,j-1,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

if strcmp(SCHEME,'CD')
    alpha = 0;
    beta = 0;
    gamma = 0;

    for k = 1:1:tStepNum
        for j = 1:1:yDispCoorNum
            for i = 2:1:xDispCoorNum-1
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*((0.5-alpha-gamma)*delta_u(i+1,j,k)+(-0.5-alpha-
2*beta+gamma)*delta_u(i-1,j,k));
            end
        end
    end

    for k = 1:1:tStepNum
        for j = 2:1:yDispCoorNum-1
            for i = 1:1:xDispCoorNum
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*((0.5-alpha-gamma)*delta_v(i,j+1,k)+(-0.5-alpha-
2*beta+gamma)*delta_v(i,j-1,k));
            end
```

```
        end
    end

    % 1-point downstream weighting approximation for left-bounded data points
    for k = 1:1:tStepNum
        for j = 1:1:yDispCoorNum
            delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*(delta_u(1,j,k)-delta_u(2,j,k));
        end
    end

    % 1-point upstream weighting approximation for right-bounded data points
    for k = 1:1:tStepNum
        for j = 1:1:yDispCoorNum
            delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*(delta_u(xDispCoorNum,j,k)-delta_u(xDispCoorNum-1,j,k));
        end
    end

    % 1-point downstream weighting approximation for lower-bounded data points
    for k = 1:1:tStepNum
        for i = 1:1:xDispCoorNum
            delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*(delta_v(i,1,k)-delta_v(i,2,k));
        end
    end

    % 1-point upstream weighting approximation for upper-bounded data points
    for k = 1:1:tStepNum
        for i = 1:1:xDispCoorNum
            delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*(delta_v(i,yDispCoorNum,k)-delta_v(i,yDispCoorNum-1,k));
        end
    end

    delta_e = delta_Exx+delta_Eyy;

end

if strcmp(SCHEME,'LUD')
    alpha = 0.5;
    beta = alpha;
    gamma = 0;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-1;
    yLP = yDispCoorNum-1;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
```

```
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*((2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-
2*beta+gamma)*delta_u(i-1,j,k)+(beta-0.5*gamma)*delta_u(i-2,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*((2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-
2*beta+gamma)*delta_v(i,j-1,k)+(beta-0.5*gamma)*delta_v(i,j-2,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

if strcmp(SCHEME,'QUD')
    alpha = 0.125;
    beta = alpha;
    gamma = 0;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-1;
    yLP = yDispCoorNum-1;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*((0.5-alpha-
gamma)*delta_u(i+1,j,k)+(2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_u(i-1,j,k)+(beta-
0.5*gamma)*delta_u(i-2,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*((0.5-alpha-
gamma)*delta_v(i,j+1,k)+(2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_v(i,j-1,k)+(beta-
0.5*gamma)*delta_v(i,j-2,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

if strcmp(SCHEME,'ELUD')
    alpha = 0.5;
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
```

```
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*u(i+2,j,k)+(0.5-alpha-
gamma)*delta_u(i+1,j,k)+(2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_u(i-1,j,k)+(beta-
0.5*gamma)*delta_u(i-2,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*v(i,j+2,k)+(0.5-alpha-
gamma)*delta_v(i,j+1,k)+(2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_v(i,j-1,k)+(beta-
0.5*gamma)*delta_v(i,j-2,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

if strcmp(SCHEME,'EQUD')
    alpha = 0.125;
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*delta_u(i+2,j,k)+(0.5-alpha-
gamma)*delta_u(i+1,j,k)+(2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_u(i-1,j,k)+(beta-
0.5*gamma)*delta_u(i-2,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*delta_v(i,j+2,k)+(0.5-alpha-
gamma)*delta_v(i,j+1,k)+(2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_v(i,j-1,k)+(beta-
0.5*gamma)*delta_v(i,j-2,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

if strcmp(SCHEME,'CUD')
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for k = 1:1:tStepNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
```

```
                delta_Exx(i,j,k) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*delta_u(i+2,j,k)+(0.5-alpha-
gamma)*delta_u(i+1,j,k)+(2*alpha+beta)*delta_u(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_u(i-1,j,k)+(beta-
0.5*gamma)*delta_u(i-2,j,k));
                delta_Eyy(i,j,k) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*delta_v(i,j+2,k)+(0.5-alpha-
gamma)*delta_v(i,j+1,k)+(2*alpha+beta)*delta_v(i,j,k)+(-0.5-alpha-2*beta+gamma)*delta_v(i,j-1,k)+(beta-
0.5*gamma)*delta_v(i,j-2,k));
                delta_e(i,j,k) = delta_Exx(i,j,k)+delta_Eyy(i,j,k);
            end
        end
    end
end

fprintf('CalcStrnDltn: Computing delta strains and dilatation has finished.\n');
```

```
% YAvgDelDltn.m
% Ka Yaw Teo
% Last updated: 12/25/2008

% DESCRIPTION: -
% This file consists of four sections of codes (variable set-up,
% dilatation averaging, plotting y-averaged dilatation, and printing results).
% A message will be displayed when each section finishes running.
% The resulting variables are:

% delta_e_avg(i,k) where i = 1:1:xDispCoorNum ,and k = 1:1:tStepNum
% std_delta_e_avg(i,k)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables for y-averaged dilatation

delta_e_avg = zeros(xDispCoorNum,tStepNum);
std_delta_e_avg = zeros(xDispCoorNum,tStepNum);

fprintf('YAvgDltn: Defining variables has finished.\n');

%% Compute y-averaged dilatation

for k = 1:1:tStepNum
    for i = 1:1:xDispCoorNum
        delta_e_avg(i,k) = mean2(delta_e(i,:,k));
        std_delta_e_avg(i,k) = std(delta_e(i,:,k));
    end
end
```

```
fprintf('YAvgDltn: Computing delta_e_avg has finished.\n');
```

```
% SpecInterLocDispDltn.m
% Ka Yaw Teo
% Last updated: 1/1/2009

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% interLoc(k) where k = 1:1:tStepNum
% specInterLocIndex(n) where n = 1:1:specInterLocNum

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Determine interface location

max_delta_u_avg_mag = zeros(tStepNum,1);
max_delta_u_avg_xCoor = zeros(tStepNum,1);
interLoc = zeros(tStepNum,1);

for k = 1:1:tStepNum
    [max_delta_u_avg_mag(k),max_delta_u_avg_xCoor(k)] = max(delta_u_avg(:,k));
    interLoc(k) = xDisp(max_delta_u_avg_xCoor(k));
end

fprintf('SpecInterLocDispDltn: Determining interface location has finished.\n');

%% Find specific interface locations and corresponding indices (SPECIFY specInterLocNum, specInterLocInc, and nCompDat)

specInterLocNum = 4;       % Number of specific interface location (starting at 1)
specInterLocInc = 1000;    % Increment between two consecutive specific interface location (a multiplication factor to
specInterLocNum)

polyFitOrder = 3;
p = polyfit(t,interLoc,polyFitOrder);
interLocFit = polyval(p,t);

fig = figure('Name','Interface location with polynomial curve fit','NumberTitle','off');
set(fig,'Visible','off');
plot(t,interLoc,'b.','MarkerSize',15);
```

```
hold on;
plot(t,interLocFit,'-');
axis([0 max(t) 0 6000]);
axis square;
axis manual;
xlabel('Time, t [s]');
ylabel('Interface location, X [um]');
title('Interface location with polynomial curve fit');
saveas(fig,strcat(savDir,'\Interface location with polynomial curve fit.png'));
close(fig);

interLocFitEqn = '';

for n = 1:1:(polyFitOrder+1)
    interLocFitEqn = strcat(interLocFitEqn,'+(',num2str(p(n)),')*t^',num2str(polyFitOrder-(n-1)));
end

specInterLocFitSol = zeros(polyFitOrder,specInterLocNum);
specInterLocFitTime = zeros(specInterLocNum,1);

for n = 1:1:specInterLocNum
    specInterLocFitSol(:,n) = solve([interLocFitEqn,'=',num2str(n*specInterLocInc)],'t');
end

for n = 1:1:specInterLocNum
    for m = 1:1:polyFitOrder
        if (specInterLocFitTime(n) == 0) && (imag(specInterLocFitSol(m,n)) == 0) && (real(specInterLocFitSol(m,n) <=
max(tStepNum*tStep))) && (real(specInterLocFitSol(m,n) > 0))
            specInterLocFitTime(n) = specInterLocFitSol(m,n);
        end
    end
end

specInterLocFitTimeIndex = zeros(specInterLocNum,1);

for n = 1:1:specInterLocNum
    for k = 1:1:tStepNum
        if (specInterLocFitTime(n) == t(k))
            specInterLocFitTimeIndex(n) = k;
        end

        if (k ~= tStepNum)
            if (specInterLocFitTime(n) ~= t(1)) && (specInterLocFitTime(n) ~= t(tStepNum))
                if (specInterLocFitTime(n) > t(k)) && (specInterLocFitTime(n) < t(k+1))
                    if (t(k+1)-specInterLocFitTime(n)) > (specInterLocFitTime(n)-t(k))
                        specInterLocFitTimeIndex(n) = k;
                    else
```

