2012

Analysis of Microstructure and Macrosegregation for Directional Solidification of Al-7 WT Si Alloy with a Cross-Section Change at Growth Speeds of 10 and 29.1 [mu]m/s

Luke Johnson
Cleveland State University

ANALYSIS OF MICROSTRUCTURE AND MACROSEGREGATION FOR DIRECTIONAL SOLIDIFICATION OF AL-7 WT% SI ALLOY WITH A CROSS-SECTION CHANGE AT GROWTH SPEEDS OF 10 AND 29.1 μM/S.

LUKE JOHNSON

Bachelor of Mathematics

Concordia University-Saint Paul

MAY 2006

Submitted in partial fulfillment of requirements for the degree

MASTER OF SCIENCE IN CHEMICAL ENGINEERING

at the

CLEVELAND STATE UNIVERSITY

May 2013

This Thesis has been approved for
the department of Chemical and Biomedical Engineering
and the College of Graduate Studies by:

_____

Dr. Surendra N. Tewari
Thesis committee Chair Person

_____

Date
Chemical and Biomedical Engineering

_____

Dr. Orhan Talu

_____

Date
Chemical and Biomedical Engineering

_____

Dr. Rolf Lustig

_____

Date
Chemical and Biomedical Engineering

ACKNOWLEDGEMENTS

First, I would like to thank my graduate advisor, Dr. Surendra Nath Tewari, for his guidance, wealth of knowledge and extensive time during the course of my studies at Cleveland State University and throughout the research and thesis composal period.

Next I would like thank the committee members: Dr. Orhan Talu and Dr. Rolf Lustig for taking the time to read my manuscript and give helpful critiques while maintaining a flexible schedule.

Thank you to the Microgravity Materials Research Program at NASA – Marshall Space Flight Center for continued support of this research, without it this would not have been possible.

Last but not least I'd like to thank my wholly loving, supportive, and beautiful wife Krista who has patiently encouraged me throughout this long process. I also give thanks for my daughter Aurora, who will always be my sunshine.

ANALYSIS OF MICROSTRUCTURE AND MACROSEGREGATION FOR DIRECTIONAL SOLIDIFICATION OF AL-7 WT% SI ALLOY WITH A CROSS-SECTION CHANGE AT GROWTH SPEEDS OF 10 AND 29.1 μM/S.

LUKE JOHNSON

ABSTRACT

Directionally solidified alloys of Al-7 wt. % Si composition were solidified in a Bridgman furnace at 10 and 29.1 μm/s growth speeds. Graphite crucibles were machined with an initial inside diameter of 9.2 mm in diameter, and undergo a sharp cross-section constriction of 3.2 mm in the middle of the crucible length, before widening again to 9.2 mm. Four thermocouples measured temperature gradients of ~30 K/cm for the slower growth speed alloy, and ~50 K/cm for the faster growth speed alloy. Subsequent transverse slices were then sliced, mounted, polished, and photographed at a high magnification using standard metallography techniques. Microstructure is measured and analyzed using past image analysis techniques, and compared to past studies. A new automatic image analysis technique is developed and used to measure phase distribution, and therefore predict concentration along the length of the alloys. The effect of cross-section change and varying growth speeds is analyzed and compared among samples using image analysis techniques to establish several trends for macrosegregation, primary arm diameter, secondary arm length, nearest neighbor spacing, primary spacing, and secondary arm orientation.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 SOLIDIFICATION OF ALLOYS

1.1.1 CONVENTIONAL CASTING

Conventional alloy casting techniques pour a hot alloy liquid into a cold mold. Therefore, heat transfer, thermal gradients and solidification speeds ar4e dictated by component and mold geometry. Solidification begins when the hot melt comes in contact with the cold mold, which then forms many equiaxed grains. Growth is dictated by competition among these nucleation sites; grains oriented more favorably to the heat extraction direction grow at the expense of the unfavorably oriented ones. This results in columnar grain morphology. As solidification proceeds, the rest of the melt continues to cool. Finally, if the melt undercools below its liquidus temperature, new grains of solid nucleate and grow. This again yields equiaxed grain morphology during the final phase of solidification which can be seen in Figure 1.

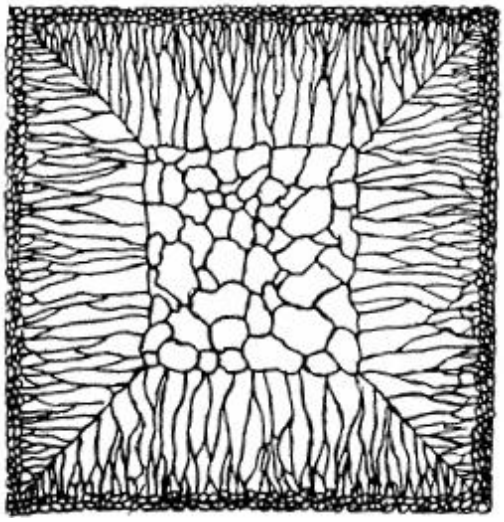

Figure 1: Sketch representing columnar to equiaxed grain transition in a traditional cast. Flemings, pg. 135 [1]

1.1.2 DIRECTIONAL SOLIDIFICATION

Directional solidification imposes a relatively constant thermal gradient in a single direction to help maintain a single liquid-solid interface. By maintaining adiabatic walls on all sides except one, heat is extracted in only one direction. To maintain the imposed thermal gradient G [K/cm] at the liquid-solid interface, the entire ampoule is moved away from the heat source at a desired 'growth' rate R [cm/s]. In a binary alloy for a high thermal gradient to growth rate ratio, the liquid-solid interface maintains a planar liquid-solid front morphology. However for most other cases the alloys are directionally solidified at a low thermal gradient to growth rate ratio and form a dendritic array (see Figure 2). Both the beginning and the end of the alloy is primarily a single phase, but the middle section with an array of dendrites dictating the liquid-solid interface is referred to as the 'mushy-zone.'

1.2 MUSHY ZONE MORPHOLOGY

1.2.1 DENDRITES

Dendrites (example shown in Figure 2) are best described as 'trees', with a main growth 'trunk' (primary dendrite), a set of four orthogonal 'branches' (secondary arms), each of which has tertiary 'branches'. The top of the dendrite right at the liquid-solid interface is referred to as the *tip* with a corresponding solid-liquid composition C_{TIP} and radius r_{TIP}. Trunk spacing, trunk diameter, and branch length are dependent on growth conditions and alloy physical properties. Faster growth speeds produce finer features and tighter packing, and therefore smaller grains [2, 3, 4, 5]. As this forest of solid trees grow, amongst the primary dendrite trunks remains the liquid melt with changing composition along its length (as schematically shown in Figure 4). Morphology and distribution of primary dendrites, secondary dendrites and tertiary dendrites depend upon the alloy composition and local solidification conditions, and determine the mechanical properties of solidified components. Therefore, any inhomogeneity in the microstructure is disconcerting.

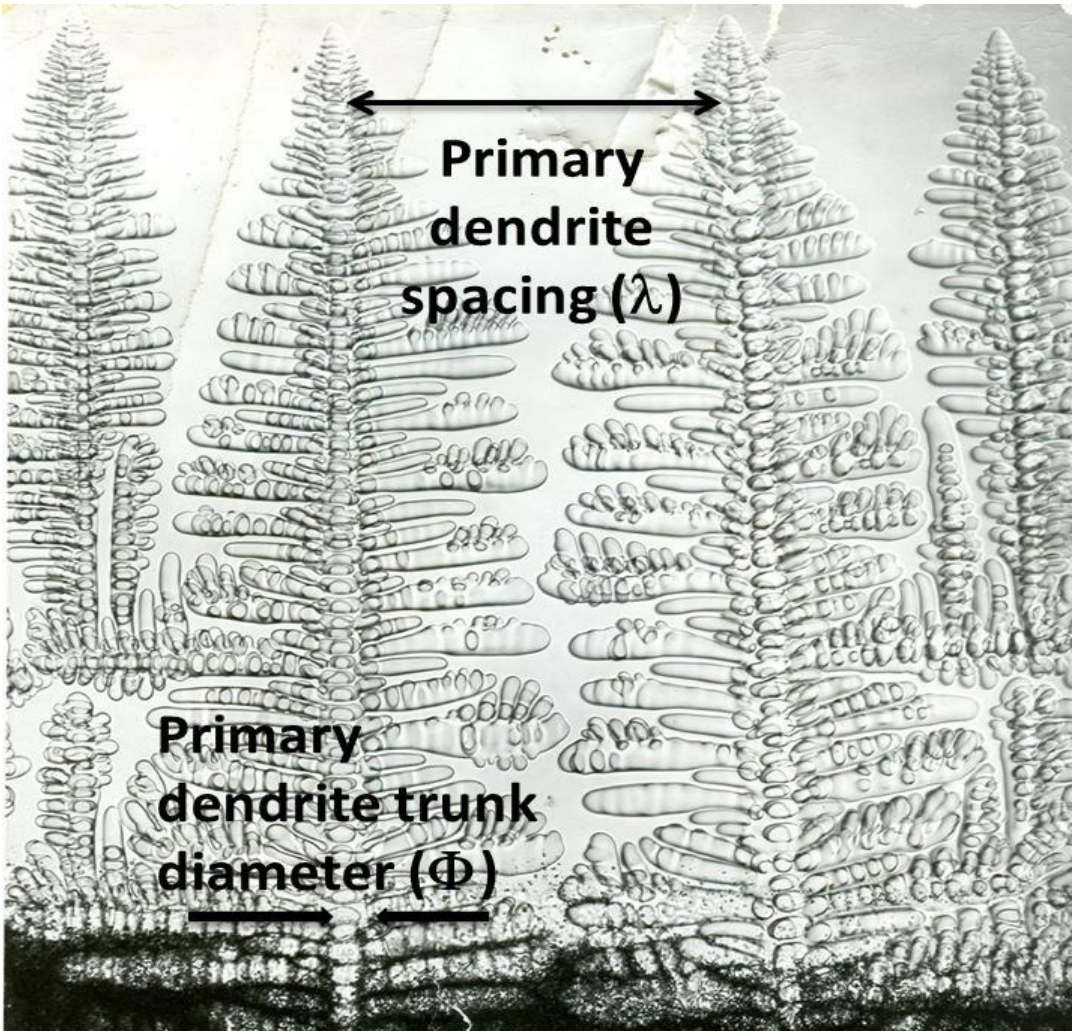

Figure 2: Succinonitrile – 9 wt. % Water "Transparent alloy" directionally solidified (~ 5μm s^{-1} ,~30Kcm^{-1}) (Dr. Grugel, NASA-MSFC)

1.2.2 EUTECTIC

The melt amongst the dendrites, rich in solute rejected by immiscibility during cooling, forms a eutectic phase. This phase is of the alloy's eutectic composition at the first point of solidification, the eutectic temperature. Unlike other binary compositions that first form a single solid phase surrounded by liquid, the eutectic immediately splits into two liquid phases which then solidifies. This simultaneous split and solidification process

brings the two phases in equilibrium with one another through diffusion, giving predictable sizes and spacing in many instances[1]. However in the case of the Al-Si system, silicon grows anisotropically compared to the isotropic nature of aluminum. Particularly at large undercoolings and thermal gradients, the eutectic phase solidifies in an irregular and unpredictable pattern[6]. Figure 3 depicts an example of irregular eutectic, with silicon rich dark elongated particles. Unlike regular eutectic, these elongated particles are not parallel with one another, will face in random directions, will not be of consistent sizes, and will not have consistent spacing.

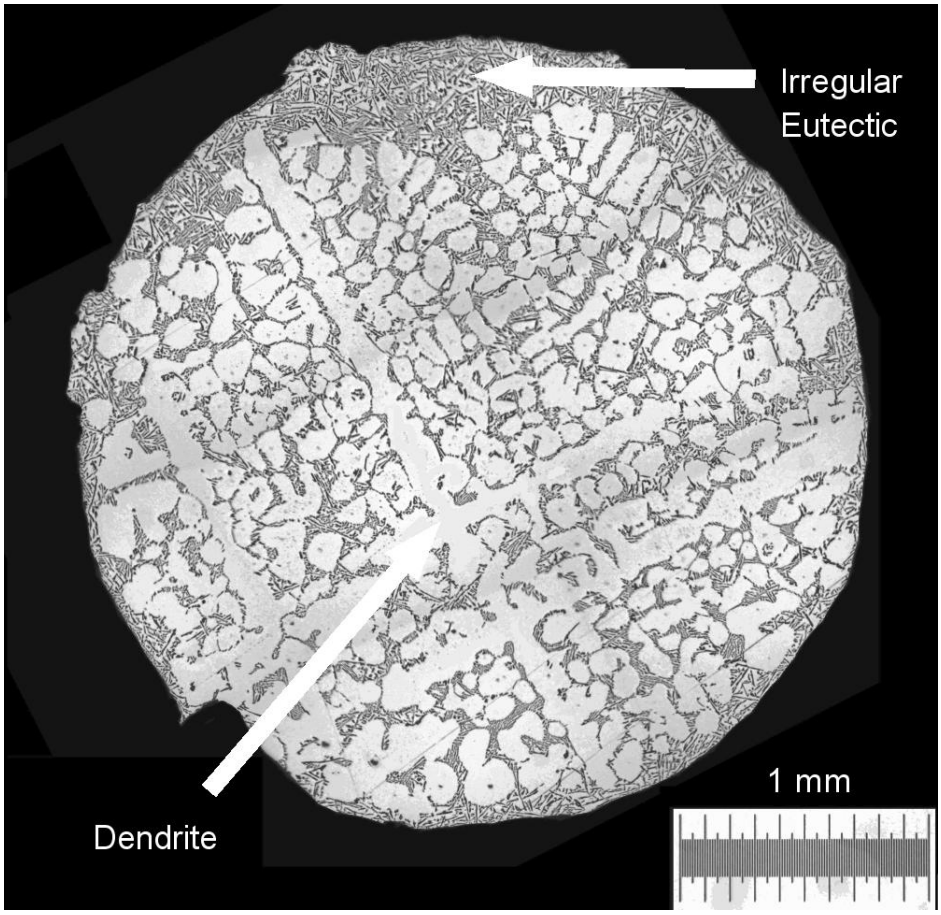

Figure 3: Transverse Image of DS Al- 7 wt. pct. Si (G=32 K/cm, R=10 µm/s)

1.2.3 SOLUTALLY DESTABILIZED INTERDENDRITIC MELT

As shown schematically in Figure 4, during directional solidification of alloys with solutal partition coefficient (solid composition/liquid composition under equilibrium) less than one, the solute content of the melt in the mushy zone decreases from the eutectic (C_E) at the bottom to the tip composition (C_{TIP}) at the array tips. The solutal build-up at the tip associated with the tip curvature decays over distances of the order of tip radius to the initial overall alloy solute content, C_o. The positive thermal gradient (growth direction is the same as heat transfer direction) in the melt creates a decreasing interdendritic melt density towards the array tip which is stabilizing against natural convection (with gravity pointing down). The increasing solute content towards the tip, can either result in increased melt density, which is the case with for example Al-19 wt% Cu alloy (increasing copper content results in higher melt density), or decreased melt density as is the case with Pb-10wt% Sn alloy (increasing solute (tin) content of tin results in reduced melt density, schematically presented in Figure.)If the combined thermal and solutal effects (called thermosolutal) create a density inversion in the melt then convection ensues and results in compositional inhomogeneities along the solidified length of the sample, called macrosegregation[7, 8, 9, 10, 1].Such severe thermosolutal convection can cause a visible defect called 'freckles'[3, 10, 11].

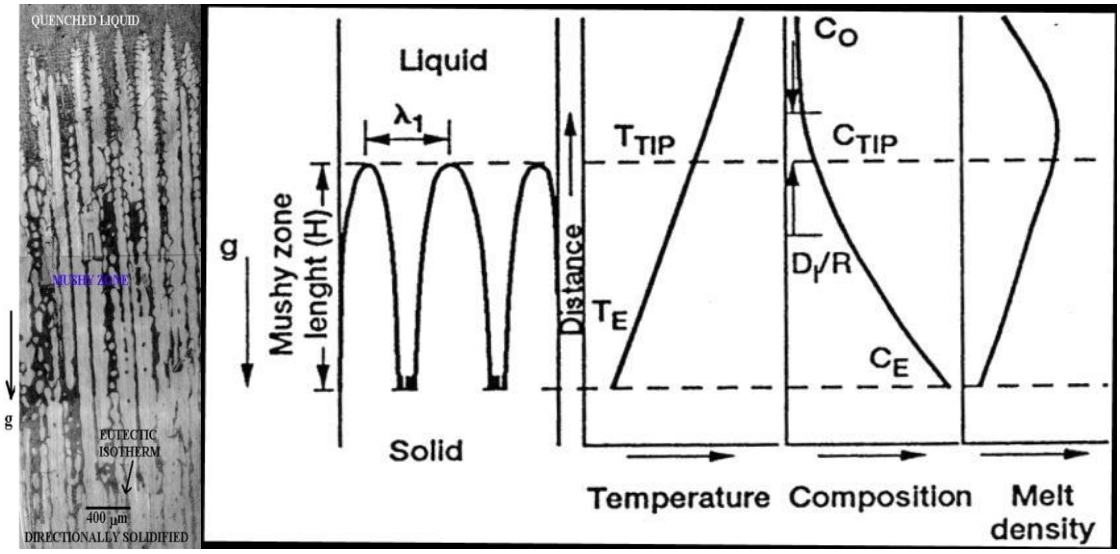

Figure 4: Schematic Representation of Density Profile of Pb-10% Sn

1.2.4 SOLUTALLY STABILIZED INTERDENDRITIC MELT

Directional solidification of alloys, such as Al - 27 wt% Cu, in a positive thermal gradient with hot melt on top and cold solid below produces a thermally and solutally stabilizing interdendritic melt density profile, since solute rich melt is heavier (Copper: 8.98 g/cm^3 to Al: 2.70 g/cm^3 at melting temperature). Therefore no convection is expected in the mushy-zone. However, if one primary dendrite lags in growth behind its neighbor then the solute rich (C_T) heavier melt from the leading dendrite tip flows down diagonally towards the tip of lagging dendrite (solute content ~ C_o). This can initiate a cascade process which results in a non-uniform array tip morphology; the mushy array tip liquid-solid interface is not flat but it is now "convex" towards the melt-side. This results in radial macrosegregation (along the sample diameter) shown in Figure 5. In a typical cross-section the primary dendrites appear clustered into a "steeple" near one wall, where the opposite

wall contains less primary phase and more interdendritic eutectic phase [12, 13]. This is macrosegregation in the transverse plane.

Figure 5: Image of transverse slice of an Al-26.5 wt% Cu copper alloy, grown with 30 K cm^{-1}thermal gradient at 4.2 µm s^{-1}[12]. This depicts an example of transverse phase macrosegregation.

1.2.5 SOLUTALLY NEUTRAL MELT

There are no binary alloys where solute enrichment during solidification does not change the melt density. However, in Al-Si alloy system, the solvent (aluminum) and solute (silicon) are very similar in densities (respectively 2.375 g/cm^3 to 2.329 g/cm^3 at melting point) and *this was the main reason why Al-7 wt% Si was selected for this study.* An Al-Si phase diagram is shown in Figure 6, and some selected thermophysical properties required by theoretical models to predict primary dendrite spacings are listed in Table 1. Liquid temperature is the melting temperature, liquid slope is the slope of the liquidus line on the

phase diagram, solute partition coefficient is the ratio of compositions for the two phases at a given temperature, eutectic temperature is the temperature at which eutectic phase forms, eutectic composition is the overall composition of the eutectic phase, Gibbs-Thomson coefficient is a constant describing the surface curvature effects for phase equilibrium, and the solutal capillary length is the characteristic length defined by the surface tension between the two phases.

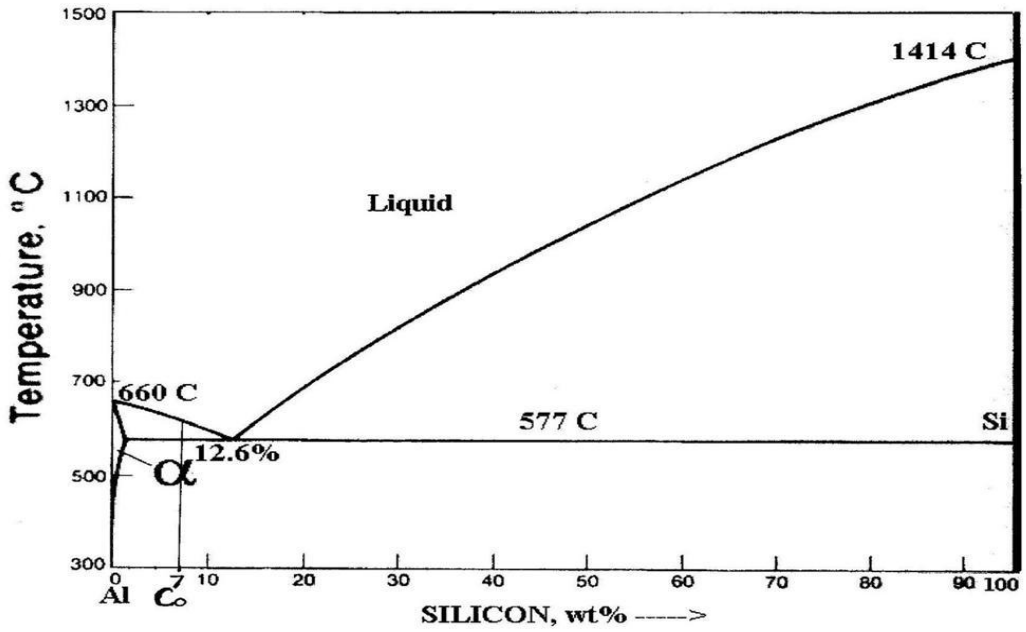

Figure 6: Al-Si Phase Diagram[15]

Table 1: Select thermophysical properties of Al – 7 wt. pct. Si System[15], used in theoretical models to predict primary dendrite spacing.

Name	Variable	Units	Value
Liquid Temperature	T_L	°C	614

Liquid Slope	m_L	K/wt%	6.62
Solute Partition Coefficient	k	dimensionless	0.13
Eutectic Temperature	T_E	^oC	577
Heat of Fusion	Δh_f	Jm^{-3}	12.6
Entropy of Fusion	ΔS_f	$Jm^{-3}\,K^{-1}$	$-9.5*10^8$
Eutectic Composition	C_E	wt%	$-1.02*10^6$
Gibbs-Thomson Coefficient	Γ	µmK	0.12
Solutal Capillary Length	S_e	m	$2.9*10^{-9}$

1.3 MATHEMATICAL MODELS TO PREDICT PRIMARY DENDRITE ARRAY MORPHOLOGY

Mushy-zone dendritic array morphology forming during directional solidification is difficult to model analytically or numerically. Primary dendrite spacing, the dendrite tip-radius, the tip composition, etc., have been analytically and numerically modeled after making many simplifying assumptions [2, 4, 5, 8, 16]. Some of these assumptions include pure thermal and solutal diffusion (no convection) and simple "needle" (unbranched) primary dendrite shape. Numerical models have been used to simulate convection through a mushy-zone of dendrites [17, 18, 19], but several simplifications were made to solve for the moving liquid-solid interface. The primary challenge for numerical simulation has been the "non-linear" nature of the phenomenon. Convective contribution to the heat and solute transport in the melt, mush, and the solid has only been calculated for a "time invariant liquid-solid mushy-zone structure" (for example known permeability), but the "mushy-

zone" dendritic array morphology (for example permeability) is changed by the presence of "convection"[1].