```
                        specInterLocFitTimeIndex(n) = k+1;
                    end
                end
            end
        end

        if (specInterLocFitTime(n) < t(1))
            specInterLocFitTimeIndex(n) = 1;
        end

        if (specInterLocFitTime(n) > t(tStepNum))
            specInterLocFitTimeIndex(n) = tStepNum;
        end
    end
end

specInterLocIndex = zeros(specInterLocNum,1);
nCompDat = 4;   % number of adjacent data points (one side) to be considered in comparison

for n = 1:1:specInterLocNum
    dif = 0;

    if (specInterLocFitTimeIndex(n) >= (nCompDat+1)) && (specInterLocFitTimeIndex(n) <= (size(interLoc,1)-nCompDat))
        dif = abs(interLoc(specInterLocFitTimeIndex(n))-(n*specInterLocInc));
        specInterLocIndex(n) = specInterLocFitTimeIndex(n);

        for m = 1:1:nCompDat
            if (abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc)) < dif)
                dif = abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc));
                specInterLocIndex(n) = specInterLocFitTimeIndex(n)-m;
            end

            if (abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc)) < dif)
                dif = abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc));
                specInterLocIndex(n) = specInterLocFitTimeIndex(n)+m;
            end
        end
    end

    if (specInterLocFitTimeIndex(n) < (nCompDat+1)) && (specInterLocFitTimeIndex(n) ~= 1)
        dif = abs(interLoc(specInterLocFitTimeIndex(n))-(n*specInterLocInc));
        specInterLocIndex(n) = specInterLocFitTimeIndex(n);

        for m = 1:1:specInterLocFitTimeIndex(n)-1
            if (abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc)) < dif)
                dif = abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc));
                specInterLocIndex(n) = specInterLocFitTimeIndex(n)-m;
```

```
        end
    end

    for m = 1:1:nCompDat
        if (abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc)) < dif)
            dif = abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc));
            specInterLocIndex(n) = specInterLocFitTimeIndex(n)+m;
        end
    end
end

if (specInterLocFitTimeIndex(n) > (size(interLoc,1)-nCompDat)) && (specInterLocFitTimeIndex(n) ~= size(interLoc,1))
    dif = abs(interLoc(specInterLocFitTimeIndex(n))-(n*specInterLocInc));
    specInterLocIndex(n) = specInterLocFitTimeIndex(n);

    for m = 1:1:nCompDat
        if (abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc)) < dif)
            dif = abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc));
            specInterLocIndex(n) = specInterLocFitTimeIndex(n)-m;
        end
    end

    for m = 1:1:(size(interLoc,1)-specInterLocFitTimeIndex(n))
        if (abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc)) < dif)
            dif = abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc));
            specInterLocIndex(n) = specInterLocFitTimeIndex(n)+m;
        end
    end
end

if (specInterLocFitTimeIndex(n) == 1)
    dif = abs(interLoc(specInterLocFitTimeIndex(n))-(n*specInterLocInc));
    specInterLocIndex(n) = specInterLocFitTimeIndex(n);

    for m = 1:1:nCompDat
        if (abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc)) < dif)
            dif = abs(interLoc(specInterLocFitTimeIndex(n)+m)-(n*specInterLocInc));
            specInterLocIndex(n) = specInterLocFitTimeIndex(n)+m;
        end
    end
end

if (specInterLocFitTimeIndex(n) == size(interLoc,1))
    dif = abs(interLoc(specInterLocFitTimeIndex(n))-(n*specInterLocInc));
    specInterLocIndex(n) = specInterLocFitTimeIndex(n);

    for m = 1:1:nCompDat
```

```
            if (abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc)) < dif)
                dif = abs(interLoc(specInterLocFitTimeIndex(n)-m)-(n*specInterLocInc));
                specInterLocIndex(n) = specInterLocFitTimeIndex(n)-m;
            end
        end
    end

end

fprintf('SpecInterLocDispDltn: Finding specific interface locations has finished.\n');

%% Plot incremental displacement (SPECIFY scaleFac)

resDir = strcat(savDir,'\Interface-location-specific incremental displacement\');
mkdir(resDir);

scaleFac = 8;   % Vector scale factor

for k = 1:1:specInterLocNum
    fig = figure('Name',['delta u and delta v at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    [x,y] = meshgrid(xDisp,yDisp);
    h = quiver(x,y,scaleFac*delta_u(:,:,specInterLocIndex(k)).',scaleFac*delta_v(:,:,specInterLocIndex(k)).');
    set(h,'AutoScale','off');
    axis([0 6000 0 6000])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('y [um]');
    title(['delta u and delta v at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s) (vector scale factor = ',num2str(scaleFac),')']);
    saveas(fig,[resDir,'delta_u_delta_v_X',num2str(k),'.png']);
    close(fig);
end

for k = 1:1:specInterLocNum
    fig = figure('Name',['delta u avg at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    errorbar(xDisp,delta_u_avg(:,specInterLocIndex(k)),std_delta_u_avg(:,specInterLocIndex(k)));
    axis([0 6000 0 100])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('delta u avg [um]');
```

```
    title(['delta u avg at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)']);
    saveas(fig,[resDir,'delta_u_avg_X',num2str(k),'.png']);
    close(fig);
end

fprintf('SpecInterLocDispDltn: Plotting incremental displacements at specific interface locations has finished.\n');

%% Plot incremental dilatation

resDir = strcat(savDir,'\Interface-location-specific incremental dilatation\');
mkdir(resDir);

for k = 1:1:size(specInterLocIndex,1)
    fig = figure('Name',['delta e at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    h = surf(xDisp,yDisp,delta_e(:,:,specInterLocIndex(k)).');
    set(h,'LineStyle','none');
    view(0,90);
    xlabel('x [um]');
    ylabel('y [um]');
    zlabel('delta e [dimensionless]');
    title(['delta e at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)']);
    axis([0 6000 0 6000 -0.1 0.1 -0.1 0.1]);
    axis square;
    axis manual;
    colorbar;
    caxis([-0.1 0.1]);
    caxis manual;
    saveas(fig,[resDir,'delta_e_X',num2str(k),'.png']);
    close(fig);
end

for k = 1:1:size(specInterLocIndex,1)
    fig = figure('Name',['delta e avg at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    errorbar(xDisp,delta_e_avg(:,specInterLocIndex(k)),std_delta_e_avg(:,specInterLocIndex(k)));
    axis([0 6000 -0.1 0.1])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('delta e avg [dimensionless]');
    title(['delta e avg at X(t) = ',num2str(interLoc(specInterLocIndex(k)),5),' um (t =
',num2str(t(specInterLocIndex(k))),' s)']);
```

```
    saveas(fig,[resDir,'delta_e_avg_X',num2str(k),'.png']);
    close(fig);
end

fprintf('SpecInterLocDispDltn: Plotting incremental dilatations at specific interface locations has finished.\n');

%% Print incremental displacements

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_u_delta_v_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_u_delta_v_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "y", "delta_u", "delta_v"\n');
    fprintf(fid,'ZONE I=32, J=32, F=POINT\n');

    for i = 1:1:xDispCoorNum
        for j = 1:1:yDispCoorNum
            fprintf(fid,'%7.2e %7.2e %7.2e
%7.2e\n',xDisp(i),yDisp(j),delta_u(i,j,specInterLocIndex(n)),delta_v(i,j,specInterLocIndex(n)));
        end
    end

    fclose(fid);
end

fprintf('SpecInterLocDispDltn: Printing incremental displacements at specific interface locations has finished.\n');
```

```
% ImportDelDisp_MultiSets.m
% Ka Yaw Teo
% Last updated: 7/22/2008

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% delta_u(i,j,k,s) where i = 1:1:xDispCoorNum, j = 1:1:yDispCoorNum, k = 1:1:tStepNum, and s = 1:1:setNum
% delta_v(i,j,k,s)
% xDispCoorNum
% yDispCoorNum
% xDisp(i)
% yDisp(j)
% tStep
% tStepNum
% t(k)
% setNum
```

```
% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Importing data files (SPECIFY dataNum)

% Specify number of data (time points)
dataNum = 50;

% Specify number of sets of data (samples)
prompt = {'Enter number of sets of data to be processed:'};
dlg_title = 'Number of data sets';
num_lines = 1;
input = inputdlg(prompt,dlg_title,num_lines);
setNum = str2double(input{1});
datDir = cell(setNum,1);

% Specify directories for retrieving data and saving results
for sNum = 1:1:setNum
    datDir{sNum} = uigetdir('C:\',strcat('Select a directory to retrieve data set #',num2str(sNum),':'));
end

savDir = uigetdir(datDir{1},'Select a directory to save results:');

% Preallocate cell arrays for storing data
dispData = cell(dataNum,setNum);

% Open and import data for displacement measurements (delta u & delta v)
for sNum = 1:1:setNum
    for dNum = 1:1:dataNum
        fid=fopen(strcat(datDir{sNum},'\D(',num2str(dNum),').dat'),'r');
        dispData{dNum,sNum} = textscan(fid,'%f %f %f %f',-1,'headerlines',3);
        fclose(fid);
    end
end

fprintf('ImportDelDisp_MultiSets: Importing data files has finished.\n');

%% Setting up parameters and variables for displacement measurements (SPECIFY convFac and tStep)

% Define conversion factor between image space and object space
convFac = 11.9;              % um/pixel (2X magnification)

% Define variables for xy-coordinates
% Note: x and y have the same dimensions (m x m)
dispCoorNum = size(dispData{1}{1},1);   % number of xy-coordinates for u & v
xDispCoorNum = sqrt(dispCoorNum);       % number of x-coordinates for u & v
```

```
yDispCoorNum = sqrt(dispCoorNum);       % number of y-coordinates for u & v
xDisp = zeros(xDispCoorNum,1);          % x-coordinate for u & v
yDisp = zeros(yDispCoorNum,1);          % y-coordinate for u & v

% Difine time step, time variable, and input values into time variable
tStepNum = dataNum;             % number of time steps
tStep = 10;                     % time step in seconds
t = zeros(tStepNum,1);          % time

for k = 1:1:tStepNum
    t(k) = (k*tStep-tStep/2);
end

% Define variables for displacement measurements
delta_u = zeros(xDispCoorNum,yDispCoorNum,tStepNum,setNum);       % delta x-displacement
delta_v = zeros(xDispCoorNum,yDispCoorNum,tStepNum,setNum);       % delta y-displacement

fprintf('ImportDelDisp_MultiSets: Setting up parameters and variables has finished.\n');

%% Inputting data into variables

% Input displacement corrdinates and convert them into object length
for i = 1:1:xDispCoorNum
    xDisp(i) = dispData{1}{1}(i)*convFac;
end

for j = 1:1:yDispCoorNum
    yDisp(j) = dispData{1}{2}(1+(j-1)*xDispCoorNum)*convFac;
end

% Input displacement measurements (u & v) and convert them into object length
for s = 1:1:setNum
    for k = 1:1:tStepNum
        count = 1;

        for j = 1:1:yDispCoorNum
            for i = 1:1:xDispCoorNum
                delta_u(i,j,k,s) = dispData{k,s}{3}(count)*convFac;
                delta_v(i,j,k,s) = dispData{k,s}{4}(count)*convFac;
                count = count+1;
            end
        end
    end
end

clear dispData
```

```
fprintf('ImportDelDisp_MultiSets: Inputting data has finished.\n');
```

```
% YAvgDelDisp_MultiSets.m
% Ka Yaw Teo
% Last updated: 10/17/2008

% DESCRIPTION: -
% This file consists of two sections of codes (variable set-up and
% displacement averaging). A message will be displayed when each section finishes running.
% The resulting variables are:

% delta_u_avg(i,k,s) where i = 1:1:xDispCoorNum, k = 1:1:tStepNum, and s = 1:1:setNum
% delta_v_avg(i,k,s)
% std_delta_u_avg(i,k,s)
% std_delta_v_avg(i,k,s)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables for y-averaged delta u and delta v

delta_u_avg = zeros(xDispCoorNum,tStepNum,setNum);
delta_v_avg = zeros(xDispCoorNum,tStepNum,setNum);
std_delta_u_avg = zeros(xDispCoorNum,tStepNum,setNum);
std_delta_v_avg = zeros(xDispCoorNum,tStepNum,setNum);

fprintf('YAvgDelDisp_MultiSets: Defining variables has finished.\n');

%% Compute y-averaged delta u and delta v

for s = 1:1:setNum
    for k = 1:1:tStepNum
        for i = 1:1:xDispCoorNum
            delta_u_nonZero = nonzeros(delta_u(i,:,k,s));
            delta_v_nonZero = nonzeros(delta_v(i,:,k,s));

            delta_u_avg(i,k,s) = mean2(delta_u_nonZero);
            delta_v_avg(i,k,s) = mean2(delta_v_nonZero);
            std_delta_u_avg(i,k,s) = std(delta_u_nonZero);
            std_delta_v_avg(i,k,s) = std(delta_v_nonZero);
        end
    end
end

fprintf('YAvgDelDisp_MultiSets: Computing y-averaged delta u and delta v has finished.\n');
```

```
% SpecInterLoc_MultiSets.m
% Ka Yaw Teo
% Last updated: 1/2/2009

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% interLoc(k,s) where k = 1:1:tStepNum and s = 1:1:setNum
% specInterLocIndex(n,s) where n = 1:1:specInterLocNum and s = 1:1:setNum

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Determine interface location

max_delta_u_avg_mag = zeros(tStepNum,setNum);
max_delta_u_avg_xCoor = zeros(tStepNum,setNum);
interLoc = zeros(tStepNum,setNum);

for s = 1:1:setNum
    for k = 1:1:tStepNum
        [max_delta_u_avg_mag(k,s),max_delta_u_avg_xCoor(k,s)] = max(delta_u_avg(:,k,s));
        interLoc(k,s) = xDisp(max_delta_u_avg_xCoor(k,s));
    end
end

fprintf('SpecInterLoc_MultiSets: Determining interface location has finished.\n');

%% Find specific interface locations and corresponding indices (SPECIFY specInterLocNum, specInterLocInc, and nCompDat)

specInterLocNum = 4;       % Number of specific interface location (starting at 1)
specInterLocInc = 1000;    % Increment between two consecutive specific interface location (a multiplication factor to
specInterLocNum)

specInterLocIndex = zeros(specInterLocNum,setNum);

for s = 1:1:setNum
    polyFitOrder = 3;
    p = polyfit(t,interLoc(:,s),polyFitOrder);
    interLocFit = polyval(p,t);
    interLocFitEqn = '';
```

```
    for n = 1:1:(polyFitOrder+1)
        interLocFitEqn = strcat(interLocFitEqn,'+(',num2str(p(n)),')*t^',num2str(polyFitOrder-(n-1)));
    end

    specInterLocFitSol = zeros(polyFitOrder,specInterLocNum);
    specInterLocFitTime = zeros(specInterLocNum,1);

    for n = 1:1:specInterLocNum
        specInterLocFitSol(:,n) = solve([interLocFitEqn,'=',num2str(n*specInterLocInc)],'t');
    end

    for n = 1:1:specInterLocNum
        for m = 1:1:polyFitOrder
            if (specInterLocFitTime(n) == 0) && (imag(specInterLocFitSol(m,n)) == 0) && (real(specInterLocFitSol(m,n)
<= max(tStepNum*tStep))) && (real(specInterLocFitSol(m,n) > 0))
                specInterLocFitTime(n) = specInterLocFitSol(m,n);
            end
        end
    end

    specInterLocFitTimeIndex = zeros(specInterLocNum,1);

    for n = 1:1:specInterLocNum
        for k = 1:1:tStepNum
            if (specInterLocFitTime(n) == t(k))
                specInterLocFitTimeIndex(n) = k;
            end

            if (k ~= tStepNum)
                if (specInterLocFitTime(n) ~= t(1)) && (specInterLocFitTime(n) ~= t(tStepNum))
                    if (specInterLocFitTime(n) > t(k)) && (specInterLocFitTime(n) < t(k+1))
                        if (t(k+1)-specInterLocFitTime(n)) > (specInterLocFitTime(n)-t(k))
                            specInterLocFitTimeIndex(n) = k;
                        else
                            specInterLocFitTimeIndex(n) = k+1;
                        end
                    end
                end
            end

            if (specInterLocFitTime(n) < t(1))
                specInterLocFitTimeIndex(n) = 1;
            end

            if (specInterLocFitTime(n) > t(tStepNum))
                specInterLocFitTimeIndex(n) = tStepNum;
            end
```