Despite these limitations, several semi-theoretical models have been proposed to predict some of the macro morphology parameters such as dendrite tip radius (ρ), dendrite tip composition (C_{tip}), and primary spacing of dendrite trunks (λ). Some popular models in literature include Kurz-Fisher [4], Trivedi [5] and Hunt-Lu [16][2]. The Hunt-Lu model is a semi-empirical model that uses experimental polynomial 'fit' parameters, and has proven to correlate well over a wide range of growth speeds, thermal gradients, and compositions. This will be the only model used to compare experimental results for primary spacing. The Hunt-Lu model [HL] is a function of the dimensionless parameters:

$$G' = \frac{G\Gamma}{(mC_o)^2}, \quad V' = \frac{V\Gamma}{DmC_o}, \quad \text{and} \quad \lambda' = \frac{\lambda mC_o}{\Gamma} \quad \text{where } \Delta T_o = \frac{m_L C_o(k-1)}{k}$$

G is the effective thermal gradient, Γ is the capillary length (proportional to the ratio of solid-fluid surface energy to the heat of fusion), D is the diffusion coefficient, k is the solute partition coefficient, C_o is the initial alloy composition, m_L is the liquidus slope, and λ is the trunk diameter. For calculation, all variables are assumed to be constant. The following is the HL model with parameters:

- Curvature Undercooling $\Delta T'_\sigma$:
  - $\Delta T'_\sigma = 0.41(V' - G')^{0.51}$
- Tip radius ρ:
  - $\rho = \frac{2\Gamma}{\Delta T'_\sigma \Delta T_o}$

- Primary spacing λ':
    - $\lambda' = 0.15596V'^{(a-0.75)}(V' - G')^{0.75}G'^{-0.6028}$
    - $a = -1.131 - 0.1555\log(G') - 0.007589[\log(G')]^2$

As referenced for the Al-7 wt. pct. Si system in Table 1, k=0.13, D=5x10^{-5} cm^{2}s^{-1}, Γ=0.12 μmK^{-1}, m=6.62 K/wt.pct. and C_o=7 wt. pct. Currently there are no analytical or numerical models to predict the primary dendrite trunk diameter.

1.4 SINGLE CRYSTAL TURBINE BLADES

First stage turbine blades are the most critical component in advanced gas turbine engines, as these are the most stressed high temperature components in the engine[20]. These nickel-based superalloy blades have a dendritic single crystal morphology with their [100] direction parallel to the loading direction (i.e. along the blade length), and are fabricated by directional solidification (DS) using investment casting shell molds. Since the [100] orientation provides the blade its maximum stress-rupture life, presence of any spurious (not aligned along [100]) grains is detrimental and must be avoided during directional solidification [21, 22]. These blades have a complex outer shape, involving cross-section changes at the blade-root and the blade-tip locations, and often have complex internal cooling channels which are formed by directional solidification of the melt though many cross-section changes. Two techniques are used to achieve the [100] orientation. The first technique uses a [100] single crystal seed kept at the shell bottom, which is re-melted and fused with the rest of the melt before directional solidification of the blade. The other

technique uses a small diameter helix shaped constriction at the shell bottom to select one near [100] orientation grain from many incoming random grains, which are generated when the hot melt, poured through the shell touches a quench block kept at the shell bottom[11]. A presence of "misoriented" spurious grains in the blade casting is a major cause of rejection during the process. It is believed that shrinkage-driven (thermal volumetric contraction) and thermosolutal convection associated with solidification through cross-section changes is responsible for the formation of these spurious grains. Severe macrosegregation (freckles"[22]) is another major cause for blade casting rejection. Macrosegregation caused by advection ("solidification shrinkage driven convection) was experimentally and numerically examined by Flemings, Nereo, and Mehrabian [7, 8, 9]and is believed to be the main cause of many solidification defects, such as, formation of stray misoriented grains, non-uniform composition in the blade (macrosegregation) and severe macrosegregation called "channel segregates" or "freckles"[22].

1.5 EXPERIMENTAL GOALS

Macrosegregation caused by a cross-section change was first examined and modeled for Al-4.5 wt% Cu [7, 8, 9], attributed to the changes in thermally induced flow. Multiple studies have indicated that convection is the primary cause of irregular mushy-zone morphology, e.g. decrease in primary spacing, secondary spacing, and misoriented grain formation [23, 10, 15, 24, 3, 6, 25, 26]. However, to date there has been no experimental study to examine both the macrosegregation and the microstructural changes caused by convection associated with cross-section change during directional solidification of metallic alloys. The purpose of this study is to examine mushy-zone morphology and composition change

associated with cross-section change during directional solidification of Al – 7 wt. pct. Si alloys at varying directional solidification growth speeds. The aluminum-silicon alloy was selected in-order to minimize the mushy zone convection caused by melt density inversion (solutal-driven convection) and to maximize the role of "cross-section change" driven convection during directional solidification. This project has four specific purposes:

1. Carry out directional solidification of Al-7wt% Si alloy through cross-section increase and cross-section decrease at two growth speeds.
2. Measure mushy-zone morphology (primary dendrite spacing, primary dendrite trunk diameter) along the length of directionally solidified samples by using image-analysis techniques and establish trends if any.
3. Establish image analysis techniques for a quantitative analysis of phase amounts distribution in the microstructure as an indicator of macrosegregation.
4. Measure transverse and longitudinal macrosegregation associated with cross-section change during directional solidification and establish trends if any.

This data is expected to help identify solidification processing parameters for a future low gravity experiment on the International Space Station.

CHAPTER II

EXPERIMENTAL PROCEDURE

2.6 DIRECTIONAL SOLIDIFICATION FURNACE

The directional solidification furnace assembly used during these experiments is shown in Figure 7. The left side of the figure is an overall view of the furnace assembly, showing the cylindrical quartz chamber supported between top and bottom stainless steel flanges. A diffusion pump vacuum ($\sim 2*10^{-4}$ torr) is maintained within the chamber during directional solidification. A sixty watt RF power supply is used to heat a 20 cm long, 5 cm outside diameter, and 2.5 cm inside diameter cylindrical hollow graphite susceptor to create the hot-zone of the Bridgman furnace assembly. The graphite susceptor is enclosed within a 28 cm long insulating alumina shell. A 0.5 cm insulating shell is kept on top of the susceptor to minimize heat loss from the above. A 2 cm long insulating ceramic hollow disk, placed at the susceptor bottom, within the alumina creates the adiabatic zone at the furnace bottom. The furnace assembly kept within the alumina shell is hung by two tantalum wires tied to the top stainless steel flange.

The close-up view shown on the right shows the alumina shell and the induction heating coils from the RF generator. The graphite crucible containing the Al-7% Si alloy cylinders (described below) is inserted into the hanging furnace from the bottom for alloy re-melting and its subsequent directional solidification. The alloy containing crucible is attached to a stainless-steel feed through rod at the bottom. Withdrawing the rod down wards or raising it upwards allows the graphite crucible to be pulled out from the Bridgeman furnace or inserted into the furnace from the bottom. The sample kept within the graphite crucible, as it comes out the furnace, is cooled by radiation to the water cooled inner walls of the graphite enclosure, kept at room temperature by the help of flowing water within the two-walled assembly of the chamber. A DC motor coupled to a worm gear arrangement is used for the sample translation described above. Four thermocouple feeds are available at the bottom flange, so that temperatures measured along the length of the graphite crucible can be recorded by a data logger.

Figure 7: Images of Bridgman-technique furnace assembly for directional solidification

2.7 GRAPHITE CRUCIBLE AND SAMPLE ASSEMBLY

The as-cast feed rods used in these experiments were provided by the ALCOA technical center in Pittsburgh. The alloy was prepared by melting together 99.99% Aluminum and 99.99% Silicon under an argon atmosphere, and then poured into a quenched block to produce 30 cm long rods, 9 mm in diameter. The ALCOA technical center also chemically analyzed these rods for impurities.

Figure 8: Image of typical graphite crucible used for solidification. 30 cm long and 1.9 cm outside diameter with 13 cm long by 9.5 mm diameter cavity on both ends, and a 5 cm long by 3.2 mm diameter cavity in the center. Thermocouples ID# TC1, TC2, TC3, and TC4 mounted at 8, 13, 18, and 23 cm respectively from the 'cold-end' or bottom.

Figure 8 shows a typical graphite crucible used for these experiments. From the cold end of the crucible, a 9 mm diameter by 1.8 mm long Al - 7 wt. pct. Si alloy seed is inserted with known [100] crystal orientation. The cast feed rods inserted from the top of the crucible are melted during heating, the melt flows down and fuses with the [100] oriented Al-7% Si alloy seed kept at the bottom. Care is taken to ensure only partial melting of the [100] seed at the bottom.

2.8 DIRECTIONAL SOLIDIFICATION

The graphite crucible containing the seed at the bottom and the feed rod at the top is inserted into the vacuum chamber and attached to the stainless steel feed through rod passing through the bottom flange. The quartz chamber is evacuated by the help of mechanical and diffusion pumps to achieve a $2*10^{-4}$ torr vacuum. The water valves are opened and RF generator turned on. The graphite crucible is heated to a preset temperature to achieve a steady-state hot zone temperature of about 750 ^{o}C. The graphite crucible is then inserted into the furnace and TC1 through TC4 temperatures are continually recorded. The feed rod is re-melted and fused with the seed below and then the sample is withdrawn from the furnace at a desired speed (e.g.10 μm $s^{-1}$for sample No. 11-17-11 or at 30 μm $s^{-1}$for sample #12-14-11).

Thermocouple ID# TC1, TC2, TC3, and TC4 temperature profiles recorded during directional solidification of sample #11-17-11 are shown in Figure 9. The temperature versus time data have been transformed to temperature versus distance data using the known sample translation speed, in this case $\mu m\ s^{-1}$. The temperature gradient G from these thermal profiles, as determined between the liquidus and the eutectic temperatures of the alloy, are 32.7, 31.6, 39.8, and 63.7 K/cm. The steeper thermal gradient recorded by TC4 is attributed to the high thermal conductivity of both the graphite crucible and solid alloy with a large fraction of length in the cooling zone extracting heat at a faster rate. The thermal gradients measured for the other sample examined in this study, #12-14-11, by the thermocouples TC1, TC2, TC3 and TC4 temperature profiles are 50.6, 50.6, 50.6, and 78.1 K/cm respectively. This sample was directionally solidified at a growth seed of 29.1 $\mu m\ s^{-1}$

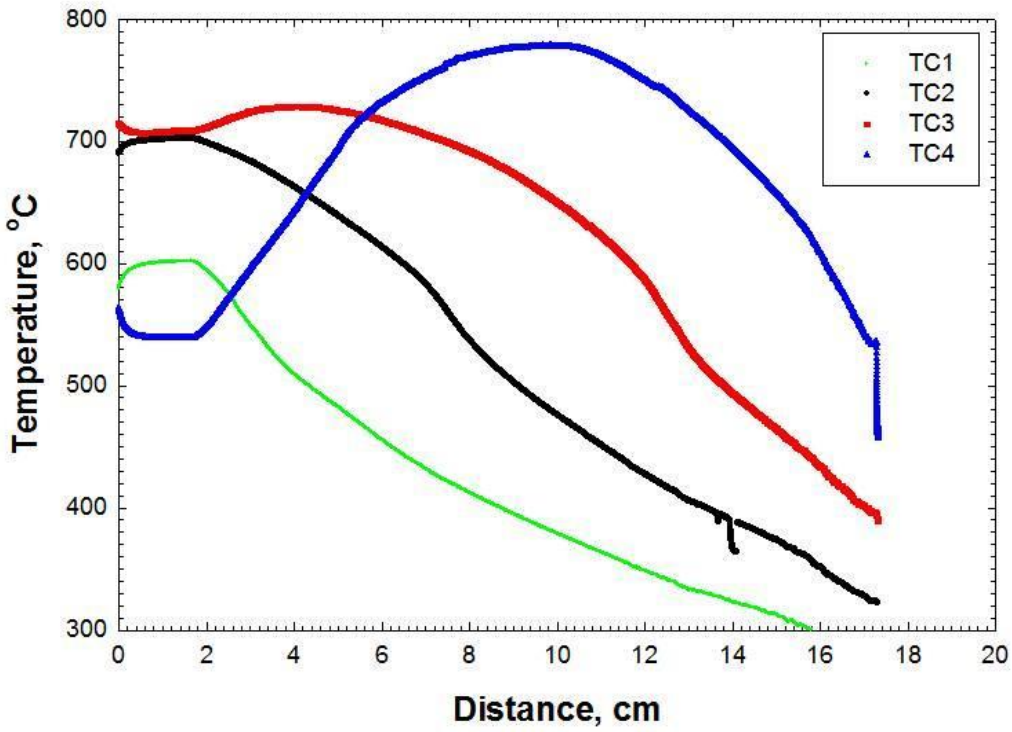

Figure 9: Temperature versus distance for thermocouples TC1, TC2, TC3, and TC4 for sample 11-17-11.

2.9 SPECIMEN PREPARATION AND METALLOGRAPHY

2.9.1 LONGITUDINAL SAMPLES

Once each crucible had completely cooled, they were removed from the furnace in preparation for cutting and mounting. Original experiment design identified longitudinal, as opposed to transverse, to be beneficial for measuring composition along the length. However, this would later prove to add to analytical difficulties. Approximately 7 cm long portion, starting from about 1 cm below the neck of contracting cross-section and ending at about 1-cm above the neck of the expanding cross-section was cut-off from the directionally solidified rod for sectioning along the length. Approximately two 2 mm thick transverse

slices were taken from the middle, and from the two ends. For both growth speeds these were labeled 11_17_2b, 11_17_3b, 11_17_4T, 11_17_5T, 12_14_1T, 12_14_3B, 12_14_3T, and 12_14_4T2. The remaining two halves were then each sliced longitudinally to create pieces 11-17-2L, 11-17-5R, 12-14-2R, and 12-14-4R where 12-14 can be seen in Figure 10. "L/R" designates left or right of the longitudinal cut while facing the cold-end. "2/4" or "2/5"is approximate distances from the beginning of the sample to the cross section change of that piece.

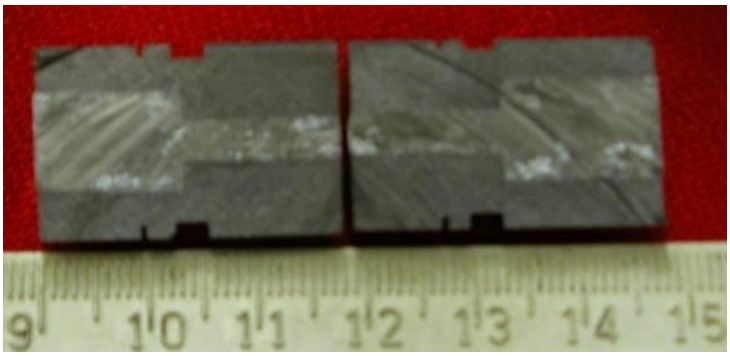

Figure 10: Image of 11-17-11 in its crucible

2.9.2 TRANSVERSE SAMPLES

After the longitudinal slices were mounted, polished, and imaged (which will be described in 2.9.3: CUTTING, MOUNTING, AND POLISHING), it was decided that transverse images along the length would still be needed to examine morphology features such as arm length and orientation. The original raw longitudinal halves were then carefully aligned and glued with their mounted partners. The now glued 'whole' pieces were sliced 2 mm at a time down the length, taking great care to not slice through the cross-section change. Each slice was marked which side was 'hot' (nearest to the aforementioned 'hot-zone' during

solidification) and the distance of 'hot-side' to the cross-section change. Each was then mounted, polished, and imaged for a total of 20 transverse slices for 11-17 and 23 transverse slices for 12-14. Though care was taken to align each longitudinal half, shifting is still likely, which will vary length coordinate between halves. Also, the process of cutting removed 0.3 mm of sample between halves, so that other coordinate data cannot be assumed constant between halves. This effect was difficult to measure, and therefore ignored. Summation of the different samples and their corresponding locations along the length are presented in Table 2.

Table 2: Slice Labels (ID) and Distance from the Cross-Section Change

SAMPLE NAME	DISTANCE FROM CROSS-SECTION CHANGE [mm]	SAMPLE NAME	DISTANCE FROM CROSS-SECTION CHANGE [mm]
Samples pertaining to cross-section decrease		Samples pertaining to cross-section decrease	
11_17_2b	-13	12_14_1T	-13.6
11-17-2H	-9.3	12-14-2I	-7.2
11-17-2G	-7.2	12-14-2H	-5.3
11-17-2F	-5.1	12-14-2G	-2.9
11-17-2E	-3	12-14-2F	-1.2
Cross-section change	0	Cross-section change	0
11-17-2D	0.7	12-14-2E	2.5
11-17-2C	2.8	12-14-2D	5
11-17-2B	4.9	12-14-2C	7.4
11-17-2A	7	12-14-2B	9.7

11_17_3b	13	12-14-2A	12
		12_14_3B	13.8
Samples pertaining to cross-section decrease		Samples pertaining to cross-section decrease	
11_17_4T	-15.5	12_14_3T	-15
11-17-5H	-8	12-14-4J	-11.9
11-17-5G	-5.6	12-14-4I	-5.7
11-17-5F	-3.3	12-14-4H	-4
11-17-5E	-1.2	12-14-4G	-1.6
Cross-section change	0	Cross-section change	0
11-17-5D	2.5	12-14-4F	0.7
11-17-5C	4.9	12-14-4E	4.4
11-17-5B	6.7	12-14-4D	6.8
11-17-5A	8.5	12-14-4C	9.2
11_17_5T	11.1	12-14-4B	11.6
		12-14-4A	14
		12_14_4T2	15

2.9.3 CUTTING, MOUNTING, AND POLISHING

Well established metallography techniques were used to prepare the specimens for microscopy[27]. A low-speed, variable RPM, precision wafer saw cut the samples. Saw blades used are 102 mm diameter by 0.3 mm thick diamond tipped, low-grit precision blades. Between 25 - 50 grams of weight were added to the cutting arm, and blades were continuously wetted with cutting oil.

Once sliced, the carefully marked specimens were mounted using a thermosetting dry granular epoxy and heated hydraulic mounting press to cure the resin. Mounting cylinder and cap were greased with silicone oil to avoid 'sticking.' The specimen and phenolic resin cured for 5 minutes at 120 ^{o}C under 21 MPa of pressure. Once hardened, each sample was engraved with its sample ID into the epoxy.

Samples were ground and polished with an automatic, wetted abrasive grinder. The several layers of polishing are described in table 3. After each round of polishing, pads and samples were thoroughly cleaned under running water with the soft 'pads' of fingers to carefully scrub away silica residue. Often samples would need to be re-polished, solvated with acetone, or cleaned in an ultrasonic bath to remove colloidal silica particles and silicone oil. The silicone oil is especially intractable, as it can become trapped between sample and epoxy during curing, releasing droplets over an extended time period, and coating polishing pads with oil. Over-polishing with the finest grit can lead to wearing away the softer alpha phase, creating an embossed surface with beveled edges. Once finished, each sample was capped to limit oxide growth.

Table 3: Polishing Procedure for Sample Preparation.

ABRASIVE GRADE (GRIT)	DOWNWARD FORCE/SAMPLE	TIME (min)	POLISHING PAD RPM
400	3 lbs	1	120
600	3 lbs	1	120
800	3 lbs	2	120

1200	3 lbs	2	120
0.05 μm	5 lbs	10	110

2.9.4 MICROSCOPY

A metallurgical brightfield inverted Nikon microscope, 5.0 megapixel digital camera attachment, and SPOT 5.0 Image software were used to capture magnified images of the sample's surface. Images were recorded at 50 times magnification for 11_17_2b, 11_17_3b, 11_17_4T, 11_17_5T, 12_14_1T, 12_14_1B, 12_14_3B, 12_14_4T2, and 100 times magnification for all other samples. For such a large field of view, several overlapping images were tediously and manually staged. At larger magnifications, 9 mm diameter cross-sections required upwards of 120 images to cover the field of view. Original digital images were saved with JPEG extensions at the highest quality. Light exposure, gamma corrections, gain, and color-filters were manually selected sample-to-sample to create best image quality. Ideal image quality was seen to have constant light intensity across the whole field of view, constant light intensity between samples and images, large light absorption differences between silicon and aluminum, and constant light intensity for one phase to avoid shadowing. To capture consistent images, proper alignment and cleanliness of the microscope's collector and projection lenses was vital; slight skew causes light intensity fading in corners. Typical settings include: 159.9 ms of exposure, a 1.0 gain factor, gamma correction of 0.5, and a green-tint light filter. For each sample, a known 1 mm scale was imaged to record pixel to millimeter ratio for images.

Mistakenly, several batches of samples were cut, mounted, polished, and stored for later microscopy. Oxide layers will grow rapidly on the metal's surface, causing the

refractive index and absorption coefficient to vary within a field of view and sample-to-sample in an unpredictable manner. Also, optical properties of both aluminum and silicon are well known, and future experiments should calibrate ideal optical conditions to separate the absorption of light between binary elements. Both these observations were only realized months from data capture, causing error for subsequent image analysis, and should be noted for future experiments. However this was not significant enough to compromise our results.

2.10 IMAGE ANALYSIS

2.10.1 IMAGE STITCHING

'Image stitching' automatically overlays multiple overlapping images by first using algorithms to identify image locations, calibrating optical aberrations, and blending where applicable. Microscopic images don't require transformations to adjust distortions, but are difficult to create 'image registration' of locations because features are often mathematically similar. Intensity differences between images, unfocused features, and most importantly intensity gradient in an image will cause difficulties with stitching. If successfully stitched, two images can be blended by averaging intensities and colors to hide seams. Larger digital files require more computing capabilities, and therefore more difficult to stitch. Because of these complications, image stitching is a time-limiting procedure that has potential to be simplified with more advanced technology. Note that the complexity of stitching will lead to computation errors, which leads to errors in image analysis.

This experiment's images were stitched using Adobe Photoshop CS5.1 on 64-bit Windows 7 on an HP z210 workstation . In most cases manual adjustments were required after final results. A 120 image stitch required 10 - 14 hours of computing time, and often

lead to computer 'freezes.' A smaller 10 - 20 image collection typically required 10 - 15 minutes to stitch.

Before analyzing the images for data it's important that subsequent slices are aligned, so that all have a similar reference point for rotation and are also centered within a background. Microscopy requires manual alignment at a high magnification, which would be cumbersome. Instead, images were aligned using the original vertical cuts between the halves. For the whole transverse end caps, dendrite feature's orientation were measured between images, and then rotated to match.

2.10.2 PRIMARY DENDRITE TRUNK DIAMETER AND DENDRITE TRUNK CENTERS

Primary dendrite trunk diameter was measured using ImageJ v. 1.46. ImageJ is an open source image analysis program that can be downloaded from http://rsb.info.nih.gov/ij/ for a variety of operating systems. Drawing a line between two pixels with (x,y) coordinates can calculate length where $d = \sqrt{x^2 + y^2}$. Before drawing, appropriate measurement values are selected by choosing Analyze --> Set Measurements -->Bounding rectangle --> OK. After choosing the line-selection tool and making a line-selection on an image, Analyze --> Measure will print a set of numbers for a 'Bounding rectangle.'

As seen in Figure 11, bounding rectangle has 6 measurements labeled BX, BY, Width, Height, Angle, and Length. For a given line with two sets of (x,y) coordinates, a rectangle can be drawn with the extreme values of x and y for the four corners of the rectangle, where the drawn line is one of the diagonals. 'BX' and 'BY' are the upper-left x and y coordinate in pixels in relation to the upper-left corner of an image, 'Width' and 'Height' are the width

and height of the bounding rectangle, 'Angle' is the minimum angle between the drawn line and an imaginary line facing due right from the 'starting' point for the drawn line, and 'Length' is the length of the drawn line.