```
        end
    end

    nCompDat = 4;   % Number of adjacent data points (one side) to be considered in comparison

    for n = 1:1:specInterLocNum
        dif = 0;

        if (specInterLocFitTimeIndex(n) >= (nCompDat+1)) && (specInterLocFitTimeIndex(n) <= (size(interLoc,1)-
nCompDat))
            dif = abs(interLoc(specInterLocFitTimeIndex(n),s)-(n*specInterLocInc));
            specInterLocIndex(n,s) = specInterLocFitTimeIndex(n);

            for m = 1:1:nCompDat
                if (abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)-m;
                end

                if (abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)+m;
                end
            end
        end

        if (specInterLocFitTimeIndex(n) < (nCompDat+1)) && (specInterLocFitTimeIndex(n) ~= 1)
            dif = abs(interLoc(specInterLocFitTimeIndex(n),s)-(n*specInterLocInc));
            specInterLocIndex(n,s) = specInterLocFitTimeIndex(n);

            for m = 1:1:specInterLocFitTimeIndex(n)-1
                if (abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)-m;
                end
            end

            for m = 1:1:nCompDat
                if (abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)+m;
                end
            end
        end

        if (specInterLocFitTimeIndex(n) > (size(interLoc,1)-nCompDat)) && (specInterLocFitTimeIndex(n) ~=
size(interLoc,1))
```

```
            dif = abs(interLoc(specInterLocFitTimeIndex(n),s)-(n*specInterLocInc));
            specInterLocIndex(n,s) = specInterLocFitTimeIndex(n);

            for m = 1:1:nCompDat
                if (abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)-m;
                end
            end

            for m = 1:1:(size(interLoc,1)-specInterLocFitTimeIndex(n))
                if (abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)+m;
                end
            end
        end

        if (specInterLocFitTimeIndex(n) == 1)
            dif = abs(interLoc(specInterLocFitTimeIndex(n),s)-(n*specInterLocInc));
            specInterLocIndex(n,s) = specInterLocFitTimeIndex(n);

            for m = 1:1:nCompDat
                if (abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)+m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)+m;
                end
            end
        end

        if (specInterLocFitTimeIndex(n) == size(interLoc,1))
            dif = abs(interLoc(specInterLocFitTimeIndex(n),s)-(n*specInterLocInc));
            specInterLocIndex(n,s) = specInterLocFitTimeIndex(n);

            for m = 1:1:nCompDat
                if (abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc)) < dif)
                    dif = abs(interLoc(specInterLocFitTimeIndex(n)-m,s)-(n*specInterLocInc));
                    specInterLocIndex(n,s) = specInterLocFitTimeIndex(n)-m;
                end
            end
        end
    end
end

fprintf('SpecInterLoc_MultiSets: Finding specific interface locations has finished.\n');
```

```
% SampAvgSpecInterLocDelDisp_MultiSets.m
% Ka Yaw Teo
% Last updated: 1/3/2009

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% delta_u_spec_sAvg(i,j,n) where i = 1:1:xDispCoorNum, j = 1:1:yDispCoorNum, and n = 1:1:specInterLocNum
% delta_v_spec_sAvg(i,j,n)
% std_delta_u_spec_sAvg(i,j,n)
% std_delta_v_spec_sAvg(i,j,n)
% t_spec_sAvg(n)
% std_t_spec_sAvg(n)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Define variables

delta_u_spec = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum,setNum);
delta_v_spec = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum,setNum);
delta_u_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);
delta_v_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);
std_delta_u_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);
std_delta_v_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);

interLoc_spec = zeros(specInterLocNum,setNum);
interLoc_spec_sAvg = zeros(specInterLocNum,1);
std_interLoc_spec_sAvg = zeros(specInterLocNum,1);

t_spec = zeros(specInterLocNum,setNum);
t_spec_sAvg = zeros(specInterLocNum,1);
std_t_spec_sAvg = zeros(specInterLocNum,1);

fprintf('SampAvgSpecInterLocDelDisp_MultiSets: Defining variables has finished.\n');

%% Average incremental displacement and time at specific interface locations from multiple sets of data

for s = 1:1:setNum
    for n = 1:1:specInterLocNum
        delta_u_spec(:,:,n,s) = delta_u(:,:,specInterLocIndex(n,s),s);
        delta_v_spec(:,:,n,s) = delta_v(:,:,specInterLocIndex(n,s),s);
    end
end
```

```
for n = 1:1:specInterLocNum
    for j = 1:1:yDispCoorNum
        for i = 1:1:xDispCoorNum
            delta_u_spec_sAvg(i,j,n) = mean2(delta_u_spec(i,j,n,:));
            delta_v_spec_sAvg(i,j,n) = mean2(delta_v_spec(i,j,n,:));
            std_delta_u_spec_sAvg(i,j,n) = std2(delta_u_spec(i,j,n,:));
            std_delta_v_spec_sAvg(i,j,n) = std2(delta_v_spec(i,j,n,:));
        end
    end
end

for s = 1:1:setNum
    for n = 1:1:specInterLocNum
        interLoc_spec(n,s) = interLoc(specInterLocIndex(n,s));
    end
end

for n = 1:1:specInterLocNum
    interLoc_spec_sAvg(n) = mean2(interLoc_spec(n,:));
    std_interLoc_spec_sAvg(n) = std2(interLoc_spec(n,:));
end

for s = 1:1:setNum
    for n = 1:1:specInterLocNum
        t_spec(n,s) = t(specInterLocIndex(n,s));
    end
end

for n = 1:1:specInterLocNum
    t_spec_sAvg(n) = mean2(t_spec(n,:));
    std_t_spec_sAvg(n) = std2(t_spec(n,:));
end

fprintf('SampAvgSpecInterLocDelDisp_MultiSets: Computing delta_u_spec_sAvg, delta_v_spec_sAvg, interLoc_spec_sAvg, and
t_spec_sAvg has finished.\n');

%% Plot incremental displacements (SPECIFY scaleFac)

resDir = strcat(savDir,'\Interface-location-specific incremental displacement\');
mkdir(resDir);

scaleFac = 8;   % Vector scale factor

for n = 1:1:specInterLocNum
```

```
    fig = figure('Name',['delta u sAvg and delta v sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    [x,y] = meshgrid(xDisp,yDisp);
    h = quiver(x,y,scaleFac*delta_u_spec_sAvg(:,:,n).',scaleFac*delta_v_spec_sAvg(:,:,n).');
    set(h,'AutoScale','off');
    axis([0 6000 0 6000])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('y [um]');
    title(['delta u sAvg and delta v sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)',sprintf('\n'),'(vector scale factor = ',num2str(scaleFac),')']);
    saveas(fig,[resDir,'delta_u_spec_sAvg_delta_v_spec_sAvg_X',num2str(n),'.png']);
    close(fig);
end

for n = 1:1:specInterLocNum
    fig = figure('Name',['std of delta u sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    h = surf(xDisp,yDisp,std_delta_u_spec_sAvg(:,:,n).');
    set(h,'LineStyle','none');
    view(0,90);
    xlabel('x [um]');
    ylabel('y [um]');
    zlabel('std of delta u sAvg [um]');
    title(['std of delta u sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    axis([0 6000 0 6000 0 25 0 25]);
    axis square;
    axis manual;
    colorbar;
    caxis([0 25]);
    caxis manual;
    saveas(fig,[resDir,'std_delta_u_spec_sAvg_X',num2str(n),'.png']);
    close(fig);
end

for n = 1:1:specInterLocNum
    fig = figure('Name',['std of delta v sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
```

```
    set(fig,'Visible','off');
    h = surf(xDisp,yDisp,std_delta_v_spec_sAvg(:,:,n).');
    set(h,'LineStyle','none');
    view(0,90);
    xlabel('x [um]');
    ylabel('y [um]');
    zlabel('std of delta v sAvg [um]');
    title(['std of delta v sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    axis([0 6000 0 6000 0 25 0 25]);
    axis square;
    axis manual;
    colorbar;
    caxis([0 25]);
    caxis manual;
    saveas(fig,[resDir,'std_delta_v_spec_sAvg_X',num2str(n),'.png']);
    close(fig);
end

fprintf('SampAvgSpecInterLocDelDisp_MultiSets: Plotting incremental displacements at specific interface locations has
finished.\n');

%% Print incremental displacements

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_u_spec_sAvg_delta_v_spec_sAvg_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_u_spec_sAvg_delta_v_spec_sAvg_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "y", "delta_u_spec_sAvg", "delta_v_spec_sAvg"\n');
    fprintf(fid,'ZONE I=32, J=32, F=POINT\n');

    for i = 1:1:xDispCoorNum
        for j = 1:1:yDispCoorNum
            fprintf(fid,'%7.2e %7.2e %7.2e
%7.2e\n',xDisp(i),yDisp(j),delta_u_spec_sAvg(i,j,n),delta_v_spec_sAvg(i,j,n));
        end
    end

    fclose(fid);
end

fprintf('SampAvgSpecInterLocDelDisp_MultiSets: Printing incremental displacements at specific interface locations has
finished.\n');
```

```
% YAvgSampAvgSpecInterLocDelDisp_MultiSets.m
% Ka Yaw Teo
% Last updated: 1/3/2009

% DESCRIPTION: -
% This file consists of three sections of codes (data import, variable
% set-up, and data input).  A message will be displayed when each section
% finishes running.  The resulting variables are:

% delta_u_spec_sAvg_yAvg(i,n) where i = 1:1:xDispCoorNum and n = 1:1:specInterLocNum
% delta_v_spec_sAvg_yAvg(i,n)
% std_delta_u_spec_sAvg_yAvg(i,n)
% std_delta_v_spec_sAvg_yAvg(i,n)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Define variables

delta_u_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);
delta_v_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);
std_delta_u_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);
std_delta_v_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);

fprintf('YAvgSampAvgSpecInterLocDelDisp_MultiSets: Defining variables has finished.\n');

%% Average incremental displacement and time at specific interface locations from multiple sets of data

for n = 1:1:specInterLocNum
    for i = 1:1:xDispCoorNum
        delta_u_spec_sAvg_yAvg(i,n) = mean2(delta_u_spec_sAvg(i,:,n));
        delta_v_spec_sAvg_yAvg(i,n) = mean2(delta_v_spec_sAvg(i,:,n));
        std_delta_u_spec_sAvg_yAvg(i,n) = std2(delta_u_spec_sAvg(i,:,n));
        std_delta_v_spec_sAvg_yAvg(i,n) = std2(delta_v_spec_sAvg(i,:,n));
    end
end

fprintf('YAvgSampAvgSpecInterLocDelDisp_MultiSets: Computing delta_u_spec_sAvg_yAvg and delta_v_spec_sAvg_yAvg has
finished.\n');

%% Plot y-averaged incremental displacements

for n = 1:1:specInterLocNum
    fig = figure('Name',['delta u sAvg yAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
```

```
    set(fig,'Visible','off');
    errorbar(xDisp,delta_u_spec_sAvg_yAvg(:,n),std_delta_u_spec_sAvg_yAvg(:,n));
    axis([0 6000 0 100])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('delta u sAvg yAvg [um]');
    title(['delta u sAvg yAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    saveas(fig,[resDir,'delta_u_spec_sAvg_yAvg_X',num2str(n),'.png']);
    close(fig);
end

fprintf('YAvgSampAvgSpecInterLocDelDisp_MultiSets: Plotting y-averaged incremental displacements at specific interface
locations has finished.\n');

%% Print y-averaged incremental displacements

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_u_spec_sAvg_yAvg_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_u_spec_sAvg_yAvg_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "delta_u_spec_sAvg_yAvg", "std_delta_u_spec_sAvg_yAvg"\n');
    fprintf(fid,'ZONE I=32, F=POINT\n');

    for i = 1:1:xDispCoorNum
        fprintf(fid,'%7.2e %7.2e %7.2e\n',xDisp(i),delta_u_spec_sAvg_yAvg(i,n),std_delta_u_spec_sAvg_yAvg(i,n));
    end

    fclose(fid);
end

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_v_spec_sAvg_yAvg_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_v_spec_sAvg_yAvg_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "delta_v_spec_sAvg_yAvg", "std_delta_v_spec_sAvg_yAvg"\n');
    fprintf(fid,'ZONE I=32, F=POINT\n');

    for i = 1:1:xDispCoorNum
        fprintf(fid,'%7.2e %7.2e %7.2e\n',xDisp(i),delta_v_spec_sAvg_yAvg(i,n),std_delta_v_spec_sAvg_yAvg(i,n));
    end

    fclose(fid);
end

fprintf('YAvgSampAvgSpecInterLocDelDisp_MultiSets: Printing y-averaged incremental displacements at specific interface
locations has finished.\n');
```

```
% CalSampAvgSpecInterLocDelStrnDltn_MultiSets.m
% Ka Yaw Teo
% Last updated: 1/3/2009

% DESCRIPTION: -
% This file consists of four sections of codes (variable set-up,
% dilatation computation, plotting contour, and printing results).
% A message will be displayed when each section finishes running.
% The resulting variable is:

% delta_Exx_spec_sAvg(i,j,n) where i = 1:1:xDispCoorNum, j = 1:1:yDispCoorNum, and n = 1:1:specInterLocNum
% delta_Eyy_spec_sAvg(i,j,n)
% delta_e_spec_sAvg(i,j,n)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables for calculated strains and dilatation

% Strains
delta_Exx_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);
delta_Eyy_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);

% Dilatation
delta_e_spec_sAvg = zeros(xDispCoorNum,yDispCoorNum,specInterLocNum);

fprintf('CalSampAvgSpecInterLocDelStrnDltn_MultiSets: Defining variables has finished.\n');

%% Compute dilatation (SPECIFY SCHEME for difference approximation)

% Cauchy's infinitesimal strains (squares and products of the partial
% derivatives of displacement are negligible compared with the first order
% terms)

SCHEME = 'CD';    % UD, CD, LUD, QUD, ELUD, EQUD, or CUD
alpha = 1/3;      % DESIGNATE alpha = 1/3 or 1/6 for CUD

% NOTE: In the following codes, only CD scheme includes difference
% approximation for boundaries.  Codes for approximating spatial
% derivatives at boundaries in other schemes have yet to be added.

if strcmp(SCHEME,'UD')
    alpha = 0.5;
    beta = 0;
    gamma = 0;
```

```
    xFP = 2;
    yFP = 2;
    xLP = xDispCoorNum;
    yLP = yDispCoorNum;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*((2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-
alpha-2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*((2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-
alpha-2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n));
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
            end
        end
    end
end

if strcmp(SCHEME,'CD')
    alpha = 0;
    beta = 0;
    gamma = 0;

    for n = 1:1:specInterLocNum
        for j = 1:1:yDispCoorNum
            for i = 2:1:xDispCoorNum-1
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*((0.5-alpha-gamma)*delta_u_spec_sAvg(i+1,j,n)+(-
0.5-alpha-2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n));
            end
        end
    end

    for n = 1:1:specInterLocNum
        for j = 2:1:yDispCoorNum-1
            for i = 1:1:xDispCoorNum
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*((0.5-alpha-gamma)*delta_v_spec_sAvg(i,j+1,n)+(-
0.5-alpha-2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n));
            end
        end
    end

    % 1-point downstream weighting approximation for left-bounded data points
    for n = 1:1:specInterLocNum
        for j = 1:1:yDispCoorNum
            delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*(delta_u_spec_sAvg(1,j,n)-delta_u_spec_sAvg(2,j,n));
        end
    end
```

```
    % 1-point upstream weighting approximation for right-bounded data points
    for n = 1:1:specInterLocNum
        for j = 1:1:yDispCoorNum
            delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*(delta_u_spec_sAvg(xDispCoorNum,j,n)-
delta_u_spec_sAvg(xDispCoorNum-1,j,n));
        end
    end

    % 1-point downstream weighting approximation for lower-bounded data points
    for n = 1:1:specInterLocNum
        for i = 1:1:xDispCoorNum
            delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*(delta_v_spec_sAvg(i,1,n)-delta_v_spec_sAvg(i,2,n));
        end
    end

    % 1-point upstream weighting approximation for upper-bounded data points
    for n = 1:1:specInterLocNum
        for i = 1:1:xDispCoorNum
            delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*(delta_v_spec_sAvg(i,yDispCoorNum,n)-
delta_v_spec_sAvg(i,yDispCoorNum-1,n));
        end
    end

    for n = 1:1:specInterLocNum
        delta_e_spec_sAvg(:,:,n) = delta_Exx_spec_sAvg(:,:,n)+delta_Eyy_spec_sAvg(:,:,n);
    end

end

if strcmp(SCHEME,'LUD')
    alpha = 0.5;
    beta = alpha;
    gamma = 0;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-1;
    yLP = yDispCoorNum-1;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*((2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-
alpha-2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n)+(beta-0.5*gamma)*delta_u_spec_sAvg(i-2,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*((2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-
alpha-2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n)+(beta-0.5*gamma)*delta_v_spec_sAvg(i,j-2,n));
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
```

```
            end
        end
    end
end

if strcmp(SCHEME,'QUD')
    alpha = 0.125;
    beta = alpha;
    gamma = 0;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-1;
    yLP = yDispCoorNum-1;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*((0.5-alpha-
gamma)*delta_u_spec_sAvg(i+1,j,n)+(2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n)+(beta-0.5*gamma)*delta_u_spec_sAvg(i-2,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*((0.5-alpha-
gamma)*delta_v_spec_sAvg(i,j+1,n)+(2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n)+(beta-0.5*gamma)*delta_v_spec_sAvg(i,j-2,n));
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
            end
        end
    end
end

if strcmp(SCHEME,'ELUD')
    alpha = 0.5;
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*u(i+2,j,n)+(0.5-alpha-
gamma)*delta_u_spec_sAvg(i+1,j,n)+(2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n)+(beta-0.5*gamma)*delta_u_spec_sAvg(i-2,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*v(i,j+2,n)+(0.5-alpha-
gamma)*delta_v_spec_sAvg(i,j+1,n)+(2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n)+(beta-0.5*gamma)*delta_v_spec_sAvg(i,j-2,n));
```