For a dendrite trunk diameter measurement, two lines are drawn such that each line is a *minimum* distance spanning the trunk, as shown in Figure 11. For a well-formed dendrite, these two lines will be perpendicular to each other. The diameter of that trunk was taken as the average of those two lines lengths in pixels.

Dendrite center was also calculated from these measurements by knowing the intersection of these two lines. This calculation was done through a computer program written in VBA coding language for Excel 2010, listed in the appendix. Though dendrite center absolute (x,y) coordinates are irrelevant, coordinates are important for spacing calculations which will be discussed in the later sections.

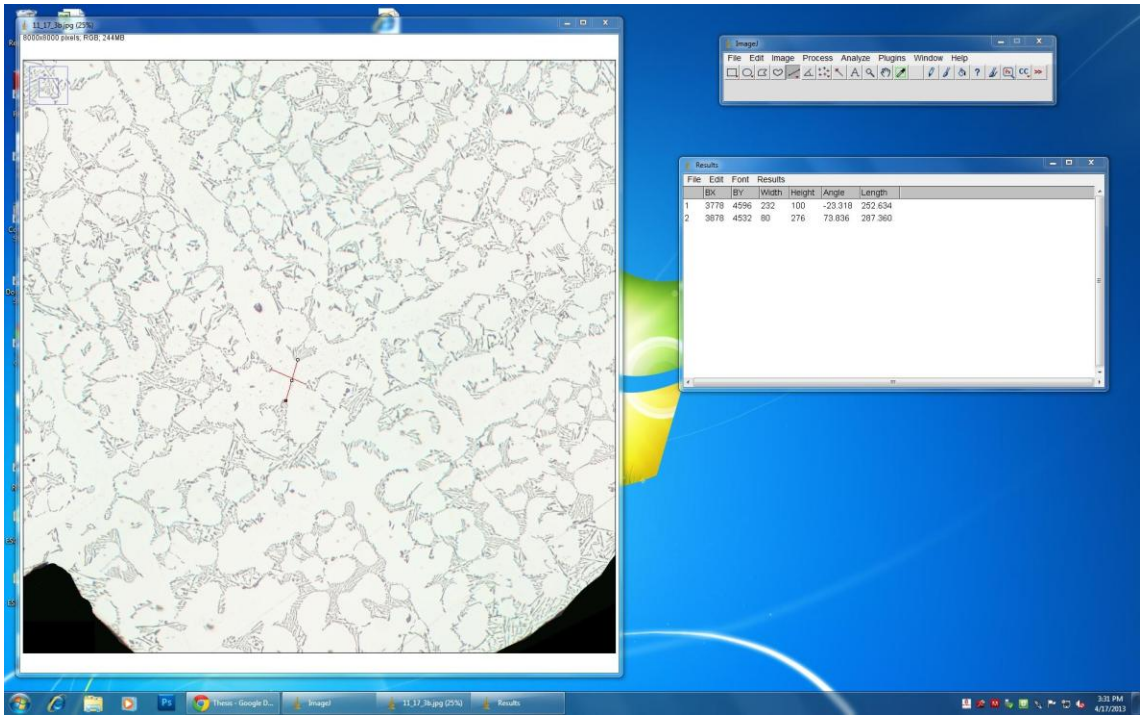

Figure 11: Screen capture of bounding rectangle measurement for trunk diameter

2.10.3 SECONDARY ARM LENGTH AND ORIENTATION

Similar to the process described in the previous section, secondary arm length is a length measurement of two lines. Again, measurement settings were chosen as 'Bounding Rectangle' before making a measurement. Once the measurement has been made, each arm 'length' is the sum of both branches since this constitutes the entire line. The line is drawn such that lines do not necessarily extend tip-to-tip since secondary arms have a certain degree of bend. Rather, arm measurements were made so that both extend through the center point, with even spacing on all sides in the trunk region.

Also from the bounding rectangle measurements can be calculated arm 'orientation.' Dendrites of the same grain will have arms in the same array, and therefore same 'orientation.' Figure 12 give an example of typical arm and angle measurements. Angle α

and β are the measured angles minus the nearest horizontal or vertical line. For a well-formed dendrite α equals β.

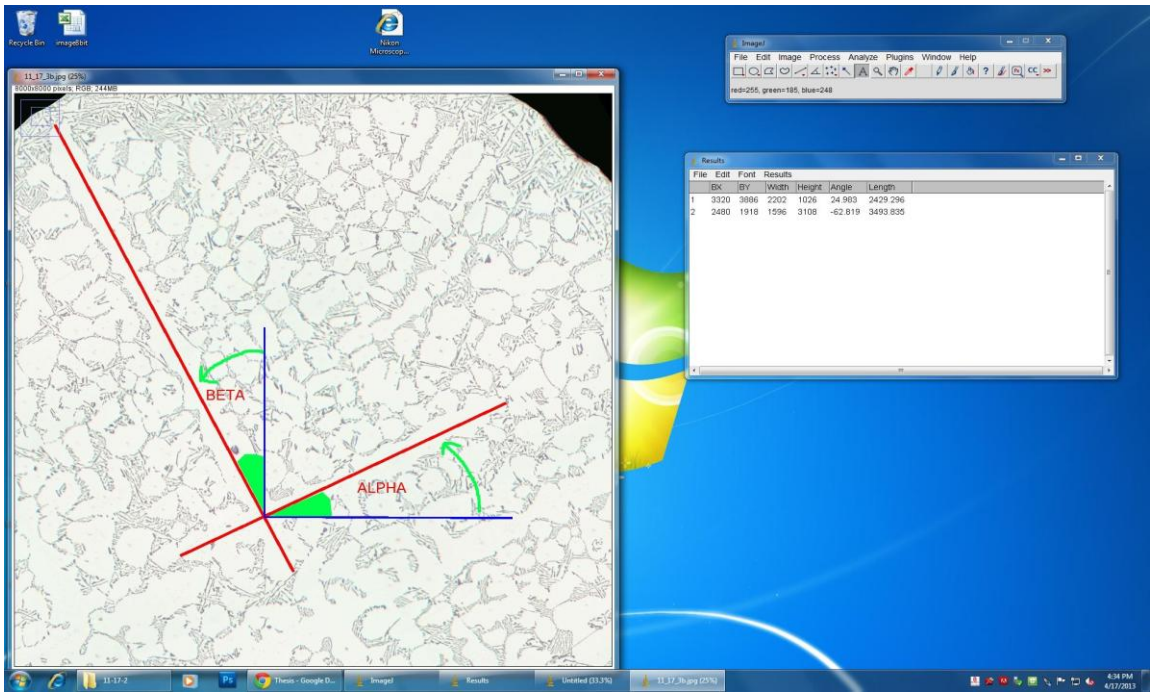

Figure 12: Screen capture of bounding rectangle measurement for arm length and orientation

2.10.4 NEAREST NEIGHBOR AND PRIMARY SPACING

Nearest neighbor and primary spacing are not an image analysis, but rather numerical analysis of (x,y) coordinates measured in ImageJ. Primary spacing is defined as

n. n. $= \sqrt{\frac{A}{N-1}}$, where A is area and N is number of points. This assumes that all points are evenly spaced from one another. Nearest neighbor is a computer calculated average distance between points for a set of points, where $d = \sqrt{x^2 + y^2}$. One distance is selected per set of (x,y) coordinates that is the minimum of the set of all possible distances, and the set of all possible distances will not contain duplicates through the symmetric property of equality[28]. Given a set of (x,y) coordinates, a VBA macro code for Excel 2010 was written to calculate nearest neighbor spacing. The ratio between nearest neighbor spacing and primary spacing will be a non-dimensional number between 0 and 1, where a low number indicates 'clustering' and a high number indicates uniform distribution of primary dendrites on a cross-section.

2.10.5 FRACTION EUTECTIC

2.10.5.1 THRESHOLDING

From the lever rule, if the area covered by eutectic divided by total area is known, then composition would also be known[1]. Aluminum rich and silicon rich phases have significantly different light absorption, and therefore they can be identified based on color intensity[27]. High intensity aluminum rich α phase can then be separated from the silicon rich eutectic.

Separating pixels based on intensity values, which for an 8-bit image are labeled 0 to 255 where 0 is black and 255 is white, is called thresholding[28]. In order to threshold, a number is picked where all numbers equaling that value or higher will be given a value of 255, and all lower a value of 0. Thresholding then transforms an 8-bit gray scale image into a binary image of black and white pixels. As not all images are the same, picking a proper

cut-off intensity value is vital for accuracy to ensure user bias is not introduced. ImageJ has a built-in thresholding algorithm named the “Huang” method which was used for thresholding [29].

Since thresholding accounts for all pixel intensities of an image to calculate an initial intensity mean, it is important that unwanted background pixels are not included in the calculation. This can be done by first selecting a ‘Region of Interest’, and then executing the algorithm. In the ImageJ menu select: Image --> Adjust --> “Threshold...” Once a proper threshold value is selected manually between background and image select: Edit --> Selection --> “Create Selection.” Then Select “reset” threshold. The same procedure is used to select a Threshold in a ‘Region of Interest’ as mentioned previously, except now the threshold is applied. The image is then inverted so now dendrite phase is black, and silicon phase is white.

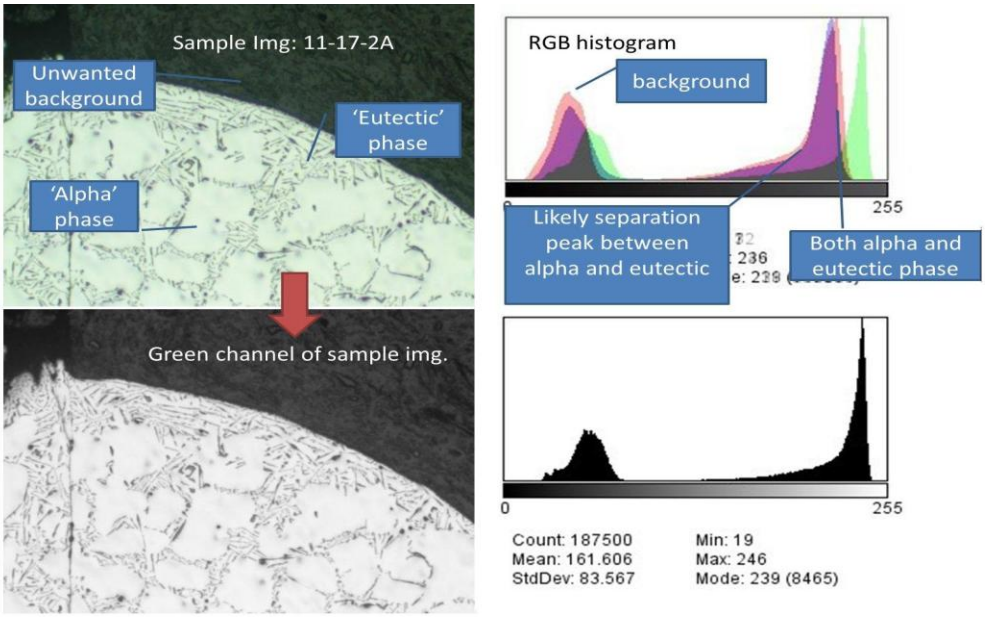

Figure 13: Typical example depicting the thresholding procedure of an RGB image of an Al-Si transverse slice with corresponding intensity histograms.

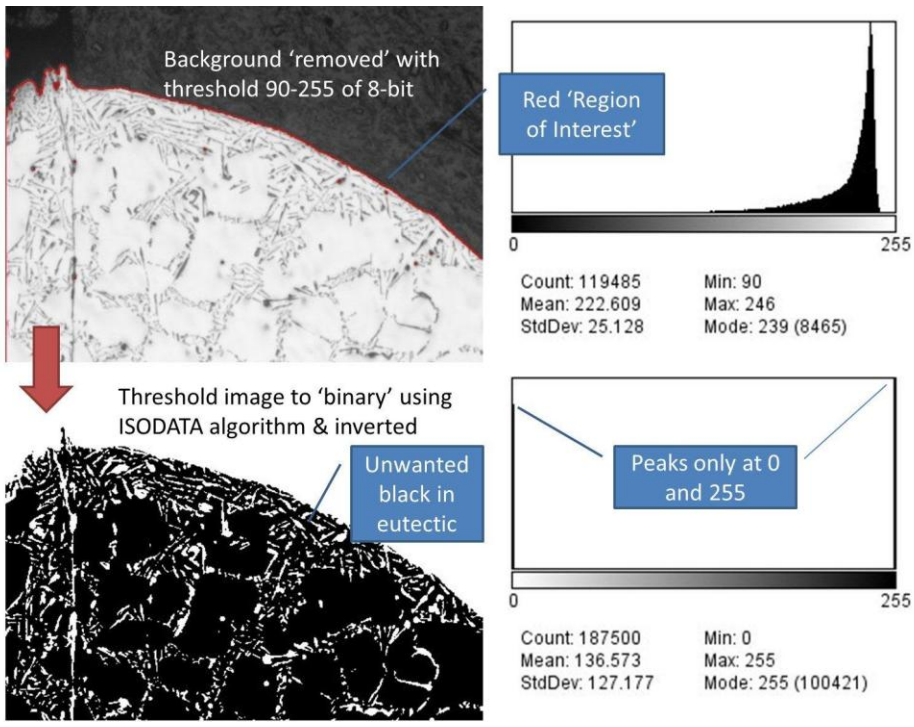

Figure 14: Typical example depicting the thresholding procedure of an 8-bit image of an Al-Si transverse slice with corresponding intensity histograms.

2.10.5.2 Watershed

Unfortunately, simply thresholding does not separate the phases by color, since not all black pixels (as seen in the bottom of Figure 14) can be considered dendrite. Within the eutectic remains aluminum rich metastable phases which are colored similarly to dendrite α phase. Solely thresholding and counting pixels would include this unwanted phase in calculations and therefore would be inaccurate. To separate α from dendrite aluminum rich phases, a binary image is transformed with a watershed algorithm.

A watershed transformation draws 1 pixel wide white pixels through geographical minimum distances between regions of white, similar to how in a rainstorm rivulets 'watershed' a landscape by seeking the lowest ground [28]. Now the binary image of silicon-rich white particles embedded into aluminum-rich black landscape are connected by a spider web of lines, sectioning the aluminum-rich black landscape into particles. Also it should be noted the drawn white pixels add to the white count and subtract from the black count by 0.5 - 1% of the total pixel count (which can be calculated by [black particle count before watershed] - [black particle count after watershed]).

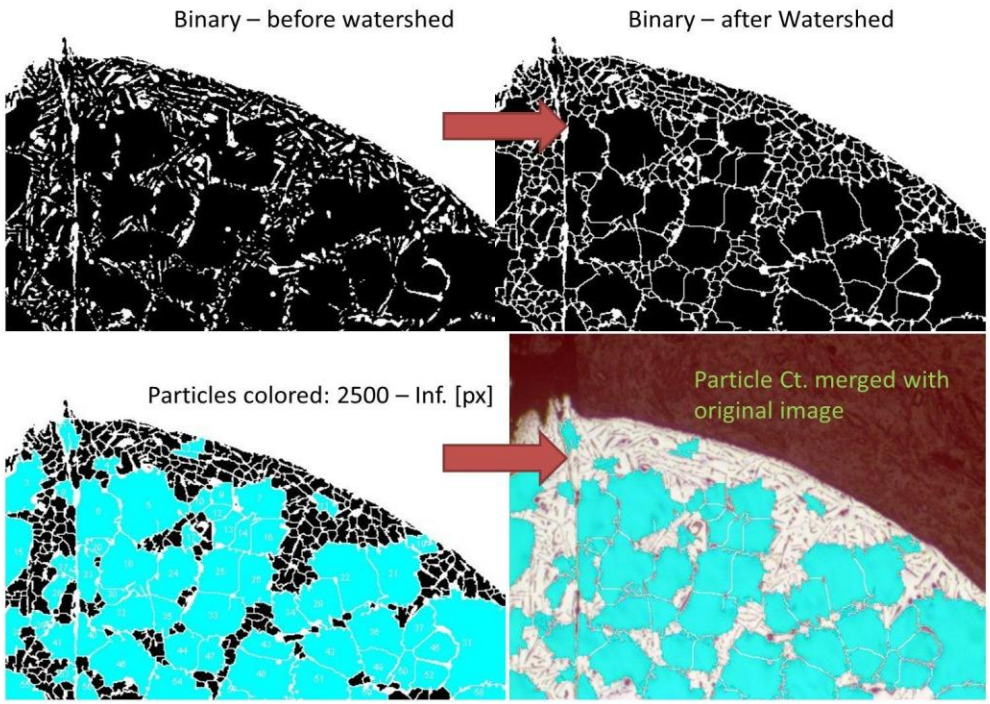

Figure 15: Typical example depicting the watershed and particle area cut-off procedure of a binary image. Aqua colored area representing alpha/dendrite phase.

2.10.5.3 Particle Size Cut-Off

In the eutectic phase, the silicon-rich black regions are tightly packed (see Figure 15). Therefore watershedding the eutectic creates small particles of aluminum-rich α-phase within the eutectic. Similarly, the sparse dendrite phases (primary α) were sectioned into much larger particles. To eliminate eutectic αphase from calculations, particle areas smaller than a certain pixel area cutoff value will be considered as being part of the eutectic. Through trial-and-error, a standard pixel area cut-off value of 8000 pixels was chosen for a 2425 pixel per mm scale, which translates to ~1000 μm^2. Different cut-off values will effect measurements by 1-2% for 100% increase or decrease in area cut-off.

2.10.5.4 Fraction Eutectic

Once the eutectic black particles (eutecticα) are eliminated, and a binary image has been created, then fraction primary $\alpha$and fraction eutectic can be calculated. The simplest way is given a 'Region of Interest', calculate the mean intensity value. Particle's values are 255 for white or 0 for black. Therefore, [mean intensity]/255 = [fraction alpha] and 1-[fraction dendrite] = [fraction eutectic]. Then from the lever rule, $C_o^* = f_e * (C_B^\alpha - C_B^\beta) + C_B^\beta$, where C_o^* is the local average composition, and f_e is fraction eutectic. As this process is standard from image to image, a macro was written for ImageJ to process a batch of images.

2.10.5.5 Core Fraction Eutectic

In the same manner, fraction eutectic could be calculated for any desired region of interest within an image. To illustrate steepling along the walls, the 'core' of the sample was measured separately to compare to the entire-cross section. The core was defined as a region $2/3^{rd}$the diameter of the smaller cross-section, and is consistent throughout the length. This small cross-section is measured by first selecting a 'Region of Interest', then subtracting the entire background.

CHAPTER III

RESULTS AND DISCUSSION

3.11 MICROSTRUCTURES: SPEED AND CROSS-SECTION CHANGE

3.11.1 LOW GROWTH SPEED AND CROSS-SECTION CONTRACTION

Figure 16shows a longitudinal section through the 11-17-2L sample, and also several transverse images at the locations indicated in the figure. These figures correspond to a low growth speed (10 μm/s). The hot end of the sample is indicated by an arrow above the longitudinal section. The transverse images are all views of the sample as seen from the "hot end". Heavy primary dendrite steepling in the middle of the sample is usually seen in all the cross-sections. Only eutectic regions (free of primary α phase) exist along the sample outer periphery on one side of the sample length. There is a concentration of "eutectic" only region at the shelf-top immediately prior to the cross-section decrease. Some of the primary α dendrites appear to continue on through the section decrease. Within the smaller diameter, immediately after contraction, the transverse and longitudinal images show primarily dendrite α phase. By the end of sample 11-17-2L (+13 mm), steepling again appears along the upper walls.

Dendrites have large diameter trunks and long arms. Only 6 to 7 dendrites exist across the large diameter transverse images, and are also clustered throughout the sample length. Transverse images show an apparent decrease in trunk spacing and arm length within the contracted diameter. Shortly after arm growth, globular tertiary branches densely fill the interdendritic region.

Trunk orientations are not well-aligned across the sample cross-section, as the longitudinal image shows divergent and convergent growth of α primary dendrites. This pattern is not affected by the cross-section contraction. Arm orientation is not completely consistent in the transverse images; occasionally side-branches do not grow perpendicular to one another. These patterns are not affected by the cross-section change.

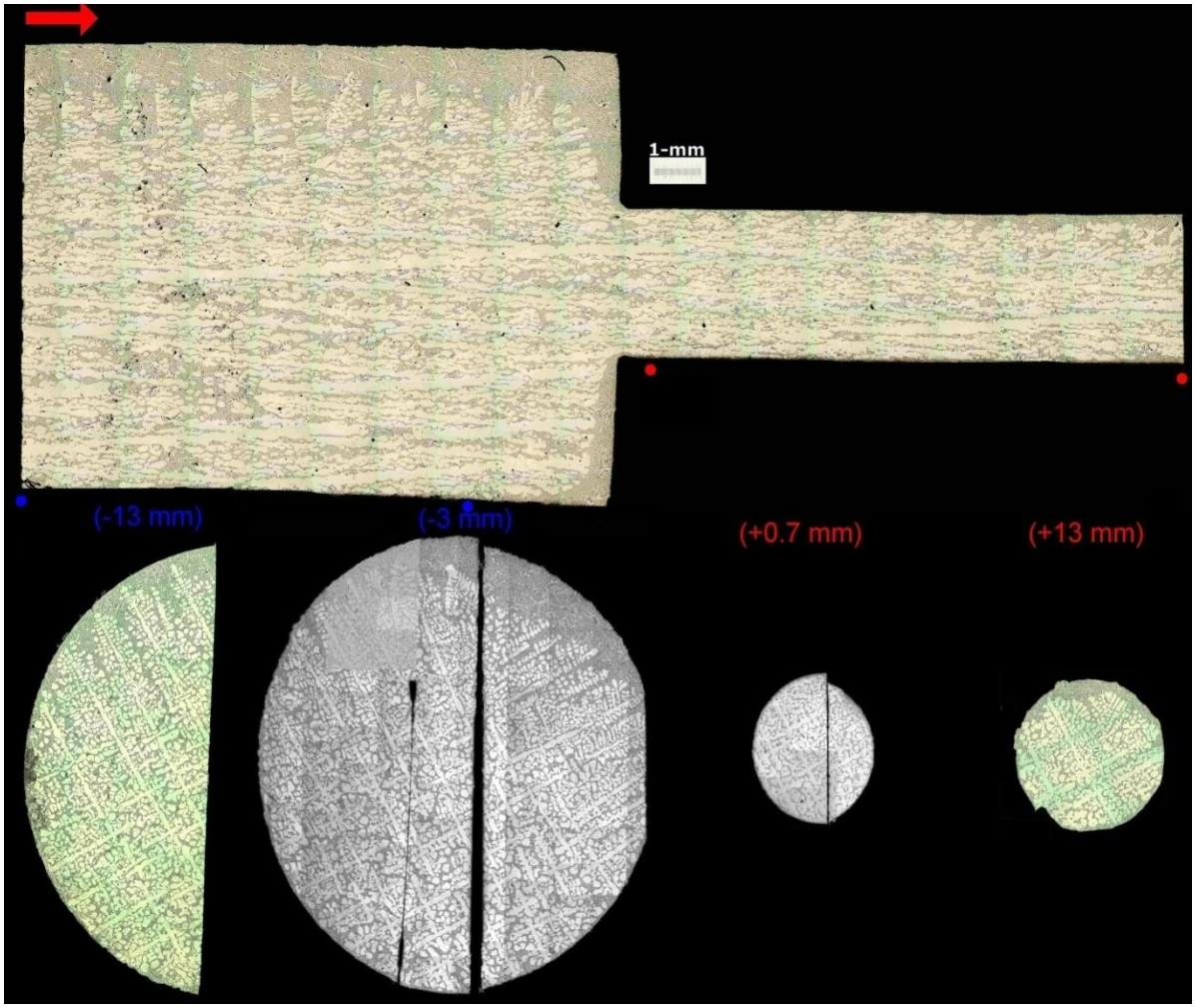

Figure 16: Longitudinal image of 11-17-2L with transverse images corresponding to -13, -3, 0.7, and 13 mm from the cross-section change (oriented to growth direction).