```
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
            end
        end
    end
end

if strcmp(SCHEME,'EQUD')
    alpha = 0.125;
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*delta_u_spec_sAvg(i+2,j,n)+(0.5-alpha-
gamma)*delta_u_spec_sAvg(i+1,j,n)+(2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n)+(beta-0.5*gamma)*delta_u_spec_sAvg(i-2,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*delta_v_spec_sAvg(i,j+2,n)+(0.5-alpha-
gamma)*delta_v_spec_sAvg(i,j+1,n)+(2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n)+(beta-0.5*gamma)*delta_v_spec_sAvg(i,j-2,n));
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
            end
        end
    end
end

if strcmp(SCHEME,'CUD')
    beta = alpha;
    gamma = alpha-1/6;
    xFP = 3;
    yFP = 3;
    xLP = xDispCoorNum-2;
    yLP = yDispCoorNum-2;

    for n = 1:1:specInterLocNum
        for j = yFP:1:yLP
            for i = xFP:1:xLP
                delta_Exx_spec_sAvg(i,j,n) = (1/(xDisp(2)-xDisp(1)))*(0.5*gamma*delta_u_spec_sAvg(i+2,j,n)+(0.5-alpha-
gamma)*delta_u_spec_sAvg(i+1,j,n)+(2*alpha+beta)*delta_u_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_u_spec_sAvg(i-1,j,n)+(beta-0.5*gamma)*delta_u_spec_sAvg(i-2,j,n));
                delta_Eyy_spec_sAvg(i,j,n) = (1/(yDisp(2)-yDisp(1)))*(0.5*gamma*delta_v_spec_sAvg(i,j+2,n)+(0.5-alpha-
gamma)*delta_v_spec_sAvg(i,j+1,n)+(2*alpha+beta)*delta_v_spec_sAvg(i,j,n)+(-0.5-alpha-
2*beta+gamma)*delta_v_spec_sAvg(i,j-1,n)+(beta-0.5*gamma)*delta_v_spec_sAvg(i,j-2,n));
```

```
                delta_e_spec_sAvg(i,j,n) = delta_Exx_spec_sAvg(i,j,n)+delta_Eyy_spec_sAvg(i,j,n);
            end
        end
    end
end

fprintf('CalSampAvgSpecInterLocDelStrnDltn_MultiSets: Computing delta strains and dilatation has finished.\n');

%% Plot incremental dilatation

resDir = strcat(savDir,'\Interface-location-specific incremental dilatation\');
mkdir(resDir);

for n = 1:1:specInterLocNum
    fig = figure('Name',['delta e sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    h = surf(xDisp,yDisp,delta_e_spec_sAvg(:,:,n).');
    set(h,'LineStyle','none');
    view(0,90);
    xlabel('x [um]');
    ylabel('y [um]');
    zlabel('delta e sAvg [dimensionless]');
    title(['delta e sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    axis([0 6000 0 6000 -0.1 0.1 -0.1 0.1]);
    axis square;
    axis manual;
    colorbar;
    caxis([-0.1 0.1]);
    caxis manual;
    saveas(fig,[resDir,'delta_e_spec_sAvg_X',num2str(n),'.png']);
    close(fig);
end

fprintf('CalSampAvgSpecInterLocDelStrnDltn_MultiSets: Plotting incremental dilatations at specific interface locations
has finished.\n');
```

```
% InterpSampAvgSpecInterLocDelDltn_MultiSets.m
% Ka Yaw Teo
% Last Updated: 1/12/2008

% DESCRIPTION: -
```

```
% This file consists of two sections of codes (variable set-up and interpolation).
% A message will be displayed when each section finishes running.
% The resulting variables are:

% delta_eI_spec_sAvg(i,j,n) where i = 1:1:xDispInterpCoorNum, j = 1:1:yDispInterpCoorNum, and n = 1:1:specInterLocNum
% xDispInterpCoorNum
% yDispInterpCoorNum
% xDispInterp(i)
% yDispInterp(j)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define interpolation parameters and variables (SPECIFY increment/resolution in xDispInterp and yDispInterp)

% Define interpolation parameters
xDispInterp = min(xDisp):10:max(xDisp);
yDispInterp = min(yDisp):10:max(yDisp);
xDispInterpCoorNum = size(xDispInterp,2);
yDispInterpCoorNum = size(yDispInterp,2);
[X,Y] = meshgrid(xDispInterp,yDispInterp);
delta_eI_spec_sAvg = zeros(xDispInterpCoorNum,yDispInterpCoorNum,specInterLocNum);

fprintf('InterpSampAvgSpecInterLocDelDltn_MultiSets: Defining interpolation parameters and variables has finished.\n');

%% Interpolate dilatation

for n = 1:1:specInterLocNum
    delta_eI_spec_sAvg(:,:,n) = interp2(xDisp,yDisp,delta_e_spec_sAvg(:,:,n).',X,Y,'cubic');
end

fprintf('InterpSampAvgSpecInterLocDelDltn_MultiSets: Computing delta_eI_spec_sAvg has finished.\n');

%% Plot dilatation contour

for n = 1:1:specInterLocNum
    fig = figure('Name',['delta eI sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    h = surf(xDispInterp,yDispInterp,delta_eI_spec_sAvg(:,:,n));
    set(h,'LineStyle','none');
    set(gca,'FontName','Arial');
    set(gca,'FontSize',16);
    set(gca,'FontWeight','bold');
    view(0,90);
    xlabel('x [um]');
```

```
    ylabel('y [um]');
    zlabel('delta eI sAvg [dimensionless]');
    title(['delta eI sAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    axis([0 6200 0 6200 -0.06 0.06 -0.06 0.06]);
    axis square;
    axis manual;
    colorbar;
    caxis([-0.06 0.06]);
    caxis manual;
    set(gca,'FontName','Arial');
    set(gca,'FontSize',16);
    set(gca,'FontWeight','bold');
    saveas(fig,[resDir,'delta_eI_spec_sAvg_X',num2str(n),'.png']);
    close(fig);
end

fprintf('InterpSampAvgSpecInterLocDelDltn_MultiSets: Plotting delta_eI_spec_sAvg contour has finished.\n');

%% Print dilatation

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_eI_spec_sAvg_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_eI_spec_sAvg_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "y", "delta_eI_spec_sAvgg"\n');
    fprintf(fid,'ZONE I=591, J=591, F=POINT\n');

    for i = 1:1:xDispInterpCoorNum
        for j = 1:1:yDispInterpCoorNum
            fprintf(fid,'%7.2e\t%7.2e\t%7.2e\n',xDispInterp(i),yDispInterp(j),delta_eI_spec_sAvg(i,j,n));
        end
    end

    fclose(fid);
end

fprintf('InterpSampAvgSpecInterLocDelDltn_MultiSets: Printing delta_eI_spec_sAvg has finished.\n');
```

```
% YAvgSampAvgSpecInterLocDelDltn_MultiSets.m
% Ka Yaw Teo
% Last updated: 1/3/2009

% DESCRIPTION: -
% This file consists of four sections of codes (variable set-up,
```

```
% dilatation averaging, plotting y-averaged dilatation, and printing results).
% A message will be displayed when each section finishes running.
% The resulting variables are:

% delta_e_spec_sAvg_yAvg(i,n) where i = 1:1:xDispCoorNum and n = 1:1:specInterLocNum
% std_delta_e_spec_sAvg_yAvg(i,n)

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's title.

%% Define variables

delta_e_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);
std_delta_e_spec_sAvg_yAvg = zeros(xDispCoorNum,specInterLocNum);

fprintf('YAvgSampAvgSpecInterLocDelDltn_MultiSets: Defining variables has finished.\n');

%% Compute y-averaged dilatation

for n = 1:1:specInterLocNum
    for i = 1:1:xDispCoorNum
        delta_e_spec_sAvg_yAvg(i,n) = mean2(delta_e_spec_sAvg(i,:,n));
        std_delta_e_spec_sAvg_yAvg(i,n) = std2(delta_e_spec_sAvg(i,:,n));
    end
end

fprintf('YAvgSampAvgSpecInterLocDelDltn_MultiSets: Computing delta_e_spec_sAvg_yAvg has finished.\n');

%% Plot y-averaged incremental dilatation

for n = 1:1:specInterLocNum
    fig = figure('Name',['delta e sAvg yAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)'],'NumberTitle','off');
    set(fig,'Visible','off');
    errorbar(xDisp,delta_e_spec_sAvg_yAvg(:,n),std_delta_e_spec_sAvg_yAvg(:,n));
    axis([0 6000 -0.1 0.1])
    axis square;
    axis manual;
    xlabel('x [um]');
    ylabel('delta e sAvg yAvg [dimensionless]');
    title(['delta e sAvg yAvg at X(t) = ',num2str(interLoc_spec_sAvg(n),'%4.2f'),' +/-
',num2str(std_interLoc_spec_sAvg(n),'%4.2f'),' um (t = ',num2str(t_spec_sAvg(n),'%3.2f'),' +/-
',num2str(std_t_spec_sAvg(n),'%3.2f'),' s)']);
    saveas(fig,[resDir,'delta_e_spec_sAvg_yAvg_X',num2str(n),'.png']);
    close(fig);
end
```

```
fprintf('YAvgSampAvgSpecInterLocDelDltn_MultiSets: Plotting y-averaged incremental dilatations at specific interface
locations has finished.\n');

%% Print y-averaged incremental dilatation

for n = 1:1:specInterLocNum
    fid = fopen(strcat(resDir,'delta_e_spec_sAvg_yAvg_X',num2str(n),'.dat'),'wt');
    fprintf(fid,strcat('TITLE = "delta_e_spec_sAvg_yAvg_X',num2str(n),'"\n'));
    fprintf(fid,'VARIABLES = "x", "delta_e_spec_sAvg_yAvg", "std_delta_e_spec_sAvg_yAvg"\n');
    fprintf(fid,'ZONE I=32, F=POINT\n');

    for i = 1:1:xDispCoorNum
        fprintf(fid,'%7.7e %7.7e %7.7e\n',xDisp(i),delta_e_spec_sAvg_yAvg(i,n),std_delta_e_spec_sAvg_yAvg(i,n));
    end

    fclose(fid);
end

fprintf('YAvgSampAvgSpecInterLocDelDltn_MultiSets: Printing y-averaged incremental dilatations at specific interface
locations has finished.\n');
```

```
% InterLoc_MultiSets.m
% Ka Yaw Teo
% Last updated: 10/17/2008

% DESCRIPTION: -
% This file consists of three sections of codes (determining interface
% location, curve fitting, and result printing).  A message will be displayed
% when each section finishes running. The resulting variable is:

% interLoc(k,s) where k = 1:1:tStepNum, and s = 1:1:setNum

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Determine interface location

max_delta_u_avg_mag = zeros(tStepNum,setNum);
max_delta_u_avg_xCoor = zeros(tStepNum,setNum);
interLoc = zeros(tStepNum,setNum);

for s = 1:1:setNum
    for k = 1:1:tStepNum
```

```
        [max_delta_u_avg_mag(k,s),max_delta_u_avg_xCoor(k,s)] = max(delta_u_avg(:,k,s));
        interLoc(k,s) = xDisp(max_delta_u_avg_xCoor(k,s));
    end
end

fprintf('InterLoc_MultiSets: Determining interface location has finished.\n');

%% Plot interface location over time

resDir = strcat(savDir,'\Interface Location Analysis\');
mkdir(resDir);

fig = figure('Name','Interface location','NumberTitle','off');
set(fig,'Visible','off');

for s = 1:1:setNum
    plot(t,interLoc(:,s),'b.','MarkerSize',15);
    hold on
end

axis([0 max(t) 0 6000]);
axis square;
axis manual;
xlabel('Time, t [s]');
ylabel('Phase change interface location, X [um]');
title('Phase Change Interface Location Vs. Time');
saveas(fig,strcat(resDir,'Interface location.png'));
close(fig);

fprintf('InterLocFit_MultiSets: Plotting interface location has finished.\n');
```

```
% AvgInterLocFit_MultiSets.m
% Ka Yaw Teo
% Last updated: 10/17/2008

% DESCRIPTION: -
% This file consists of three sections of codes (determining interface
% location, curve fitting, and result printing).  A message will be displayed
% when each section finishes running.

% IMPORTANT NOTE: -
% Please specify required information, if any, as stated in each section's
% title.

%% Average interface location from multiple sets
```

```
avgInterLoc = zeros(tStepNum,1);
std_avgInterLoc = zeros(tStepNum,1);

for k = 1:1:tStepNum
    avgInterLoc(k) = mean2(interLoc(k,:));
    std_avgInterLoc(k) = std(interLoc(k,:));
end

fprintf('AvgInterLocFit_MultiSets: Determining interface location has finished.\n');

%% Curve fitting

XFunc = inline('2*a*sqrt((10^-6)*t)','a','t');
a0 = 100;
N = 100;
var = 0.01;
sd = sqrt(var);
aList = zeros(1,N);

X = zeros(1,size(avgInterLoc,1));
T = zeros(1,size(t,1));

for count = 1:1:size(avgInterLoc,1)
        X(1,count) = avgInterLoc(count);
end

for count = 1:1:size(t,1)
        T(1,count) = t(count);
end

for count = 1:1:N
    % Form randomized, normal distributed initial parameter guesses
    a = a0*(1+sd*randn);

    % Curve fitting
    [a,R,J] = nlinfit(T,X,XFunc,a);
    aList(:,count) = a;
end

aMean = mean2(aList(1,:));

avgInterLocRec = zeros(4,1);
avgInterLocFitEqn = strcat('2*(',num2str(aMean),')*sqrt((10^-6)*t)');

avgInterLocRec(1) = solve(strcat(avgInterLocFitEqn,'= 1000'),'t');
avgInterLocRec(2) = solve(strcat(avgInterLocFitEqn,'= 2000'),'t');
```

```
avgInterLocRec(3) = solve(strcat(avgInterLocFitEqn,'= 3000'),'t');
avgInterLocRec(4) = solve(strcat(avgInterLocFitEqn,'= 4000'),'t');

fig = figure('Name','Interface location with curve fit','NumberTitle','off');
set(fig,'Visible','off');
plot(t,XFunc(aMean,t),'r-');
hold on;
errorbar(t,avgInterLoc,std_avgInterLoc,'b.','MarkerSize',15);
text(20,5200,texlabel('Curve fitting using Neumann solution:'));
text(20,4900,texlabel(strcat('X(t) = 2*lambda*(alpha*t)^(1/2)')));
text(20,4600,texlabel(strcat('lambda = ',num2str(aMean/(10^6),3))));
text(20,4300,texlabel(strcat('alpha = 1 mm^2/s')));
axis([0 max(t) 0 6000]);
axis square;
axis manual;
xlabel('Time, t [s]');
ylabel('Phase change interface location, X [um]');
saveas(fig,strcat(resDir,'Average interface location with curve fit.png'));
close(fig);

fprintf('AvgInterLocFit_MultiSets: Curve fitting has finished.\n');

%% Print avgInterLoc

fid = fopen(strcat(resDir,'avgInterLoc.txt'),'wt');
fprintf(fid,'t [s] \tx [um] \tx std [um] \tx_fit [um] \n');

for k = 1:1:tStepNum
    fprintf(fid,'%7.2e\t%7.2e\t%7.2e\t%7.2e\n',t(k),avgInterLoc(k),std_avgInterLoc(k),XFunc(aMean,t(k)));
end

fclose(fid);