3.11.2 LOW GROWTH SPEED AND CROSS-SECTION EXPANSION

In Figure 17 is the image of longitudinal sample 11-17-5R with four select transverse images corresponding to locations indicated in the figure. 11-17-5R was grown at low growth speed (10 μm/s), with the images captured on the hot-end of the cross-section expansion. Steepling exists along the walls in the smaller diameter before cross-section expansion, and also in the larger diameter after the expansion.

Trunk diameter remains large throughout the length. Immediately following the expansion the transverse image (+2.5 mm) shows a low count of primary dendrites, with long arms, clustered in the center. By the end of 11-17-5R (+11. 1 mm), the primary dendrites have formed an evenly spaced well-aligned array; this is illustrated in both the transverse and the longitudinal images. Dendrite count appears to be higher near the end of 11-17-5R, even higher than the original seed in 11-17-2L. Tertiary globular branches fill the interdendritic region along the length. Also noticeable is the existence of a "primary α rich" region just after the cross-section expansion; here there is hardly any eutectic, except towards the very end of the shelf.

Within the smaller diameter and immediately following expansion, arm orientation is not consistent among the array. However, by the end of 11-17-5R, arms and trunks have formed a well-aligned array.

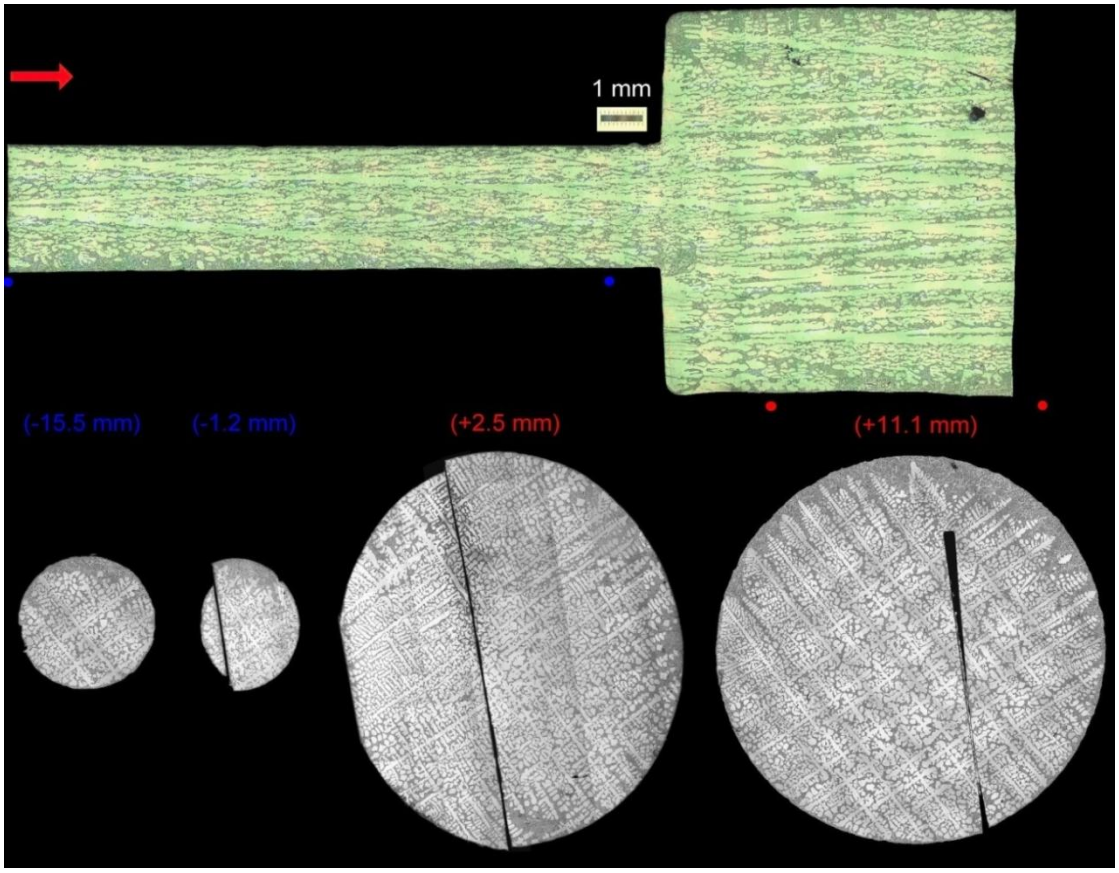

Figure 17: Longitudinal image of 11-17-5R with transverse images corresponding to -15.5, -1.2, 2.5, and 11.1 mm from the cross-section change (oriented to growth direction).

3.11.3 HIGH GROWTH SPEED AND CROSS-SECTION CONTRACTION

In Figure 18 is the image of longitudinal sample 12-14-2R with select transverse images all corresponding to a high growth speed (29.1 μm/s) on the cold-end of the cross-section change. In contrast to the 11-17-11 sample, only light-steepling exists at this growth speed. Presence of eutectic-only regions near the sample outer region and at the shelf of the larger cross-section portion (just before contraction) seen at 10 μm s^{-1} (Figure 14) is not seen at 29.1 μm s^{-1}. Dendrite α phase fills the walls immediately following the contraction.

Dendrites have narrower trunks, with short well-defined secondary and tertiary arms. Trunks are tightly packed, with as many as 10 to 12 across the large diameter and evenly spaced throughout a cross-section. After arm growth, perpendicular tertiary arm growth does not fill the interdendritic region leaving grain separation well visible. After contraction, primary dendrite alignment appears to improve. Secondary arms continue to grow perpendicular to one another after contraction.

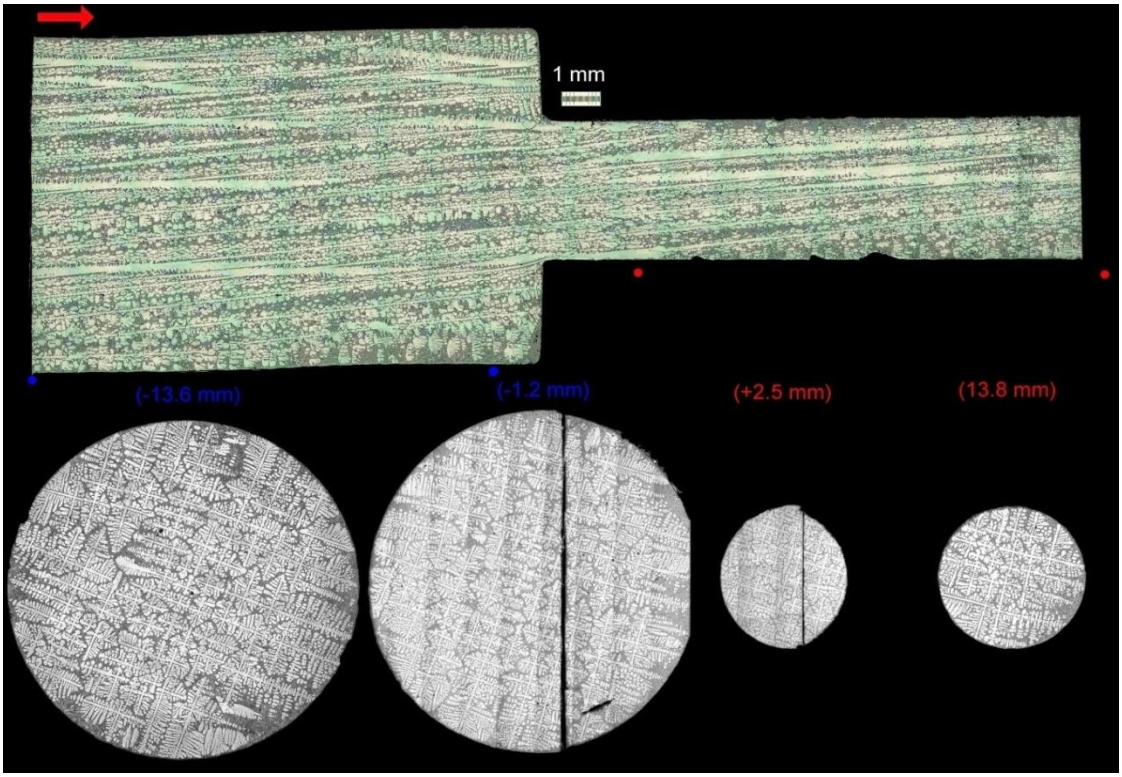

Figure 18: Longitudinal image of 12-14-2R with transverse images corresponding to -13.6, -1.2, 2.5, and 13.8 mm from the cross-section change (oriented to growth direction.)

3.11.4 HIGH GROWTH SPEED AND CROSS-SECTION EXPANSION

In Figure 19 is the image of longitudinal sample 12-14-4R with select transverse images all corresponding to a high growth speed (29.1 μm/s) on the hot-end of the cross-

section change. Light steepling is only present 15 mm after cross-section expansion. Alpha phase fills the diameter immediately following the expansion, similar to 11-17-5R.

Trunk diameter remains narrow, with short arm length except immediately after diameter expansion, again similar to 11-17-5R. Primary dendrite spacing appears to increase after expansion. Secondary and tertiary arm growth remains well defined, with low packing ratio between dendrites. The primary dendrites appear to maintain their alignment along the sample length, even though they are not aligned with the heat-extraction direction, i.e., parallel to the directional solidification direction.

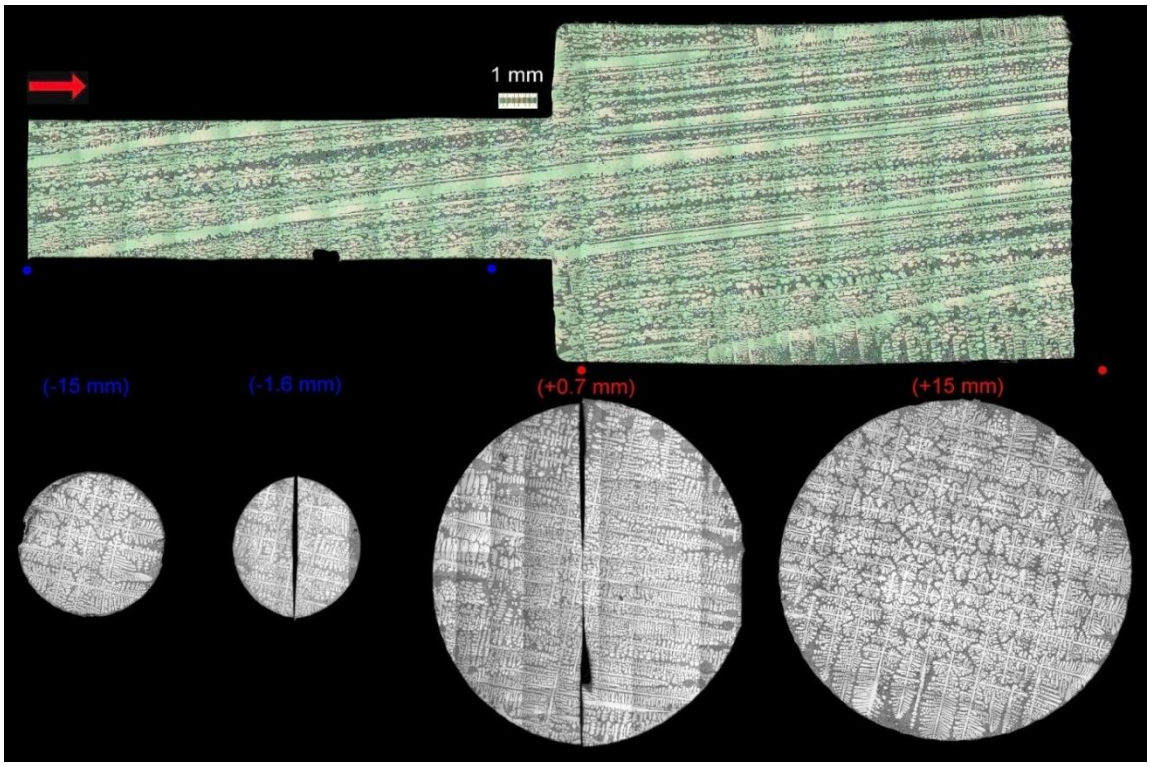

Figure 19: Longitudinal Image of 12-14-2R with transverse images corresponding to -15, -1.6, 0.7, and 15 mm from the cross-section change (oriented to growth direction.)

3.12 PRIMARY SPACING DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE

3.12.1 DISTANCE FROM CONTRACTION

Figure 20 illustrates the growth speed dependence of primary dendrite spacing as determined from $A/\sqrt{(N-1)}$ as a function of distance from the cross-section decrease. However it should be noted that for such a low count of primary dendrites (~3 to 4) on a sample cross-section, large errors are possible, i.e. in the 3.2 mm diameter portions of the sample. This would be especially true for the sample grown at lower growth speed of 10 μm s^{-1}(11-17-2 or 11-17-5), where primary spacing is nearly half as large as the crucible diameter. Therefore, any meaningful conclusions about the effect of cross-section change on the primary spacing should be limited to the sample grown at 30 μm s^{-1}. For this sample the primary arm spacing appears to decrease immediately after the cross-section decrease (Figure 19). Arm spacing then widens to its original distance after about 2 mm of further directional solidification (Compare data from Figure 19 and Figure 20 for 12-14-2 and 12-14-5 samples). Similar trend is seen after a cross-section increase for this sample as indicated in Figure 20. Also as the growth speed increases from 10 to 29.1 μm s^{-1}, the primary spacing decreases, as is expected from theoretical models [16, 2, 5, 4, 8].

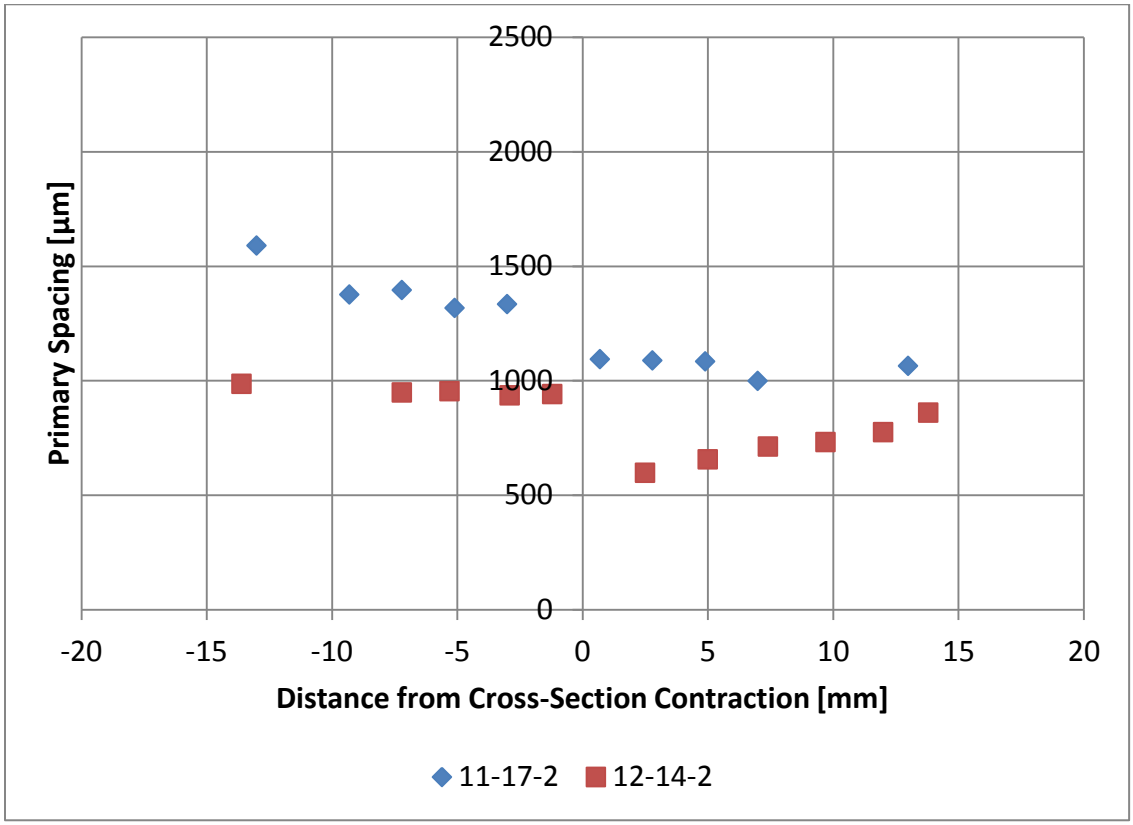

Figure 20: Primary spacing [µm] versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red)

3.12.2 DISTANCE FROM EXPANSION

Figure 21 shows the variation in primary spacing at two growth speeds as the solidification front enters into the sample's cross-section expansion region. Again, spacing for the faster growth speed remains consistently lower in comparison to that at the slower growth speed. 12-14-4, the sample grown at 29.1 µm s^{-1}, appears to show a sudden spacing increase immediately after the cross-section expansion. This correlates with the previous microstructure examination (See 3.11.4), where it was observed that mainly primary α dendrites are seen after expansion (Figure 19). Following is a large sharp decrease in the spacing, followed by a slow increase to approach steady-state spacing expected under these

growth conditions (see Section 1.3.) The 11-17-5 sample, grown at 10 μm s^{-1} shows a similar trend, but because of the large scatter in the data before the section increase, no meaningful conclusions should be drawn.

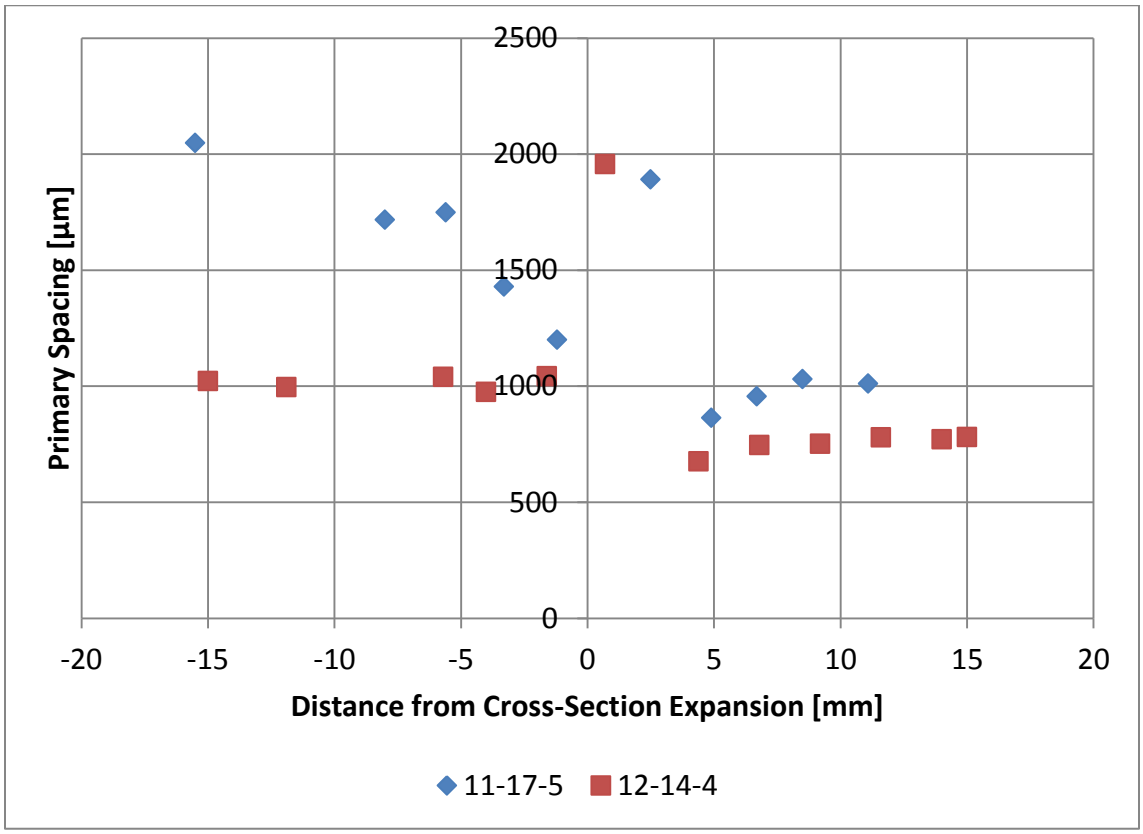

Figure 21: Primary spacing [μm] versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 μm/s (red)

3.13 NEAREST NEIGHBOR SPACING DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE

3.13.1 DISTANCE FROM CONTRACTION

Figure 22 plots nearest neighbor spacing as a function of distance from a cross-section contraction for both growth speeds. Unlike primary spacing, nearest neighbor spacing will account for clustering of dendrite centers, and ignores any relation to the walls. Overall nearest neighbor spacing is lower than primary spacing for both growth speeds, which expectedly indicates a degree of clustering. Similar to primary spacing, nearest neighbor spacing is lower for the faster growth speed, except for one point. *Unlike* primary spacing, nearest neighbor spacing does not experience any significant drop in spacing at the cross-section change, and remains constant throughout, especially for the faster growth speed.

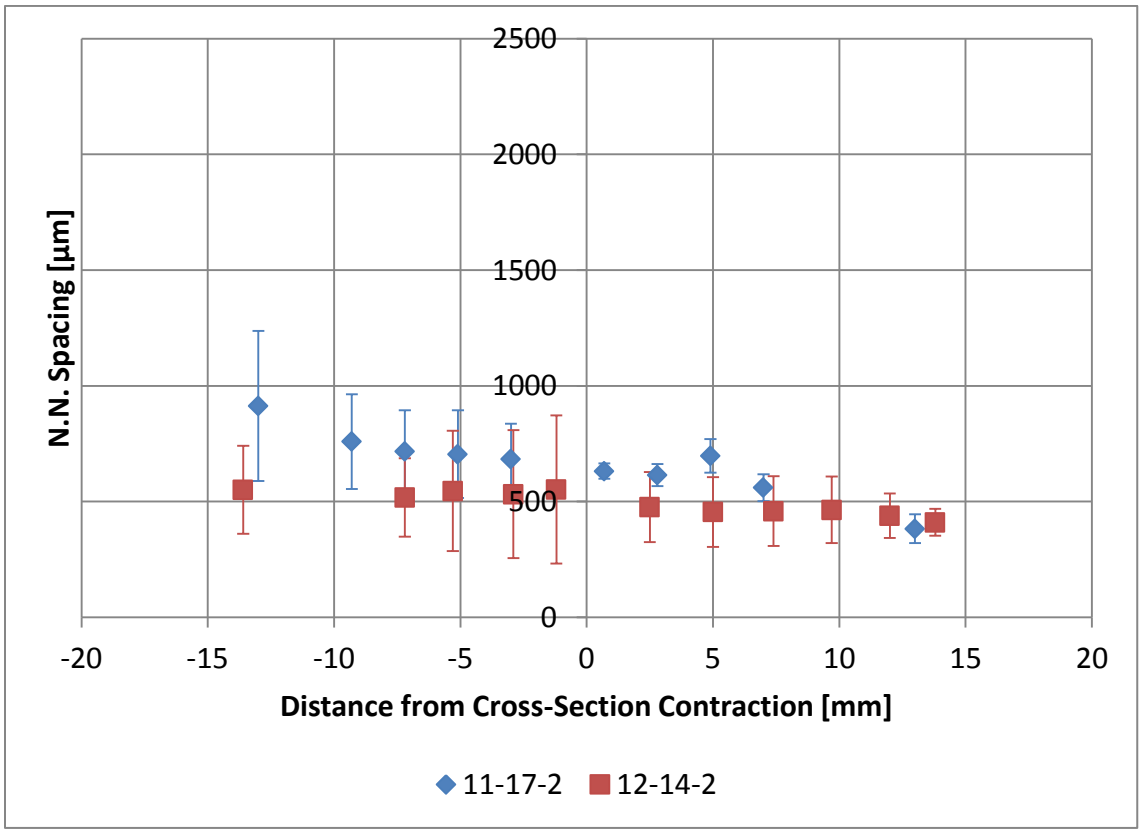

Figure 22: Nearest neighbor (N.N.) spacing [µm] versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red)

3.13.2 DISTANCE FROM EXPANSION

Figure 23 illustrates nearest neighbor spacing dependence on distance from the expansion in cross-sectional area. Similar to the primary spacing, there is an increase immediately following the point of expansion followed by a drop. Since nearest neighbor spacing does not account for distance from the walls, this indicates that spacing increases only between dendrite centers.

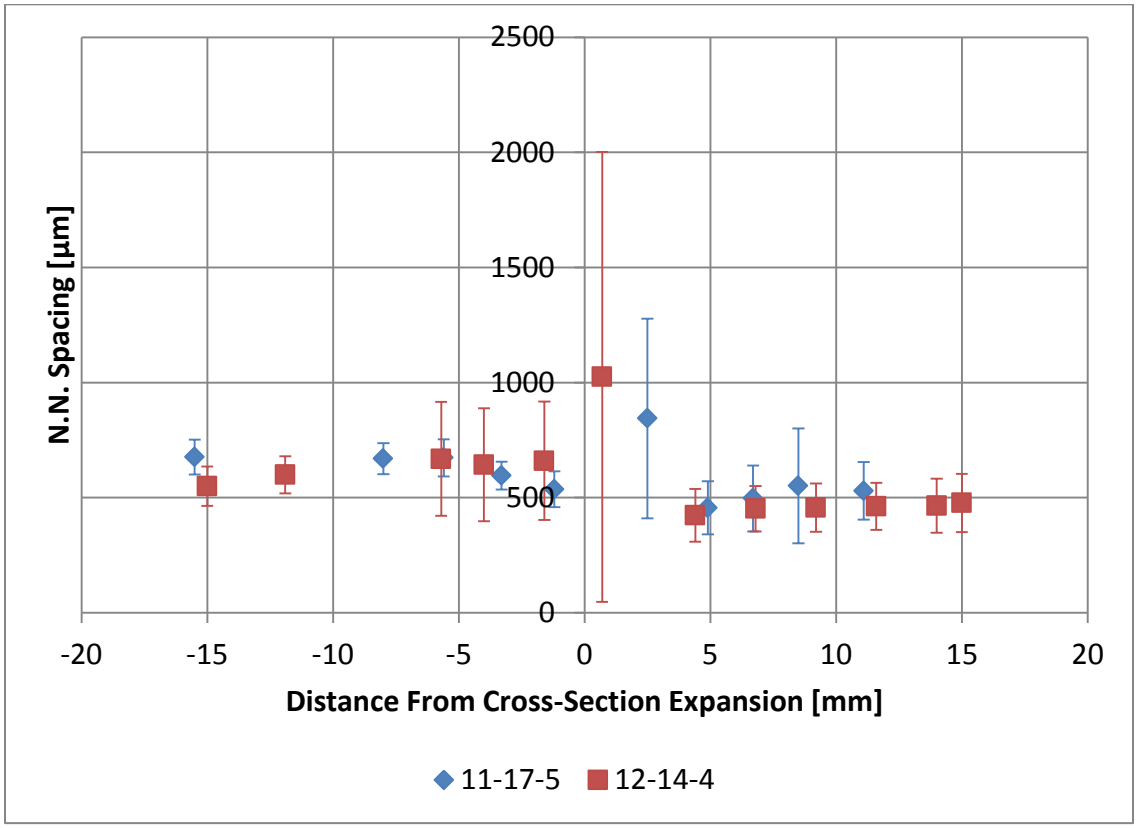

Figure 23: Nearest neighbor (N.N.) spacing [µm] versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red)

3.14 NEAREST-NEIGHBOR SPACING TO PRIMARY SPACING RATIO DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE

3.14.1 DISTANCE FROM CONTRACTION

Figure 24 plots the nearest neighbor spacing to primary spacing ratio as a function of distance from the cross-section change for both growth speeds. As indicated by the figure, except for the seed, the slower growth speed is always more clustered than the faster growth speed. In the smaller diameter, counts can be as low as 2 to 4 dendrites, so large fluctuations are expected and will be ignored.

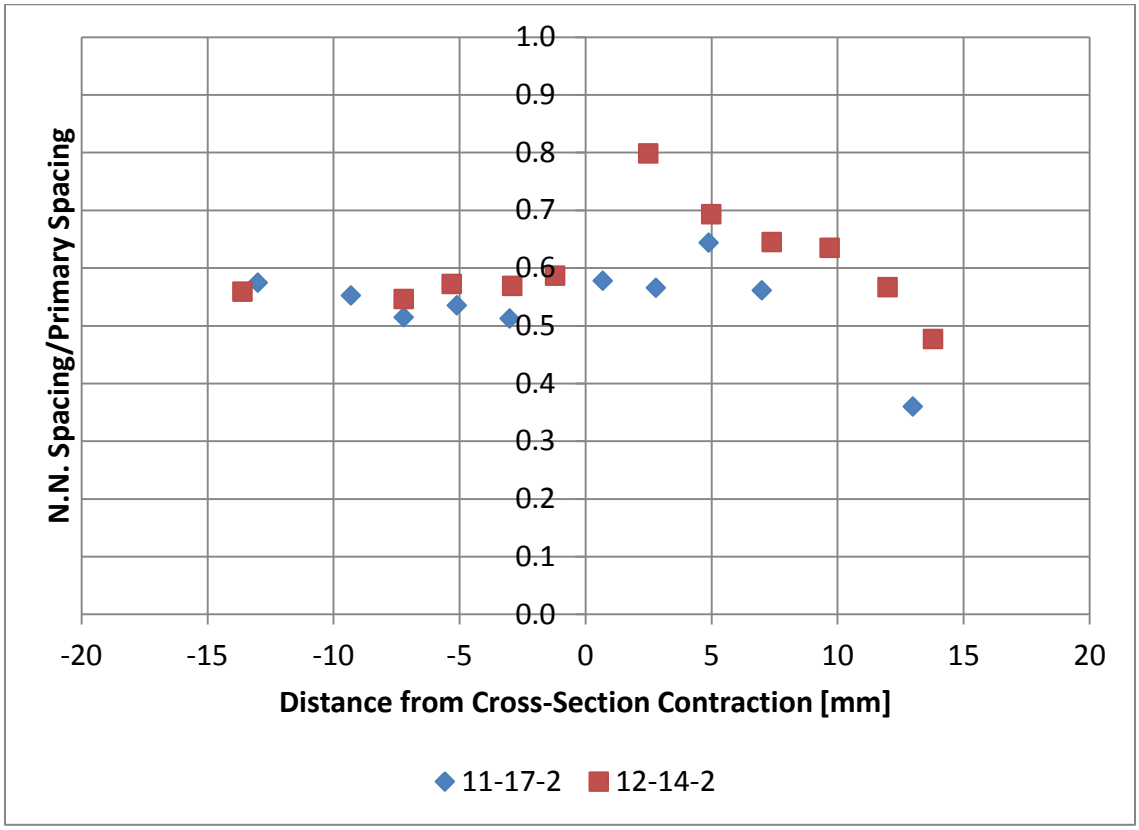

Figure 24: Nearest neighbor spacing to primary spacing ratio versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.14.2 DISTANCE FROM EXPANSION

Figure 25 plots the nearest neighbor spacing to primary spacing ratio as a function of the distance from cross-section expansion for both growth speeds. Again, this ratio shows that steepling is consistently more likely for the lower growth speeds. Note that immediately following the expansion, the sample grown at a faster growth speed experiences a minimum, indicating the formation of new grains.

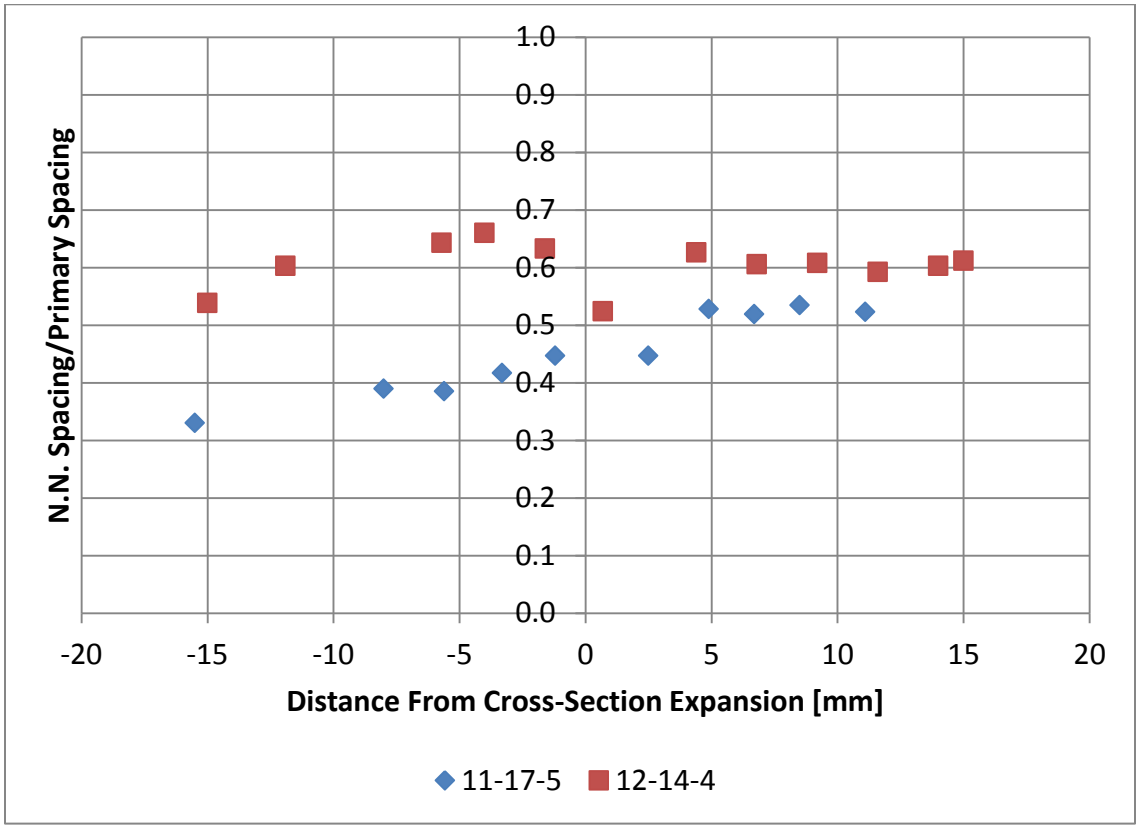

Figure 25: Nearest neighbor spacing to primary spacing ratio versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

Table 4 lists the theoretically predicted primary spacings, based on Hunt-Lu model [2], the experimentally measured spacings ($A/\sqrt{(N-1)}$), and the experimentally measured nearest neighbor spacing. The theoretical calculations are based on the thermal gradients at the liquidus temperature as recorded by the various thermocouples during these experiments. The physical properties used in these calculations are also indicated below. The table examines the primary arm spacings measured from the 9.5 mm diameter portion of the samples from the two ends, i.e., from sections at least 1.1 cm away from the cross-section change, as these would not be affected by the section change and would represent array

morphology under steady state growth conditions. As the growth speed increases the primary spacing decreases, as also predicted from the model. However, the measured primary spacings are 46 to 78% larger than the predicted values. The model predictions, as shown in Table 4, are in much better agreement with the experimentally measured nearest neighbor spacings rather than spacing. It should be noted that the "model" predicts the nearest neighbor spacing and not the spacing measured by $A/\sqrt{(N-1)}$, yet very often people compare their primary spacing data measured from the $A/\sqrt{(N-1)}$ values to the predictions from the theoretical models.

Table 4: Comparison of primary dendrite spacing predicted from Hunt-Lu model (using thermal gradient at the tip: G_l)*with the experimentally observed values.

ID	TC	G_l [Kcm^{-1}]	R [μms^{-1}]	r_t [µm]	Spacing [µm]	Sample ID	Measured Primary spacing, [µm]	Measured nearest neighbor spacing, [µm]
11-7-11	TC2	26.1	10	8.35	907.01	11-17-2b	1589.8	913.3 ±303
	TC3	32.52	10	8.35	807.52			
	TC4	44.4	10	8.35	688.91	11-17-5T	1011.6	529.1± 125
12-14-11	TC1	33.5	29.1	4.84	553.83	12-14-1T	986.4	551.2± 189
	TC3	35.3	29.1	4.84	539.03			
	TC4	44.4	29.1	4.84	479.32	12-14-4T	780.4	477.5±126.6

Table 5: Physical properties used in Hunt-Lu Calculations.

Alloy	m_l, K/wt%	k	D_l, cm^2/s	Γ, µmK	Tm, °C
Al-7%Si	6.31	0.1	$4.3X10^{-5}$	0.196	660.37

3.15 TRUNK DIAMETER DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE

3.15.1 DISTANCE FROM CONTRACTION

Figure 26 shows the primary dendrite trunk diameter dependence on distance from cross-section contraction for the growth speeds of 10 and 29.1 μm/s as measured by the earlier described image analysis (see Section 2.10.2) on the transverse sections. Mean with standard deviation error bars are plotted for both growth speeds. The mean trunk diameter decreases with increasing growth speeds. The figure indicates that primary dendrite trunk diameter remains unaffected by the cross-section change induced convection during cross-section contraction.

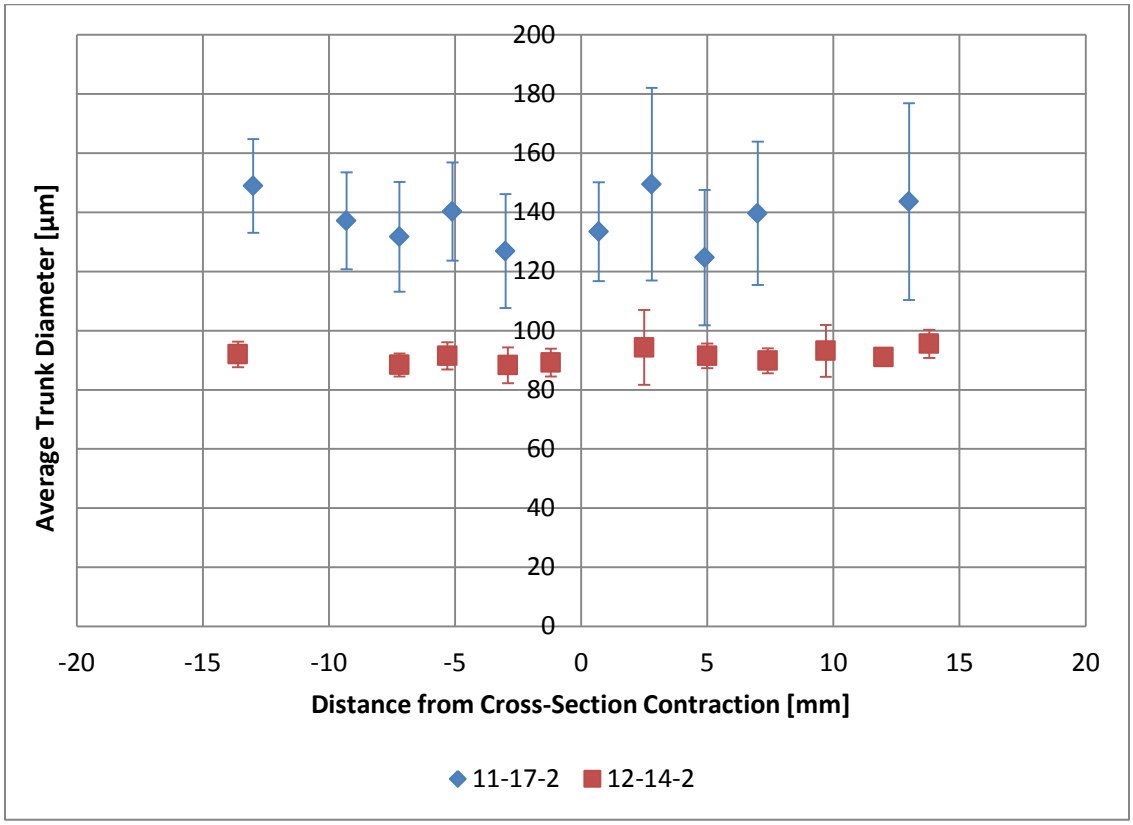

Figure 26: Average trunk diameter [µm] versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.15.2 DISTANCE FROM EXPANSION

Figure 27 shows the primary dendrite trunk diameter dependence on distance from cross-section expansion for 10 and 30 µm s^{-1}speeds. As mentioned above, no trend can be detected from this data concerning the effect of cross-section increase on the primary dendrite trunk diameter.

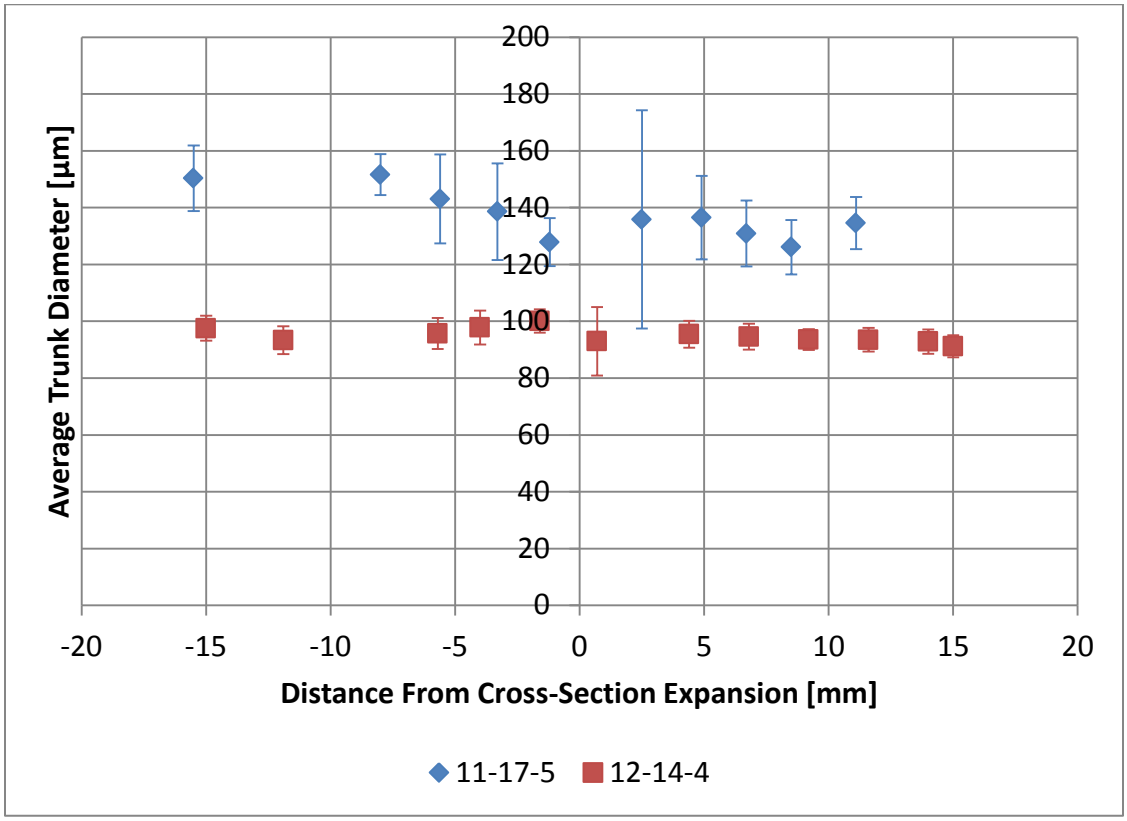

Figure 27: Average trunk diameter [µm] versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.16 SECONDARY ARM LENGTH DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE

3.16.1 DISTANCE FROM CONTRACTION

Figure 28 shows secondary arm length dependence on growth speed and distance from cross-section contraction. As shown by Pakiru [24], arm length is dependent on growth speed, though weakly. According to Pakiru [24], a 20 times increase in growth speed only corresponded to ~200 µm reduction in average arm length. As a result of this

study, arm length is only weakly dependent on growth speed, with scatter nearly as large as the mean. However, there is a strong reduction of average arm length after the cross-section decrease for both of the growth speeds: ~700 µm for the slower growth speed sample (11-17-2) and ~500 µm for the faster growth speed (12-14-2). This could be caused by the decreased spacing, and therefore decreased room for arm growth.

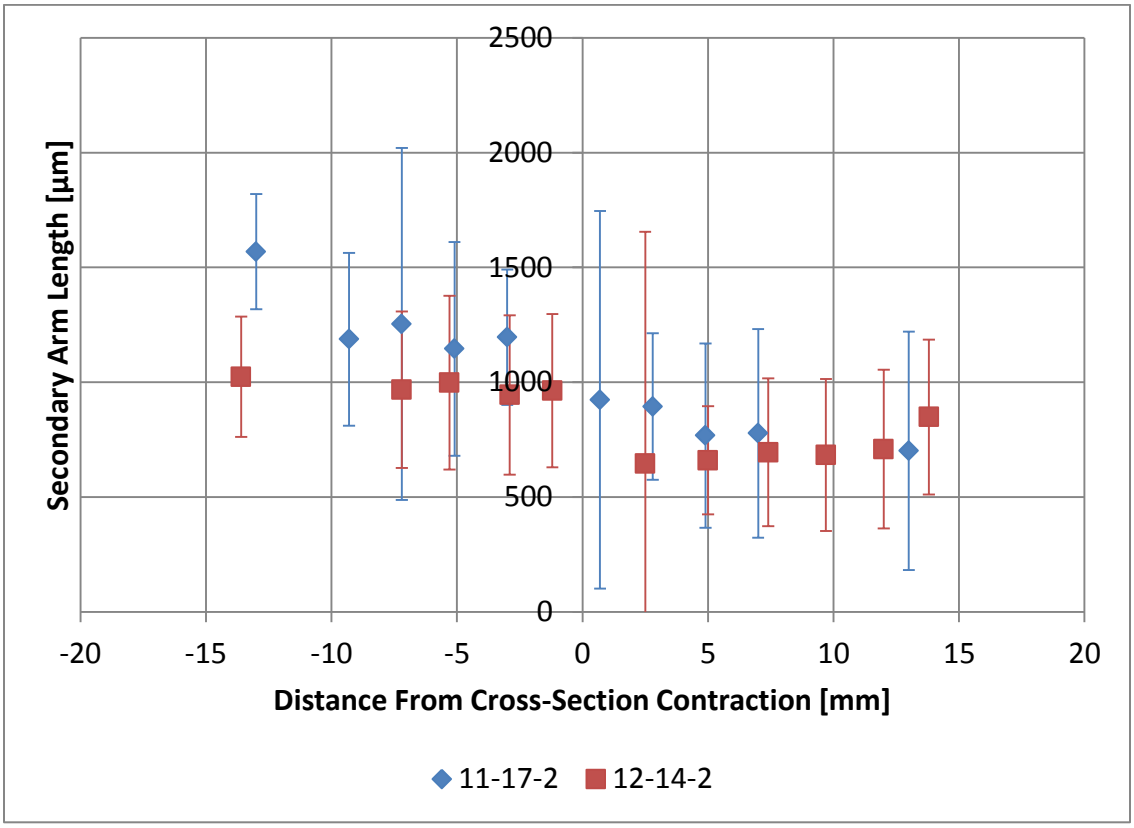

Figure 28: Average arm length [µm] versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

If arm length were entirely dependent on spacing, the arm length to spacing ratio would be nearly unity. Figure 29 plots arm length to primary spacing ratio for both growth speeds. At faster growth speed the arm length is nearly one, largely unaffected by the cross section change. However, sample 11-17-2 (a lower growth speed) experiences a significant

gradual decrease in arm length to spacing ratio at the cross-section contraction. This may be because at a lower growth speed the mushy-zone solidification time is larger allowing the convection more time to modify the mushy-zone morphology. Arm length to primary spacing ratio being less than unity is an indication of "blunter" dendrite trees, which may be a convection related phenomenon.