fprintf('AvgInterLocFit_MultiSets: Printing avgInterLoc has finished.\n');
```

REFERENCES

[1] Nerem, R. M., 2006, "Tissue Engineering: The Hope, the Hype, and the Future," Tissue Engineering, 12(5), pp. 1143-1150.

[2] Pancrazio, J. J., Wang, F., and Kelley, C. A., 2007, "Enabling Tools for Tissue Engineering," Biosensors & Bioelectronics, 22(12), pp. 2803-2811.

[3] Archer, R., and Williams, D. J., 2005, "Why Tissue Engineering Needs Process Engineering," Nature Biotechnology, 23(11), pp. 1353-1355.

[4] Coger, R., and Toner, R., 1995, "The Biomedical Engineering Handbook," CRC Press, Boca Raton, pp. 1567-1577.

[5] Han, B., and Bischof, J. C., 2004, "Engineering Challenges in Tissue Preservation," Cell Preservation Technology, 2, pp. 91-112.

[6] Karlsson, J. O. M., and Toner, M., 1996, "Long-Term Storage of Tissues by Cryopreservation: Critical Issues," Biomaterials, 17(3), pp. 243-256.

[7] Karlsson, J.O.M., and Toner, M., 2000, "Principles of Tissue Engineering," Academic Press, San Diego, CA, pp. 293-307.

[8] Schenke-Layland, K., Madershahian, N., Riemann, I., 2006, "Impact of Cryopreservation on Extracellular Matrix Structures of Heart Valve Leaflets," The Annals of Thoracic Surgery, 81(3), pp. 918-926.

[9] Schenke-Layland, K., Xie, J., Heydarkhan-Hagvall, S., 2007, "Optimized Preservation of Extracellular Matrix in Cardiac Tissues: Implications for Long-Term Graft Durability," The Annals of Thoracic Surgery, 83(5), pp. 1641-1650.

[10] Narine, K., Ing, E. C., Cornelissen, M., 2006, "Readily Available Porcine Aortic Valve Matrices for Use in Tissue Valve Engineering. Is Cryopreservation an Option?" Cryobiology, 53(2), pp. 169-181.

[11] Toner, M., 1993, "Advances in Low-Temperature Biology," JAI Press, London, pp. 1-51.

[12] Karlsson, J. O. M., Cravalho, E. G., and Toner, M., 1994, "A Model of Diffusion-Limited Ice Growth Inside Biological Cells during Freezing," Journal of Applied Physics, 75, pp. 4442-4445.

[13] Fahy, G. M., 1980, "Analysis of "Solution Effects" Injury. Equations for Calculating Phase Diagram Information for the Ternary Systems NaCl-Dimethylsulfoxide-Water and NaCl-Glycerol-Water," Biophysical Journal, 32(2), pp. 837-850.

[14] Pegg, D. E., and Diaper, M. P., 1989, "The 'Unfrozen Fraction' Hypothesis of Freezing Injury to Human Erythrocytes: A Critical Examination of the Evidence," Cryobiology, 26(1), pp. 30-43.

[15] Caffrey, M., 1987, "The Combined and Separate Effects of Low Temperature and Freezing on Membrane Lipid Mesomorphic Phase Behavior: Relevance to Cryobiology," Biochimica et Biophysica Acta, 896(1), pp. 123-127.

[16] Crowe, J. H., Hoekstra, F. A., Crowe, L. M., 1989, "Lipid Phase Transitions Measured in Intact Cells with Fourier Transform Infrared Spectroscopy," Cryobiology, 26(1), pp. 76-84.

[17] Jameel, F., Bogner, R., Mauri, F., 1997, "Investigation of Physicochemical Changes to L-Asparaginase during Freeze-Thaw Cycling," Journal of Pharmacy and Pharmacology, 49(5), pp. 472-477.

[18] Pikal-Cleland, K. A., Rodriguez-Hornedo, N., Amidon, G. L., 2000, "Protein Denaturation during Freezing and Thawing in Phosphate Buffer Systems: Monomeric and Tetrameric Beta-Galactosidase," Archives of Biochemistry and Biophysics, 384(2), pp. 398-406.

[19] Mazur, P., 1970, "Cryobiology: The Freezing of Biological Systems," Science, 168(934), pp. 939-949.

[20] Bischof, J., Hunt, C. J., Rubinsky, B., 1990, "Effects of Cooling Rate and Glycerol Concentration on the Structure of the Frozen Kidney: Assessment by Cryo-Scanning Electron Microscopy," Cryobiology, 27(3), pp. 301-310.

[21] Hong, J. S., and Rubinsky, B., 1994, "Patterns of Ice Formation in Normal and Malignant Breast Tissue," Cryobiology, 31(2), pp. 109-120.

[22] Pazhayannur, P. V., and Bischof, J. C., 1997, "Measurement and Simulation of Water Transport during Freezing in Mammalian Liver Tissue," Journal of Biomechanical Engineering, 119(3), pp. 269-277.

[23] Muldrew, K., Novak, K., Yang, H., 2000, "Cryobiology of Articular Cartilage: Ice Morphology and Recovery of Chondrocytes," Cryobiology, 40(2), pp. 102-109.

[24] Han, B., Grassl, E. D., Barocas, V. H., 2005, "A Cryoinjury Model using Engineered Tissue Equivalents for Cryosurgical Applications," Annals of Biomedical Engineering, 33(7), pp. 972-982.

[25] Han, B., Miller, J. D., and Jung, J. K., 2009, "Freezing-Induced Fluid-Matrix Interaction in Poroelastic Material," Journal of Biomechanical Engineering, 131(2), pp. 021002.

[26] Rhee, S., and Grinnell, F., 2007, "Fibroblast Mechanics in 3D Collagen Matrices," Advanced Drug Delivery Reviews, 59(13), pp. 1299-1305.

[27] Tamariz, E., and Grinnell, F., 2002, "Modulation of Fibroblast Morphology and Adhesion during Collagen Matrix Remodeling," Molecular Biology of the Cell, 13(11), pp. 3915-3929.

[28] Grinnell, F., 1994, "Fibroblasts, Myofibroblasts, and Wound Contraction," The Journal of Cell Biology, 124(4), pp. 401-404.

[29] Bell, E., Ivarsson, B., and Merrill, C., 1979, "Production of a Tissue-Like Structure by Contraction of Collagen Lattices by Human Fibroblasts of Different Proliferative Potential in Vitro," Proceedings of the National Academy of Sciences, 76(3), pp. 1274-1278.

[30] Raffel, M., 2007, "Particle Image Velocimetry: a Practical Guide," Springer, New York, pp. 86-88.

[31] Özısık, M.N., 1993, "Heat Conduction," Wiley, New York, pp. 405-408.

[32] Pedersen, J. A., and Swartz, M. A., 2005, "Mechanobiology in the Third Dimension," Annals of Biomedical Engineering, 33(11), pp. 1469-1490.

[33] Venkatasubramanian, R. T., Grassl, E. D., Barocas, V. H., 2006, "Effects of Freezing and Cryopreservation on the Mechanical Properties of Arteries," Annals of Biomedical Engineering, 34(5), pp. 823-832.

[34] Devireddy, R. V., Neidert, M. R., Bischof, J. C., 2003, "Cryopreservation of Collagen-Based Tissue Equivalents. I. Effect of Freezing in the Absence of Cryoprotective Agents," Tissue Engineering, 9(6), pp. 1089-1100.

[35] Han, B., and Bischof, J. C., 2004, "Thermodynamic Nonequilibrium Phase Change Behavior and Thermal Properties of Biological Solutions for Cryobiology Applications," Journal of Biomechanical Engineering, 126(2), pp. 196-203.

[36] Han, B., Choi, J. H., Dantzig, J. A., 2006, "A Quantitative Analysis on Latent Heat of an Aqueous Binary Mixture," Cryobiology, 52(1), pp. 146-151.

[37] Petroll, W. M., and Ma, L., 2003, "Direct, Dynamic Assessment of Cell-Matrix Interactions Inside Fibrillar Collagen Lattices," Cell Motility and the Cytoskeleton, 55(4), pp. 254-264.

[38] Roeder, B. A., Kokini, K., Robinson, J. P., 2004, "Local, Three-Dimensional Strain Measurements within Largely Deformed Extracellular Matrix Constructs," Journal of Biomechanical Engineering, 126(6), pp. 699-708.

[39] Petroll, W. M., Cavanagh, H. D., and Jester, J. V., 2004, "Dynamic Three-Dimensional Visualization of Collagen Matrix Remodeling and Cytoskeletal Organization in Living Corneal Fibroblasts," Scanning, 26(1), pp. 1-10.

[40] Kim, A., Lakshman, N., and Petroll, W. M., 2006, "Quantitative Assessment of Local Collagen Matrix Remodeling in 3-D Culture: The Role of Rho Kinase," Experimental Cell Research, 312(18), pp. 3683-3692.

[41] Huang, H., Dong, C. Y., Kwon, H. S., 2002, "Three-Dimensional Cellular Deformation Analysis with a Two-Photon Magnetic Manipulator Workstation," Biophysical Journal, 82(4), pp. 2211-2223.

[42] Roy, P., Petroll, W. M., Cavanagh, H. D., 1997, "An in Vitro Force Measurement Assay to Study the Early Mechanical Interaction between Corneal Fibroblasts and Collagen Matrix," Experimental Cell Research, 232(1), pp. 106-117.

[43] Wang, C. C., Deng, J. M., Ateshian, G. A., 2002, "An Automated Approach for Direct

Measurement of Two-Dimensional Strain Distributions within Articular Cartilage Under Unconfined Compression," Journal of Biomechanical Engineering, 124(5), pp. 557-567.

[44] Olsen, M. G., Bauer, J. M., and Beebe, D. J., 2000, "Particle Imaging Technique for Measuring the Deformation Rate of Hydrogel Microstructures," Applied Physics Letters, 76(22), pp. 3310-3312.

[45] Johnson, B. D., Bauer, J. M., Niedermaier, D. J., 2004, "Experimental Techniques for Mechanical Characterization of Hydrogels at the Microscale," Experimental Mechanics, 44(1), pp. 21-28.

BIOGRAPHICAL INFORMATION

Ka Yaw Teo is currently working towards his B.S. in Biology and M.S. in Biomedical Engineering at the University of Texas at Arlington. After graduation in the fall of 2009, he intends to pursue his Ph.D. in Mechanical Engineering with an emphasis in Bioengineering at Purdue University and continue to conduct research in biotransport phenomena under the guidance of Dr. Bumsoo Han.