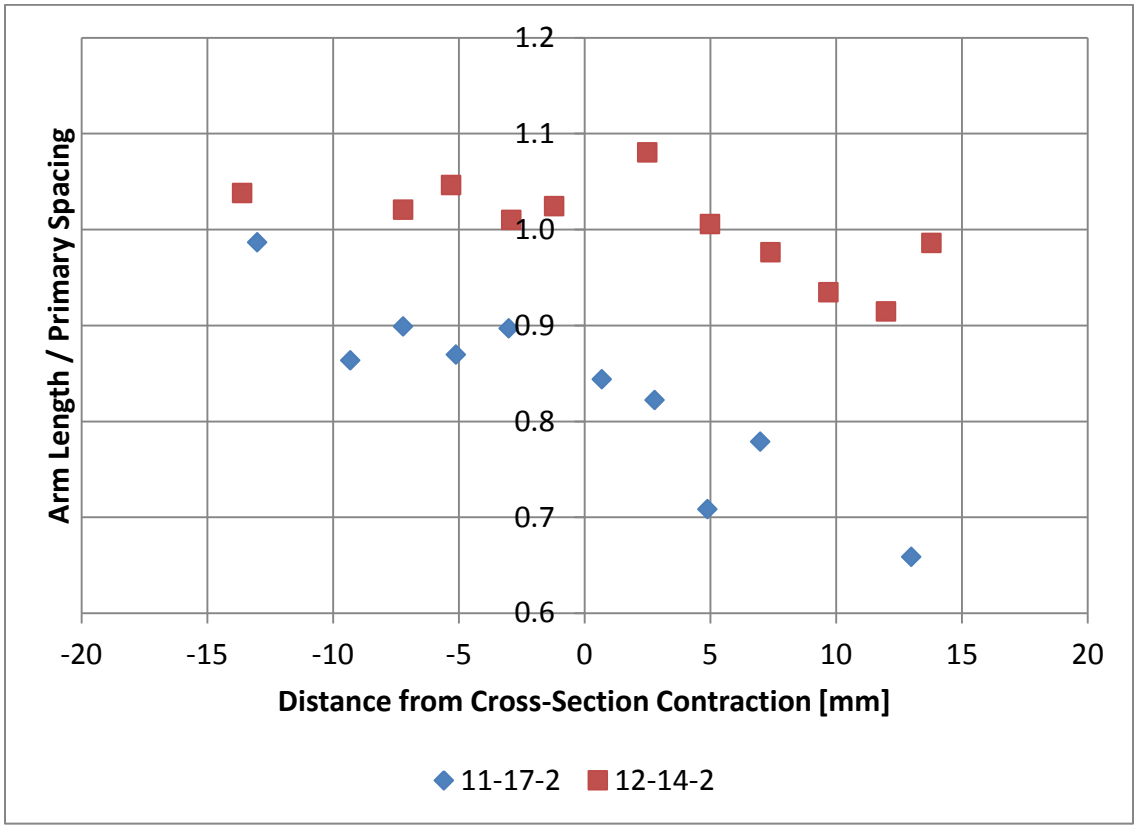

Figure 29: Arm length to primary spacing ratio versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.16.2 DISTANCE FROM EXPANSION

Figure 30 shows arm length dependence on growth speed and distance from the cross-section expansion. The figure similar to Figure 28, is also weakly dependent on growth speed. However, immediately following cross-section expansion, there is a 700 µm

spike in arm length, well outside the noise. This can be correlated with the observed longitudinal section morphology of dendrites as they enter the larger cross-section (Figure 17, Figure 19). Side branches of the primary dendrites emerging into the larger cross-section portion will spread sideways and grow. From these secondary branches, tertiary branches form and grow parallel to the growth direction becoming primary dendrites.

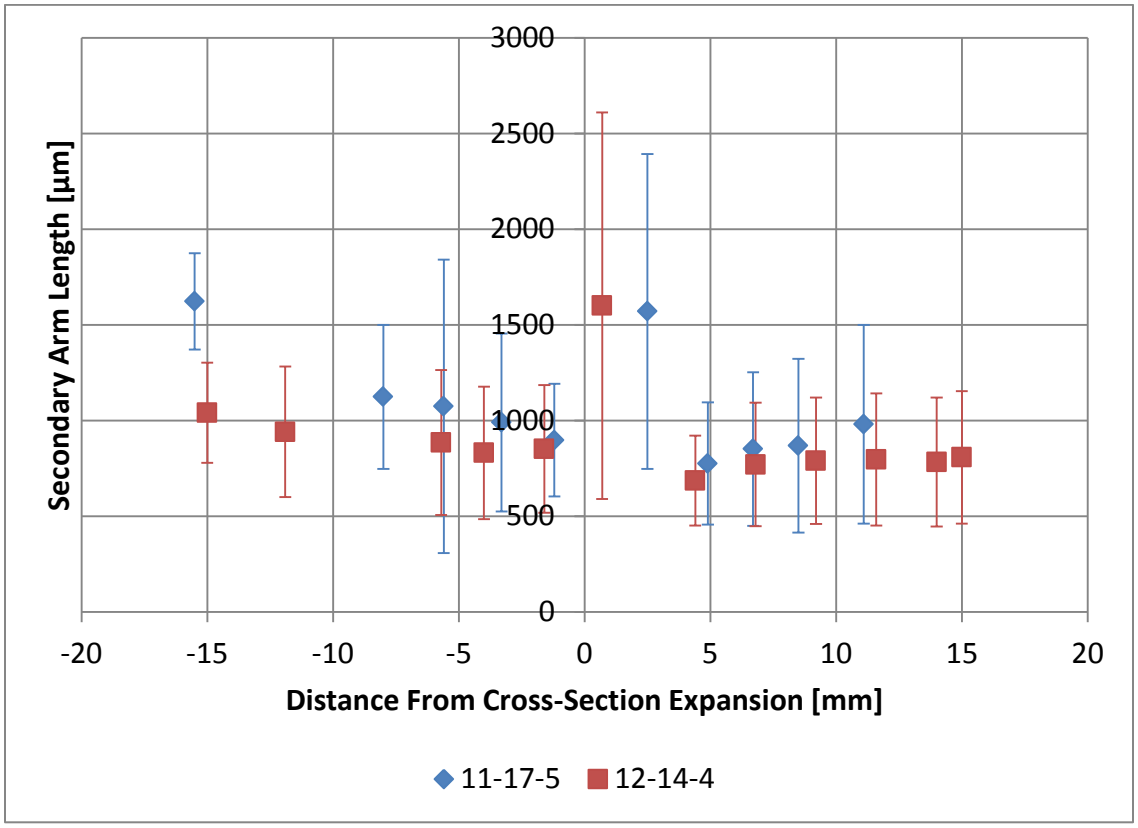

Figure 30: Average arm length [µm] versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

Figure 31 shows the arm length to primary spacing ratio dependence on distance from a cross-section expansion. In the case of expansion, arm length actually remained relatively consistent for both growth speeds not including the large spike. However despite the spike and consistent numbers, arm length to primary spacing ratio *drops* near

expansion, for both the growth speeds. In this case, large increases in primary spacing followed by large decreases accounts for this drop.

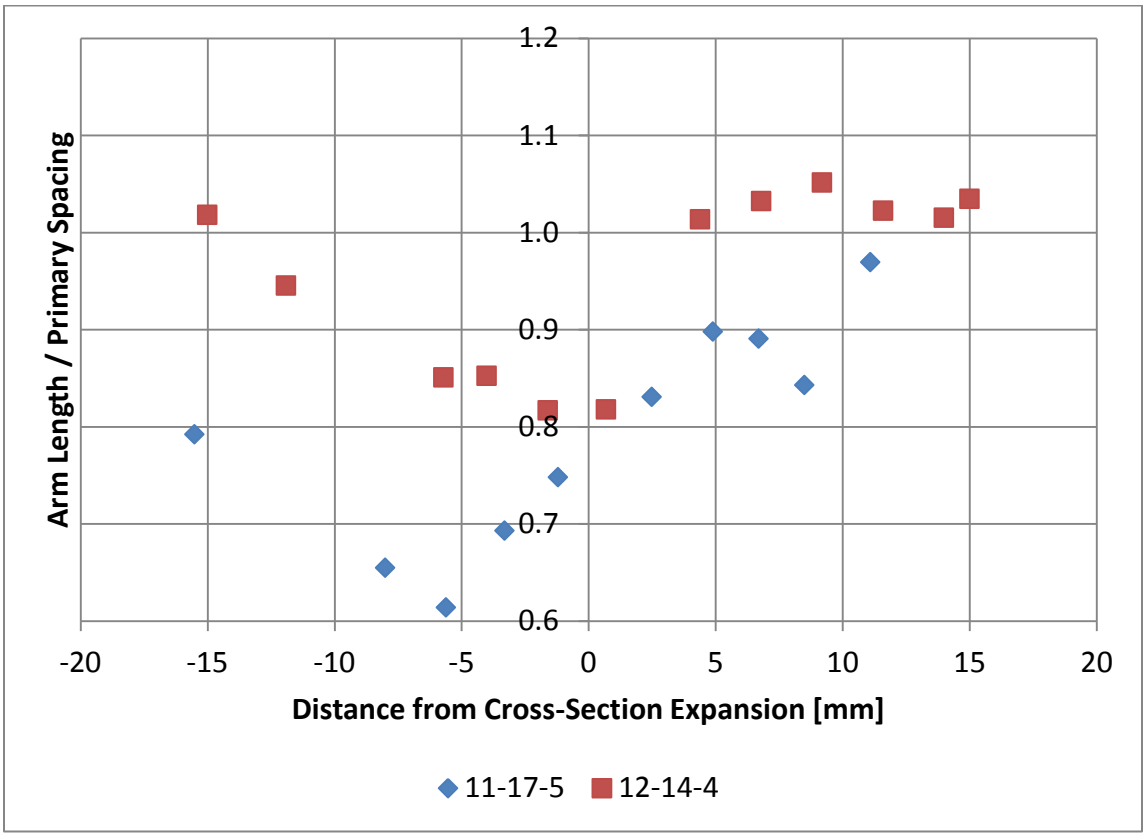

Figure 31: Average arm length to primary spacing ratio versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.17 ARM ORIENTATION DEPENDENCE ON GROWTH SPEED AND DISTANCE FROM CROSS-SECTION CHANGE.

3.17.1 DISTANCE FROM CONTRACTION

Figure 32 plots average side-arm (secondary dendrite) orientation as a function of distance from the cross-section contraction for both the growth speeds with error bars representing standard deviation. It is not important that the relative orientation values are

different for the two growth speeds, as this is dependent on at what orientation the original image was captured. Average side-branch orientation remains constant along the directional solidification length. This indicates that mostly the grains in the seed portion have continued to grow along the entire sample length, without significant numbers of new grains forming. Scatter for both growth speeds is extremely low, varying at maximum by 2 to 3 degrees.

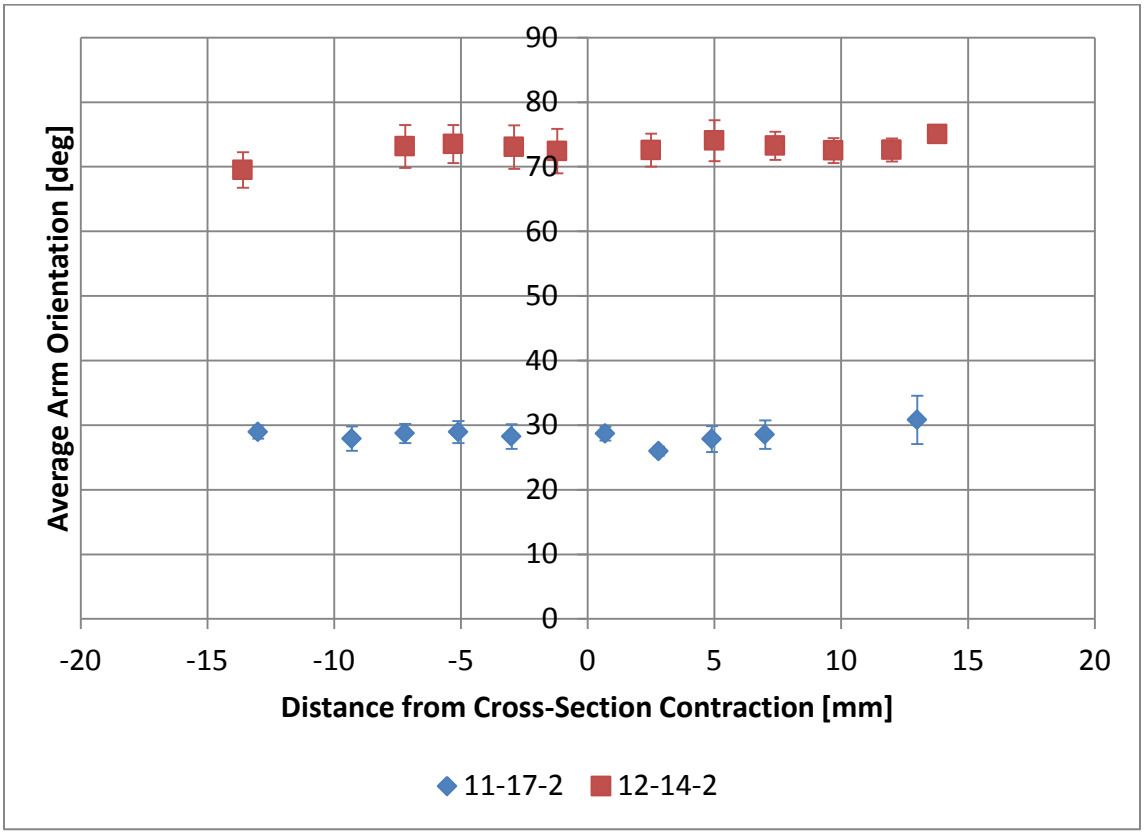

Figure 32: Average arm orientation [deg] versus distance from cross-section contraction [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.17.2 DISTANCE FROM EXPANSION

Figure 33 plots the average side-arm orientation as a function of distance from the cross-section expansion for both growth speeds with error bars representing standard

deviation. Again it is not important that the relative orientations are different between growth speeds. In the case of expansion, one important feature to note is the sharp increase in scatter immediately following the cross-section change for the faster growth speed. These large scatter values indicate that a significant number of spurious grains formed as the mushy-zone entered the larger cross-section portion of the ingot.

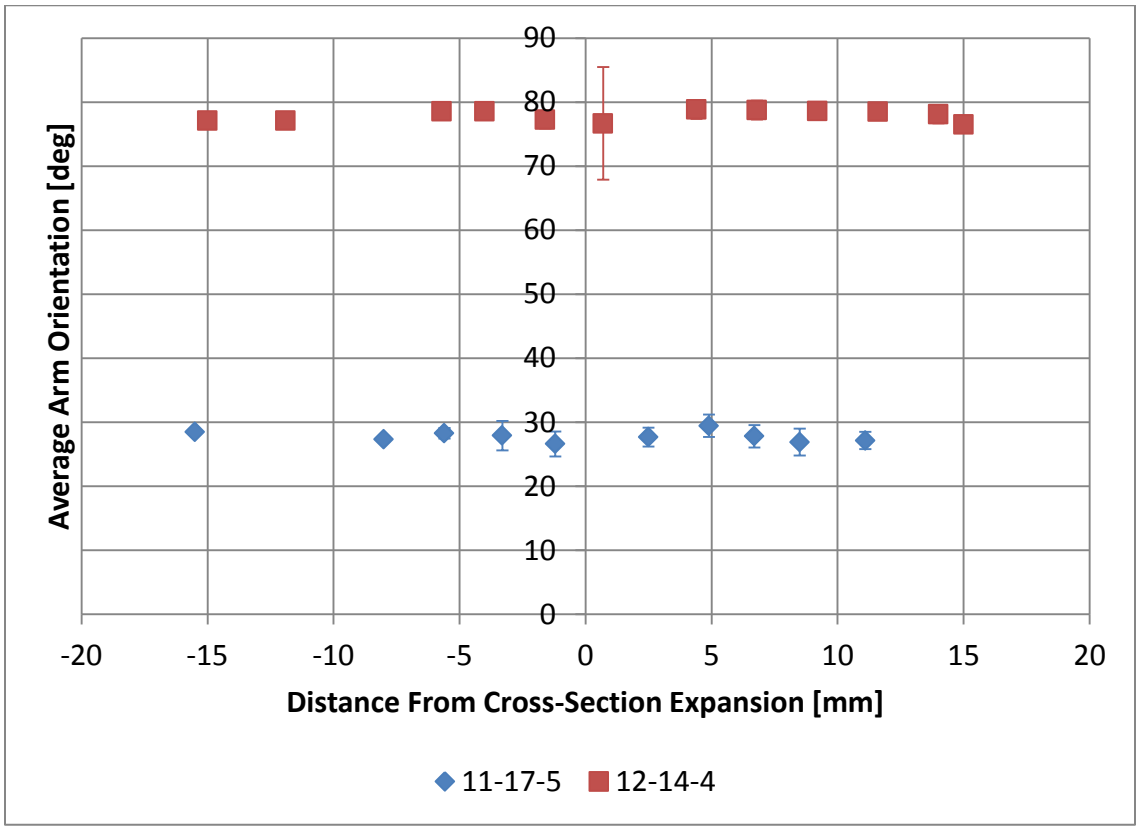

Figure 33: Average arm orientation [deg] versus distance from cross-section expansion [mm] for growth speeds of 10 (blue) and 29.1 µm/s (red).

3.18 FRACTION EUTECTIC DEPENDENCE ON GROWTH SPEED, RADIUS AND DISTANCE FROM CROSS-SECTION CHANGE

3.18.1 SAMPLE 11-17-11 CORE VERSUS ENTIRE CROSS-SECTION

Figure 34 graphs fraction eutectic dependence on distance from cross-section contraction for both the 'core' of the sample and the entire cross-section for the slower growth speed. Consistently the core-region has a lower fraction eutectic then the entire cross section, which indicates radial macrosegregation. Also, the fraction eutectic dips by 20% at the contraction, indicating that this pinch-point is mostly dendrite phase.

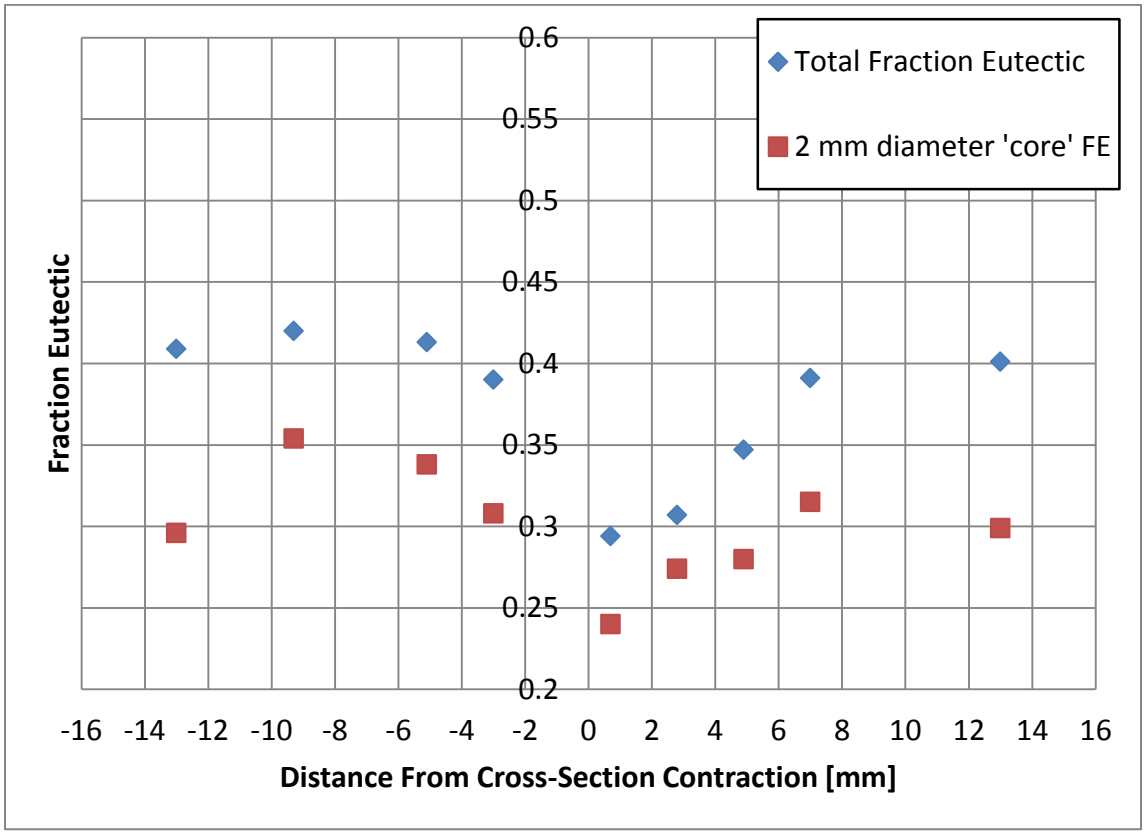

Figure 34: Fraction eutectic versus distance from cross-section contraction [mm] for growth speeds of 10 µm/s examined for an entire cross-section and the core of the cross-section.

Figure 35 graphs fraction eutectic dependence on distance from cross-section expansion for both the 'core' of the sample and the entire cross-section. Again, the core has a consistently lower fraction eutectic, indicating that it is mostly of the dendrite phase. Through expansion, fraction eutectic remains largely unaffected, with maybe a slight drop indicating more dendrite phase. There is a large drop in the 'core' region, indicating a large radial macrosegregation exists, with mostly eutectic phase near the walls.

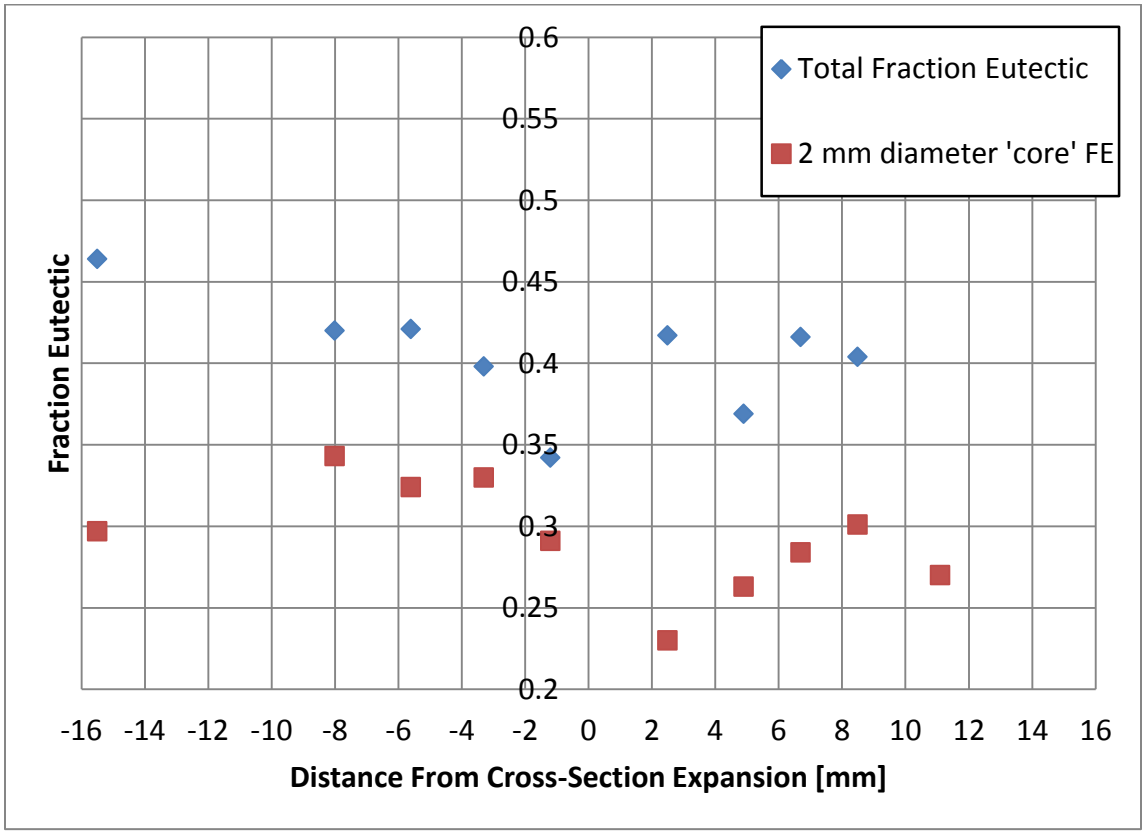

Figure 35: Fraction eutectic versus distance from cross-section expansion [mm] for growth speeds of 10 µm/s examined for an entire cross-section and the core of the cross-section.

3.18.2 SAMPLE 12-14-11 CORE VERSUS ENTIRE CROSS-SECTION

Figure 36 graphs fraction eutectic dependence on distance from cross-section contraction for both the 'core' of the sample and the entire cross-section for the faster growth speed. Again, consistently the core-region has a lower fraction eutectic then the entire cross section, which indicates radial macrosegregation. Also, the fraction eutectic dips by ~15% at the contraction, indicating that this pinch-point is mostly dendrite phase. Unlike the slower growth speed, hardly any macrosegregation exists at the pinch point, and is dendrite phase all the way to the walls. Also, sample 11-17-2 (slower) has an overall larger separation between total and core fraction eutectic then sample 12-14-2 (faster).

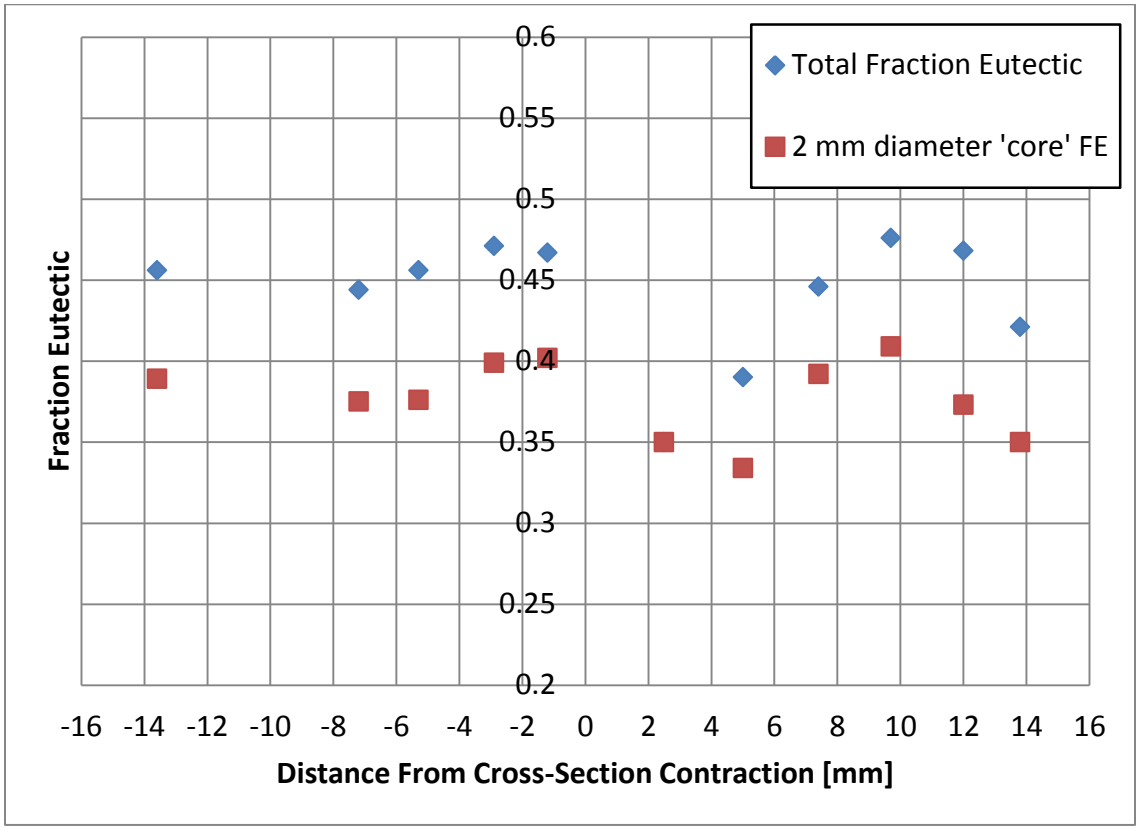

Figure 36: Fraction eutectic versus distance from cross-section contraction [mm] for growth speeds of 29.1 µm/s examined for an entire cross-section and the core of the cross-section.

Figure 37 graphs fraction eutectic dependence on distance from cross-section expansion for both the 'core' of the sample and the entire cross-section for the faster growth speed. Again, consistently the core-region has a lower fraction eutectic then the entire cross section, which indicates radial macrosegregation. Similar to sample 11-17-5, though not as severe, overall fraction eutectic remains largely unaffected by the cross section expansion, but core fraction eutectic drops. This large separation indicates dense dendrite phase along the walls immediately following expansion.

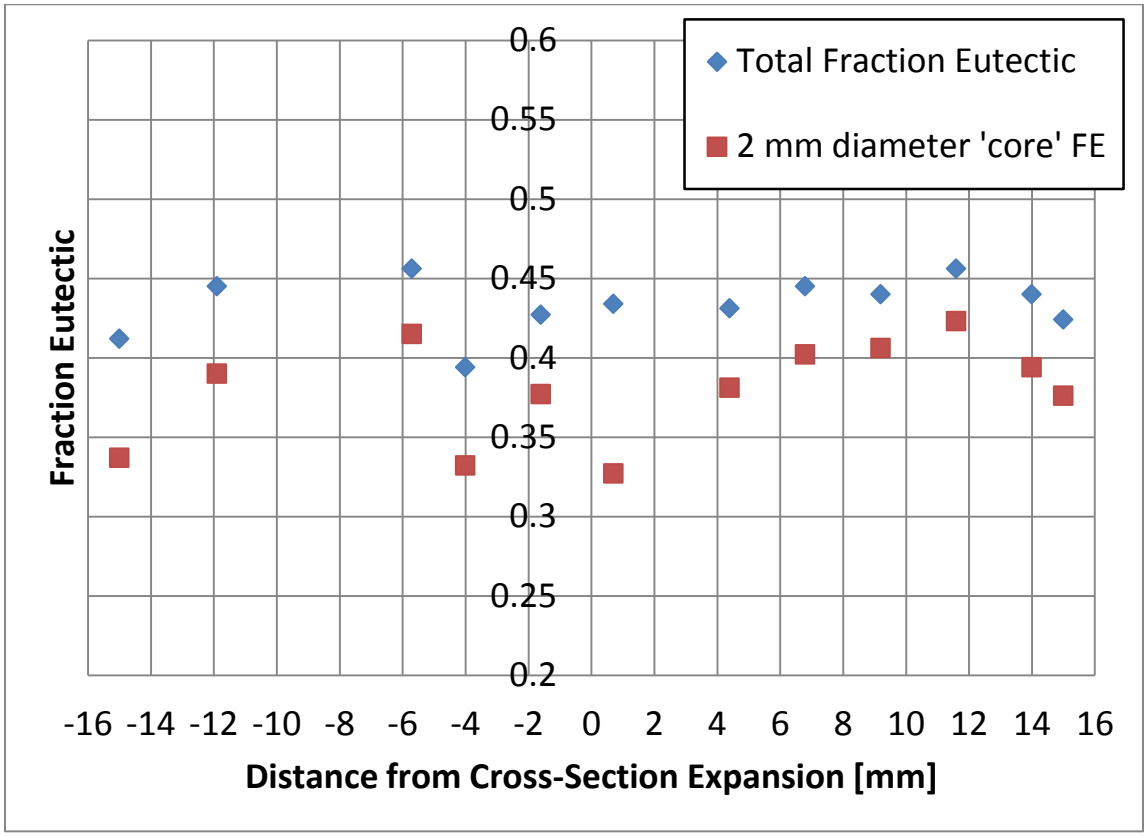

Figure 37: Fraction Eutectic versus Distance from Cross-Section Expansion [mm] for growth speeds of 29.1 µm/s examined for an entire cross-section and the core of the cross-section

CHAPTER IV

SUMMARY

Multiple conclusions can be drawn from this study on the directional solidification of Al-7 wt. pct. Si at 10 μm/s and 29.1 μm/s.

1 Increase in growth speed correlates to lowered primary spacing, trunk diameter, and arm length. This was to be expected from the Hunt-Lu Model [16]. The 3x increase in growth speed corresponded to ~50% reduction in spacing, ~25% reduction in trunk diameter, and ~10% reduction in arm length.

2 Increase in growth speed correlates to a decrease in radial macrosegregation and 'steepling'. This was shown by both nearest neighbor to primary spacing ratio, as well as core versus whole cross-section fraction eutectic. If nearest neighbor to primary spacing ratio is low, then dendrite centers are more clustered. This value was consistently lower for the slower growth speed. Also, the separation between 'core' versus entire cross-section for fraction eutectic averaged ~8-9% for 11-17-11, compared to only 4-6% for the faster growth speed. This is a stronger indicator of

3 Convection then nearest neighbor spacing because this takes into account secondary and tertiary arms, and not just the clustering of trunks. Though it is important that they both agree.

4 Cross-section contraction lowers arm length. This is expected from the decreased allowable space for growth.

5 Cross-section contraction induces radial macrosegregation immediately before the pinch-point for both growth speeds. This is observed clearly in longitudinal images, and confirmed by fraction eutectic data as well as arm length to primary spacing ratios.

6 Cross-section contraction has no effect on trunk diameter.

7 Cross-section contraction did not form spurious grains at these growth speeds

8 Cross-section expansion produces immediate spikes in arm length. This is to be expected through increased spacing.

9 Cross-section expansion induces radial macrosegregation and steepling. This was seen in both fraction eutectic data, as well as nearest neighbor to primary spacing ratio.

10 Cross-section expansion has no effect on trunk diameter.

11 Cross-section expansion form spurious grains only for the faster growth speed. This was illustrated by the increased scatter in arm orientation.

In summary, directional solidification of Al-7 wt% Si alloy through a cross-section change for two different growth speeds was executed. Mushy zone morphology along the

length of the samples was successfully measured and analyzed using image-analysis techniques. Several trends were identified for both growth speeds. A new automatic image analysis technique was developed using ImageJ, and executed successfully to measure fraction eutectic. Through this technique, trends regarding fraction eutectic were established for directionally solidified alloys through a cross-section change a varying growth speeds.

CHAPTER V

RECOMMENDATIONS FOR FUTURE RESEARCH

1 Low gravity directional solidification under identical alloy processing conditions is required to examine the role of convection. Such an experiment would help eliminate natural convection from density differences, and convection would only be caused by volume shrinkage. It is expected that a low-gravity environment will produce well defined microstructures with lowered radial macrosegregation.

2 A range of different degrees of cross-section changes should be examined, to see if gradual change will have more or less effect on macrosegregation and microstructure.

3 A range of larger cross-sections should be examined. In the smaller diameter, a maximum of 7-8 dendrites are present which leads to large fluctuations of measurements.

4 A range of different compositions and other binary alloys need to be examined to understand the role of thermophysical properties with a cross-section change.

5 A more advanced fluid model needs to be developed to have a better understanding of why a spurious grain may form and where.

6 Ideal and consistent microscopy methods and settings needs to be examined for Al-7 wt. pct. Si to optimize image analysis.

7 Micro-cutting techniques would be desirable to examine macrosegregation in small intervals in the narrow region of the cross-section change.

REFERENCES

[1] M. Flemings, Solidification Processing, McGraw-Hill, 1974.

[2] J. Hunt and S. Lu, "Numerical Modeling of Cellular/Dendritic Array Growth: Spacing and Structure Predictions," *Metall. Mater. Trans. A,* vol. 27A, pp. 611-623, 1996.

[3] S. Tewari and R. Shah, "Macrosegregation During Dendritic Array Arrayed Growth of Hypoeutectic Pb-Sn Alloys: Influence of Primary Arm Spacing and Mushy Zone Length," *Metall. Mater. Trans. A,* vol. 27A, no. 5, 1996.

[4] W. Kurz and D. Fisher, "Dendritic Growth at the Limit Stability of Tip Radius and Spacing," *Acta Metall.,* vol. 29, p. 11, 1981.

[5] R. Trivedi, "Interdendritic Spacing 2: A Comparison of Theory and Experiment," *Metall. Trans. A,* vol. 15A, p. 977, 1984.

[6] M. Gunduz, H. Kaya and E. Cadirli, "Dendritic Growth in an Aluminum-Silicon Alloy," *J. Mater. Eng. Perf.,* vol. 16, pp. 12-21, 2007.

[7] M. Flemings and G. Nereo, "Macrosegregation: Part I," *Trans. TMS-AIME,* vol. 239, p. 1449, 1967.

[8] M. Flemings, R. Mehrabian and G. Nereo, "Macrosegregation: Part II," *Trans. TMS-AIME,* vol. 242, pp. 41-49, 1968.

[9] M. Flemings and G. Nereo, "Macrosegregation: Part III," *Trans. TMS-AIME,* vol. 242, pp. 50-55, 1968.

[10] S. Tewari, R. Shah and M. Chopra, "Thermosolutal Convection and Macrosegregation Caused By Solute Rejection at Cell/Dendrite Tips," *Metall. Mater. Trans. A,* vol. 24A, no. 7, 1993.

[11] H. Dong, H. Dai, J.-C. Geblin, M. Newell, R. Reed, N. D'Souza and P. Brown, "Grain Selection during Solidification in Spiral Grain Selector," *Journal of the Minerals, Metals and Materials Society,* pp. 367-374, 2008.

[12] M. Dupouy and D. Camel, "Effects of Gravity on Columnar Dendritic Growth of Metallic Alloys: Flow Pattern and Mass Transfer," *J. Crystal Growth,* vol. 183, pp. 469-489, 1998.

[13] R. Trivedi, S. Liu, Maumder and E. Simsek, "Microstructure Development In the Directionally Solidified Al - 4.0 Wt. % Cu. Alloy System," *Metall. Mater. Trans. A,* vol. 27, no. 3, 1996.

[14] American Society for Metals, Metals Handbook, vol. 8, 1973, p. 263.

[15] R. Rajamure, *Directional Solidification of Al-7 Wt. % Si Alloy,* Cleveland, OH: Cleveland State University, Department of Chemical and Biomedical Engineering, 2010.

[16] J. Hunt and S. Lu, "A Numerical Analysis of Dendritic and Cellular Array Growth: The Spacing Adjustment Mechanisms," *Journal of Crystal Growth,* vol. 123, pp. 17-34, 1992.

[17] D. Poirier, P. Zhao and J. Heinrich, "Dendritic Solidification of Binary Alloys with Free and Forced Convection," *Int. J. Numer. Meth. Fluids,* vol. 49, pp. 233-266, 2005.

[18] N. Al-Rawahi and G. Tryggvason, "Numerical Simulation of Dendritic Solidification with Convection : Three-Dimensional Flow," *J. Comp. Physics,* vol. 194, pp. 677-696, 2004.

[19] R. Katz and M. Worster, "Simulation of Directional Solidification, Thermochemical Convection, and Chimney Formation in a Hele-Shaw Cell," *J. Comp. Physics,* vol. 227, pp. 9823-9840, 2008.

[20] D. Duhl, M. Gell and A. Giamei, "The Development of Single Crystal Superalloy Turbine Blades," in *Superalloys 1980*, Champion, PA, 1980.

[21] H. Dong, "Analysis of Grain Selection During Directional Solidification of Gas Turbine Blades," in *Proceedings of the World Congress on Engineering*, 2007.

[22] T. Pollock, W. Murphy, E. Goldman, D. Uram and J. Tu, "Grain Defect Formation During Directional Solidification of Nickel Base Single Crystals," in *Superalloys 1992*, Warrendale, PA, 1992.

[23] R. Trivedi, J. Mason, J. Verhoeven and W. Kurz, "Eutectic Spacing Selection in Lead Based Alloy Systems," *Metall. Mater. Trans. A,* vol. 22A, 1990.

[24] S. Pakiru, *Effect of Step Change in Growth Speed During Directional Solidification on Array Morphology of Al - 7 Wt. % Si Alloy,* Cleveland, OH: Cleveland State University, Department of Chemical and Biomedical Engineering, 2011.

[25] R. Trivedi, S. Tewari, H. Miyahara, P. Mazumder and E. Simsek, "Directional Solidification Microstructures in Diffusive and Convective Regimes," *J. Crystal Growth,* vol. 222, pp. 365-379, 2001.

[26] S. Tewari, J. Chen, G. Magadi and H. D. Groh, "Effect of Crucible Diameter Reduction on the Convection, Macrosegregation, and Dendritic Morphology during Directional Solidification of Pb-2.2 Wt Pct Sb Alloy," *Metall. Mater. Trans. A,* vol. 34A, pp. 2985-2990, 2003.

[27] R. Cahn and P. Haasen, Physical Metallurgy, Amsterdam: Elsevier Science Publishers B.V., 1983.

[28] J. Serra, Image Analysis and Mathematical Morphology, London: Academic Press Inc., 1982.

[29] L.-K. Huang and M.-J. Wang, "Image Thresholding by Minimizing the Measures of Fuzziness," *Pattern Recognition,* vol. 28, pp. 41-51, 1995.

[30] J. Hunt, *Solidification of Casting of Metals,* vol. 192, London: The Metals Society, 1979, p. 3.

[31] R. Grugel and R. Pratt, "Microstructural Response to Controlled Accelerations During the Directional Solidification of Al - 6 wt. % Alloys," *Materials Characterization,* vol. 31, no. 4, 1993.

[32] W. Kurz and D. Fisher, Fundamentals of Solidification, Trans. Tech. Publications, 1986.

[33] L. Sturz, H. Diepers, G. Zimmerman and S. Rex, "Directional Solidification of Cellular Arrays in Transparent Alloys," *Microgravity Sci. Technol.,* vol. XVI, pp. 116-119, 2005.

[34] S. Tewari, Y.-H. Weng, G. Ding and R. Trivedi, "Cellular Array Morphology during Directional Solidification," *Metall. Trans. A,* vol. 33, p. 1229, 2002.

[35] L. Sturz, H. Diepers, G. Zimmermann and S. Rex, "Directional Solidification of Cellular Arrays in Transparent Alloys," *Microgravity Sci. Technol.,* vol. XVI, pp. 116-119, 2005.

[36] P. Curreri, P. Lee and D. Stefanescu, "Dendritic Solidification of Alloys in Low Gravity," *Metall. Trans. A,* vol. 19A, pp. 2671-2676, 1998.

[37] P. Lee, N. D'Souza and X. Yang, "Stray Grain Formation in the Seed Region of Single Crystal Turbine Blades," *Journal of the Minerals, Metals, and Materials Society,* vol. 57, pp. 40-44, 2005.

[38] P. Lee, D. Stefanescu and P. Curreri, "Dendritic Solidification of Alloys in Low Gravity," *Metall. Trans. A,* vol. 19A, pp. 2671 - 2676, 1998.

APPENDIX

FRACTION EUTECTIC IMAGEJ MACRO

//this macro takes a batch transverse RGB Al-Si images and converts it to a binary image of alpha and eutectic phase, measures the area fraction of each phase by measuring pixels.

//This program also measures separately the 'core' of the sample fraction eutectic. It requires variables to be changed by the user to adapt to varying growth speeds.

```
macro "Batch Watershed and Concentric Circle Count"{
        requires("1.33s");//uses current version of Imagej
        setBatchMode(true); //runs faster in 'batch mode'
        dir = getDirectory("Choose a Directory "); //choose where images are located
        list = getFileList(dir); //converts into array of files
        list=remove_txt_file_from_list(list); //removes any txt files from the directory
        dir2=getDirectory("Choose a Directory");  //choose where the transformed image goes

        //imagej macro is a limited language.  Creates several arrays based on number of images to be
processed
        px_scale_A=newArray(list.length);
        XM_A=newArray(list.length);
        YM_A=newArray(list.length);
        XC_A=newArray(list.length);
        YC_A=newArray(list.length);
        R_A=newArray(list.length);
        slice_area_A=newArray(list.length);
        slice_mean_A=newArray(list.length);
        watershed_error_A=newArray(list.length);
        alpha_in_eut_A=newArray(list.length);
        frac_alpha_A=newArray(list.length);
        frac_eut_A=newArray(list.length);
        part_count_A=newArray(list.length);
        core_frac_eut_A=newArray(list.length);
```

```
rings_area_A=newArray(list.length*12);
rings_frac_eut_A=newArray(list.length*12);

prtcle_cutoff_A=newArray(list.length);
dist_A=newArray(list.length);

//main loop
for(i=0; i<list.length; i++){

	path = dir+list[i];
	path2=dir2+list[i];
	open(path);

	run("8-bit");

	getDimensions(width,height,channels,slices,frames);

	px_scale_A[i]=get_scale(width,height);// assumes images of certain 'size' are of a certain
scale.  Can be set manually
	remove_scale(px_scale_A[i],width,height); //removes image of a physical 1 mm scale
	add_slice_to_ROI(30,255);//gets outer boundary of slice
	convert_binary();
	mask_small_particles(400*px_scale_A[i]/2425,list[i]);
	ID=getImageID();
	roiManager("select",0);
	getStatistics(area,mean);
	slice_area_A[i]=area;
	slice_mean_A[i]=mean;
	roiManager("deselect");
	run("Select None");

	run("Watershed");

	roiManager("select",0);
	getStatistics(area,mean);
	watershed_error_A[i]=slice_mean_A[i]/255.000-mean/255.000;
	//!VERY IMPORTANT!!  Sets what size particle and under is considered eutectic
	ptcl_area_cutoff=8000*px_scale_A[i]*px_scale_A[i]/2425.00/2425.00;
	prtcle_cutoff_A[i]=8000/2.425/2.425;
	//imagej program to count all particles of a certain size and over to measure total area
```

```
		run("Analyze Particles...", "size="+ptcl_area_cutoff+"-Infinity circularity=0.00-1.00
show=[Masks]");
		selectImage(ID);
		close();

		roiManager("select",0);
		getStatistics(area,mean);

		frac_alpha_A[i]=mean/255.000;
		frac_eut_A[i]=1-frac_alpha_A[i];
		part_count_A[i]=nResults;

		alpha_in_eut_A[i]=slice_mean_A[i]/255.000-frac_alpha_A[i];

		measure_alpha_CoM();  //gets center of mass of alpha phase and centroid of sample
		XM_A[i] = getResult("XM");
		YM_A[i] = getResult("YM");
		XC_A[i] = getResult("X");
		YC_A[i] = getResult("Y");
		dist_A[i]=sqrt(pow((XM_A[i]-XC_A[i])/px_scale_A[i],2)+pow((YM_A[i]-
YC_A[i])/px_scale_A[i],2));
		R_A[i]=getResult("Feret")/2/px_scale_A[i];

		run("Clear Results");

		//program that creates several concentric ring ROIs to measure eutectic separately, based on
an XY-c enter and desired size.
		create_and_measure_rings(XM_A[i],YM_A[i],12,px_scale_A[i]*0.5);

		//cannot transfer several arrays of results, so have to print and read
		for(j=0;j<nResults;j++){
			rings_area_A[12*i+j]=getResult("ringsArea",j);
			rings_frac_eut_A[12*i+j]=1-getResult("ringsMean",j)/255.000;
		}

		rings_ROI_index_A=newArray(nResults);

		for(k=0;k<nResults;k++){
			rings_ROI_index_A[k]=roiManager("count")-k-1;
		}

		//main program that measures fraction eutectic of varying ROIs
		core_frac_eut_A[i]=measure_core(XC_A[i],YC_A[i],1.0*px_scale_A[i]);

		//saves the binary image
		save_with_tag(path2,"-bin.jpg");

		run("Cyan");
		run("RGB Color");

		//physical draws what ROIs were measured with different colors
```

```
                drawROI(rings_ROI_index_A,255,0,255);
                drawROI(roiManager("count")-1,0,255,0);

                roiManager("reset");

                open(path);
                run("8-bit");

                //overlays with original image
        stack_and_project(list[i],"[Average Intensity]");
                getDimensions(width,height,channels,slices,frames);
                run("Size...","width="+width*0.2+" height="+height*0.2+ " constrain average
interpolation=Bilinear");
        save_with_tag(path2,"-stacked.jpg");
                close();
                selectWindow("Results");
                run("Close");
                //function that prints measurements to an .XLS file in case ImageJ crashes midway
        print_current_results(list,dir2,px_scale_A,XC_A,YC_A,XM_A,YM_A,R_A,dist_A,slice_area_A,frac_alpha_A,fr
ac_eut_A,core_frac_eut_A,watershed_error_A,alpha_in_eut_A,prtcle_cutoff_A,rings_frac_eut_A,rings_area_A);

                call("java.lang.System.gc");
        }
}

function
print_current_results(list,dir2,px_scale_A,XC_A,YC_A,XM_A,YM_A,R_A,dist_A,slice_area_A,frac_alpha_A,frac_eut_A,
core_frac_eut_A,watershed_error_A,alpha_in_eut_A,prtcle_cutoff_A,rings_frac_eut_A,rings_area_A){

        for(k=0;k<list.length;k++){
                setResult("Label",k,list[k]);
                setResult("scale px:1mm",k,px_scale_A[k]);
                setResult("Radius of slice [mm]",k,R_A[k]);
                setResult("x-coord centroid [px]",k,XC_A[k]);
                setResult("y-coord centroid [px]",k,YC_A[k]);
                setResult("x-coord cent of mass [px]",k,XM_A[k]);
                setResult("y-coord cent of mass [px]",k,YM_A[k]);
                setResult("Fraction of total radius of CoM from Centroid",k,dist_A[k]);
                setResult("total slice area [mm^2]",k,slice_area_A[k]/(px_scale_A[k]*px_scale_A[k]));
                setResult("fraction alpha",k,frac_alpha_A[k]);
                setResult("fraction eutectic",k,frac_eut_A[k]);
                setResult("2.0 mm diameter core fraction eutectic",k,core_frac_eut_A[k]);
                setResult("watershed error",k,watershed_error_A[k]);
                setResult("fraction of eutectic 'white'",k,alpha_in_eut_A[k]);
                setResult("particle area cutoff [um^2]",k,prtcle_cutoff_A[k]);
                for(m=0;m<12;m++){
                        if(rings_frac_eut_A[12*k+m]>0){
                                setResult("ratio of ring frac eutectic to total frac
eut"+m,k,rings_frac_eut_A[12*k+m]/frac_eut_A[k]);
                        } else {
                                setResult("ratio of ring frac eutectic to total frac eut"+m,k,0);
```

```
                        }
                }
                for(m=0;m<12;m++){
                        if(rings_frac_eut_A[12*k+m]>0){
                                setResult("ring's frac of total slice
area"+m,k,rings_area_A[12*k+m]/slice_area_A[k]);
                        } else {
                                setResult("ring's frac of total slice area"+m,k,0);
                        }
                }
                for(m=0;m<12;m++){
                        if(rings_frac_eut_A[12*k+m]>0){
                                setResult("fraction of total slice radius from CoM
"+m,k,(m+1)*0.5/R_A[k]);
                        } else {
                                setResult("fraction of total slice radius from CoM "+m,k,0);
                        }
                }

        }
        updateResults();
        selectWindow("Results");
        saveAs("results", dir2+"frac_eutectic.xls");
        run("Close");
}

function measure_core(XC,YC,radius){
                makeOval(XC-radius,YC-radius,2*radius,2*radius);
                roiManager("add");
                tmp_A=newArray(0,roiManager("count")-1);
                roiManager("select",tmp_A);
                roiManager("and");
                roiManager("add");
                getStatistics(area,mean);
                roiManager("select",roiManager("count")-2);
                roiManager("delete");
                return 1-mean/255.00;
}

function mask_small_particles(size,name){
        if(is("Inverting LUT")==1)
                run("Invert LUT");
        ID=getImageID();
        run("Analyze Particles...", "size=0-"+size+" circularity=0.00-1.00 show=[Masks]");
        if(is("Inverting LUT")==1)
                run("Invert LUT");
        stack_and_project(name,"[Sum Slices]");
        run("8-bit");
        run("Make Binary");
        if(is("Inverting LUT")==0)
                run("Invert");
```

```
}

function drawROI(A,R,G,B){
	run("Line Width...", "line=20");
	setForegroundColor(R,G,B);
	roiManager("select",A);
	roiManager("Draw");
}

function stack_and_project(name,type){
		run("Images to Stack", "name="+name+" stack title="+name+" use");
		ID=getImageID();
	run("Z Project...", "start=1 stop=2 projection="+type);
		selectImage(ID);
		close();
}

function save_with_tag(tmp_path,tag){
	dotIndex = lastIndexOf(tmp_path, ".");
if (dotIndex!=-1)
	tmp_path = substring(tmp_path, 0, dotIndex); // remove extension
save(tmp_path+tag);
}

function resize(scale){
	getDimensions(width,height,channels,slices,frames);
	run("Size...", "width="+width*scale+" height="+height*scale+" constrain average
interpolation=Bicubic");
}

function remove_txt_file_from_list(A){

	for(i=0;i<A.length;i++){
		if(1==endsWith(A[i],".xls")){
			tmp_A1=Array.slice(A,0,i);
			tmp_A2=Array.slice(A,i+1,A.length);
			A=Array.concat(tmp_A1,tmp_A2);
		}
	}
return A;
}

function create_ROI(xmin,ymin,xmax,ymax,XM,YM,count,radius){

	for (i=1;i<count;i++){

		cxmin=XM-radius*i;
		cxmax=XM+radius*i;
		cymin=YM-radius*i;
		cymax=YM+radius*i;
		outofbounds=0;
```

```
                    if(xmin>cxmin&&ymin>cymin&&xmax<cxmax&&ymax<cymax)
                            outofbounds=1;
                    if(outofbounds==0){
                            makeOval(cxmin,cymin,2*radius*i,2*radius*i);
                            roiManager("add");
                            tmp_A=newArray(0,i);
                            roiManager("select",tmp_A);
                            roiManager("and");
                            roiManager("add");
                            roiManager("select",i);
                            roiManager("delete");
                    }

            }
}

function create_and_measure_rings(XM,YM,count,radius){

            roiManager("select",0);
            getSelectionBounds(x,y,width,height);
            xmin=x;
            ymin=y;
            xmax=y+width;
            ymax=x+width;
            create_ROI(xmin,ymin,xmax,ymax,XM,YM,count,radius);

            run("Set Measurements...", "mean redirect=None decimal=3");
            ROIcount=roiManager("count");
            ringsArea_A=newArray(ROIcount);
            ringsMean_A=newArray(ROIcount);

            roiManager("select",1);
            getStatistics(area,mean);
            ringsArea_A[0]=area;
            ringsMean_A[0]=mean;

            for (i=1;i<ROIcount-1;i++){

                    tmp_A=newArray(i,i+1);
                    roiManager("select",tmp_A);
                    roiManager("xor");
                    getStatistics(area,mean);
                    ringsArea_A[i]=area;
                    ringsMean_A[i]=mean;
                    roiManager("add");
            }

            tmp_A=newArray(0,ROIcount-1);
            roiManager("select",tmp_A);
            roiManager("xor");
            getStatistics(area,mean);
```

```
		ringsArea_A[ROIcount-1]=area;
		ringsMean_A[ROIcount-1]=mean;
		roiManager("add");

		for (j=0;j<ringsArea_A.length;j++){
				setResult("ringsArea",j,ringsArea_A[j]);
				setResult("ringsMean",j,ringsMean_A[j]);
		}
		updateResults();

}

function measure_alpha_CoM(){
		run("Clear Results");
		run("Set Measurements...", "centroid center feret's redirect=None decimal=3");
		roiManager("select",0);
		run("Measure");
}

function particle_area(){
		sum=0;
		for(i=0;i<nResults;i++){
			sum=getResult("Area",i)+sum;
		}

		return sum;
}


function convert_binary(){
		setAutoThreshold("Huang dark");
		 //run("Threshold...");
		getThreshold(lower,upper);
		setThreshold(lower,upper);
		run("Convert to Mask");
}

function add_slice_to_ROI(lo_thresh,hi_thresh){
		setAutoThreshold("Default Dark");
		setThreshold(lo_thresh,hi_thresh,"red");
		run("Threshold...");
		run("Create Selection");
		roiManager("Add");
		resetThreshold();
		selectWindow("Threshold");
		run("Close");
}

function remove_scale(scale,width,height){
		setColor(0,0,0);
		doWand(width-1200*scale/2425, height-500*scale/2425, 125.0, "Legacy");
```

```
	fill();
	run("Select None");
}

function get_scale(width,height){
	if(width<9000){
		scale=1745.00;
	} else if(width>20000){
		scale=2425.00;
	} else
if(parseFloat(IJ.currentMemory())/(width*height)>12&&parseFloat(IJ.currentMemory())/(width*height)<25){
		scale=1745.00;
	} else{
		scale=2425.00;
	}
return scale;
}
```

NEAREST NEIGHBOR EXCEL 2010 VBA MACRO

```
'this program calculates nearest neighbor distances with several sheets of sample measurements with standard
bounding rectangle.  Comments are denoted by ''.  '
Option Base 1 'this  changes the standard array indices at 1 rather than 0'
Sub finding_min_distances_different_categories()

'creates the several arrays needed.  Sizes are decided by number of sheets later'
Dim raw_matrix() As Variant
Dim matrix() As Variant
Dim dist_matrix() As Variant
Dim min_dist_matrix() As Variant
Dim min_core_NA_array() As Variant
Dim min_noncore_NA_array() As Variant
Dim min_core_left_array() As Variant
Dim min_noncore_left_array() As Variant
Dim min_core_right_array() As Variant
Dim min_noncore_right_array() As Variant
Dim scale1 As Variant
Dim summary_stats_cube() As Variant
Dim scale_array As Variant
ReDim scale_array(Worksheets.count - 1, 1)
Dim dendrite_count_array()
ReDim dendrite_count_array(Worksheets.count - 1, 1)

For i = 1 To Worksheets.count - 1

Worksheets(i).Activate

Range(Cells(1, 15), Cells(500, 500)).Clear
```

```
Range(Cells(2, 7), Cells(500, 13)).Clear

   label_headers 'function that creates label headers'

   counter2 = count_rows
   dendrite_count_array(i, 1) = counter2
   scale1 = Cells(2, 14).Value
   scale_array(i, 1) = Cells(2, 14).Value
'applying the array size for this sheet based on how many calculations there are'
   ReDim min_dist_matrix(counter2, 6) As Variant
   ReDim min_core_NA_array(counter2) As Variant
   ReDim min_noncore_NA_array(counter2) As Variant
   ReDim min_core_left_array(counter2) As Variant
   ReDim min_noncore_left_array(counter2) As Variant
   ReDim min_core_right_array(counter2) As Variant
   ReDim min_noncore_right_array(counter2) As Variant
   ReDim Preserve summary_stats_cube(Worksheets.count - 1, 6, 4) As Variant

   ReDim raw_matrix(counter2, 4)
   raw_matrix = read_sheet_and_load_matrix(counter2, 4)

   ReDim dist_matrix(counter2, counter2)
   dist_matrix = dist_between_points_matrix(raw_matrix, 2)

'because the first set of Al-Si data was split in half, minimum distances were calculated separately for left or right
size, and then also calculated separately for Core or Non-Core, leaving 6 different categories of minimum
distances'
   min_noncore_NA_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix,
"N/A", 3)
   min_core_NA_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix, "N/A", 3,
"Core", 4)
   min_noncore_left_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix,
"Left", 3)
   min_core_left_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix, "Left", 3,
"Core", 4)
   min_noncore_right_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix,
"Right", 3)
   min_core_right_array = zero_undesirable_matrix_columns_and_find_minimum(dist_matrix, raw_matrix,
"Right", 3, "Core", 4)

'takes out all the doubles for the arrays'
   min_dist_matrix = combine_arrays_to_matrix(min_noncore_NA_array, min_core_NA_array,
min_noncore_left_array, min_core_left_array, min_noncore_right_array, min_core_right_array)
   array_to_spreadsheet min_dist_matrix, 2, 7
   min_dist_matrix = remove_matrix_doubles_in_columns(min_dist_matrix)

'takes all the stats gathered for all the sheets and prints a summary on one sheet'
   For j = 1 To 6
      summary_stats_cube(i, j, 1) = non_empty_count_of_matrix(min_dist_matrix, j)
      summary_stats_cube(i, j, 2) = avg_of_matrix(min_dist_matrix, summary_stats_cube(i, j, 1), j)
```

```
    summary_stats_cube(i, j, 3) = stdev_of_matrix(min_dist_matrix, summary_stats_cube(i, j, 2),
summary_stats_cube(i, j, 1), j)
    summary_stats_cube(i, j, 4) = Worksheets(i).Name
  Next j

  If summary_stats_cube(i, 1, 1) < 1 Then
    summary_stats_cube(i, 1, 1) = summary_stats_cube(i, 3, 1) + summary_stats_cube(i, 5, 1)
    summary_stats_cube(i, 2, 1) = summary_stats_cube(i, 4, 1) + summary_stats_cube(i, 6, 1)
    summary_stats_cube(i, 1, 2) = combine_avg(summary_stats_cube(i, 3, 1), summary_stats_cube(i, 5, 1),
summary_stats_cube(i, 3, 2), summary_stats_cube(i, 5, 2))
    summary_stats_cube(i, 2, 2) = combine_avg(summary_stats_cube(i, 4, 1), summary_stats_cube(i, 6, 1),
summary_stats_cube(i, 4, 2), summary_stats_cube(i, 6, 2))
    summary_stats_cube(i, 1, 3) = combine_stdev(summary_stats_cube(i, 3, 1), summary_stats_cube(i, 5, 1),
summary_stats_cube(i, 3, 2), summary_stats_cube(i, 5, 2), summary_stats_cube(i, 1, 2), summary_stats_cube(i, 3,
3), summary_stats_cube(i, 5, 3))
    summary_stats_cube(i, 2, 3) = combine_stdev(summary_stats_cube(i, 4, 1), summary_stats_cube(i, 6, 1),
summary_stats_cube(i, 4, 2), summary_stats_cube(i, 6, 2), summary_stats_cube(i, 2, 2), summary_stats_cube(i, 4,
3), summary_stats_cube(i, 6, 3))
  End If
Next i

Worksheets(Worksheets.count).Activate
For a = 1 To i - 1
  For b = 1 To 3
Cells(a + 1, b + 1).Value = summary_stats_cube(a, 1, b)
Cells(a + 1, b + 4).Value = summary_stats_cube(a, 2, b)
  Next b
Cells(a + 1, 1).Value = summary_stats_cube(a, 1, 4)
Next a
summary_page_headers

array_to_spreadsheet scale_array, 2, 10
array_to_spreadsheet dendrite_count_array, 2, 9

End Sub
'end of the main'

Function summary_page_headers()
Range("A1:G1").ColumnWidth = 15
Range("A1:G1").RowHeight = 90
columns("B").NumberFormat = "0"
columns("E").NumberFormat = "0"

Range("A1").Value = "Sample Name"
Range("B1").Value = "total sample: unique n.n. dist. count (adds left & right counts if applicable)"
Range("C1").Value = "total sample: n.n. avg (weighted average of left & right if applicable) [µm]"
Range("D1").Value = "total sample: n.n. stdev (pooled stdev of left & right population if applicable) [µm]"
Range("e1").Value = "core of sample: n.n. count (adds left & right counts if applicable)"
Range("f1").Value = "core of sample: n.n. avg (weighted average of left & right if applicable) [µm]"
Range("g1").Value = "core of sample: n.n. stdev (pooled stdev of left & right population if applicable) [µm]"
Range("h1").Value = "Distance from Dia. transition in growth direction  [mm]"
```

```
Range("i1").Value = "Dendrite Count"
Range("j1").Value = "PX:mm Scale"
Range("k1").Value = "AREA of sample from 'Fraction Eutectic' spreadsheet [px]^2"
Range("l1").Value = "AREA [µm]^2"
Range("m1").Value = "Primary Spacing: sqrt (area/(count-1))"
Range("n1").Value = "N.N. avg/Primary Spacing"


Range("A1:N1").WrapText = True
End Function
Function combine_avg(count1 As Variant, count2 As Variant, avg1 As Variant, avg2 As Variant) As Variant

If (count1 = 0 And count2 = 0) Then
combine_avg = Empty
Else
combine_avg = (count1 * avg1 + count2 * avg2) / (count1 + count2)
End If
End Function
Function combine_stdev(count1 As Variant, count2 As Variant, avg1 As Variant, avg2 As Variant, avgtot As
Variant, stdev1 As Variant, stdev2 As Variant) As Variant

If count1 = 0 And count2 = 0 Then
combine_stdev = Empty
ElseIf count1 = Empty Then
combine_stdev = stdev2
ElseIf count2 = Empty Then
combine_stdev = stdev1
Else
combine_stdev = Sqr((count1 * (avg1 ^ 2 + stdev1 ^ 2) + count2 * (avg2 ^ 2 + stdev2 ^ 2)) / (count1 + count2) -
avgtot ^ 2)
End If
End Function
Function combine_arrays_to_matrix(array1 As Variant, array2 As Variant, Optional array3 As Variant, Optional
array4 As Variant, Optional array5 As Variant, Optional array6 As Variant) As Variant

column = 0

   If IsEmpty(array1) = False Then column = column + 1
   If IsEmpty(array2) = False Then column = column + 1
   If IsEmpty(array3) = False Then column = column + 1
   If IsEmpty(array4) = False Then column = column + 1
   If IsEmpty(array5) = False Then column = column + 1
   If IsEmpty(array6) = False Then column = column + 1
ReDim matrix(UBound(array1), column) As Variant
For i = 1 To column
   For j = 1 To UBound(array1)
      If i = 1 Then matrix(j, i) = array1(j, 1)
      If i = 2 Then matrix(j, i) = array2(j, 1)
      If i = 3 Then matrix(j, i) = array3(j, 1)
      If i = 4 Then matrix(j, i) = array4(j, 1)
      If i = 5 Then matrix(j, i) = array5(j, 1)
```

```
    If i = 6 Then matrix(j, i) = array6(j, 1)
  Next j
Next i

combine_arrays_to_matrix = matrix
End Function
Function non_empty_count_of_matrix(matrix As Variant, Optional column As Variant = -1) As Variant
count = 0
  If column < 0 Then
  For i = 1 To UBound(matrix, 1)
    For j = 1 To UBound(matrix, 2)
      If IsEmpty(matrix(i, j)) = False Then count = count + 1
    Next j
  Next i
  Else
  For i = 1 To UBound(matrix, 1)
    If IsEmpty(matrix(i, column)) = False Then count = count + 1
  Next i
  End If

  non_empty_count_of_matrix = count

End Function
Function avg_of_matrix(matrix As Variant, count As Variant, Optional column As Variant = -1) As Variant

avg = 0
Sum = 0

  If column < 0 Then
  For i = 1 To UBound(matrix, 1)
    For j = 1 To UBound(matrix, 2)
      If matrix(i, j) <> Empty Then Sum = Sum + matrix(i, j)
    Next j
  Next i
  Else
  For i = 1 To UBound(matrix, 1)
    If matrix(i, column) <> Empty Then Sum = Sum + matrix(i, column)
  Next i
  End If

If count > 0 Then avg_of_matrix = Sum / count Else avg_of_matrix = Empty
End Function
Function stdev_of_matrix(matrix As Variant, avg As Variant, count As Variant, Optional column As Variant = -1)
As Variant

  Sum = 0
  If column < 0 Then
  For i = 1 To UBound(matrix, 1)
    For j = 1 To UBound(matrix, 2)
      If matrix(i, j) <> Empty Then Sum = Sum + (avg - matrix(i, j)) ^ 2
    Next j
```

```
  Next i
  Else
  For i = 1 To UBound(matrix, 1)
    If matrix(i, column) <> Empty Then Sum = Sum + (avg - matrix(i, column)) ^ 2
  Next i
  End If

If count > 0 Then stdev_of_matrix = Sqr(Sum / count) Else stdev_of_matrix = Empty
End Function
Function label_headers()
Range("A1:N1").WrapText = True
Range("A1:N1").ColumnWidth = 12
columns("G:N").NumberFormat = "0.0"
Range("A1").Value = "X-coord [px]"
Range("B1").Value = "Y-coord [px]"
Range("C1").Value = "Left or Right (or N/A)"
Range("D1").Value = "Core or not-Core"
Range("E1").Value = "x-coord [µm]"
Range("F1").Value = "y-coord [µm]"
Range("G1").Value = "n.n. distance: whole sample [µm]"
Range("H1").Value = "n.n. distance: whole core of sample [µm]"
Range("I1").Value = "n.n. distance: left-piece of sample [µm]"
Range("J1").Value = "n.n. distance: left-piece core of sample [µm]"
Range("K1").Value = "n.n. distance: right-piece of sample [µm]"
Range("L1").Value = "n.n. distance: right-piece core of sample [µm]"
Range("M1").Clear
Range("N1").Value = "Scale: px per mm"


End Function
Function array_to_spreadsheet(matrix As Variant, start_row As Integer, start_column As Integer)

For i = 1 To UBound(matrix, 1)
  For j = 1 To UBound(matrix, 2)
Cells(i + start_row - 1, j + start_column - 1).Value = matrix(i, j)
  Next j
Next i

End Function

Function zero_undesirable_matrix_columns_and_find_minimum(matrix2 As Variant, identity_matrix As Variant,
identity_value1 As String, identity_column1 As Integer, Optional identity_value2 As String, Optional
identity_column2 As Integer) As Variant

ReDim matrix(UBound(matrix2, 1), UBound(matrix2, 2))

For i = 1 To UBound(matrix2, 1)
  For j = 1 To UBound(matrix2, 2)
    If identity_value2 = "" Then
      If identity_matrix(j, identity_column1) = identity_value1 And identity_matrix(i, identity_column1) =
identity_value1 Then
```

```
matrix(j, i) = matrix2(j, i)
        Else
matrix(j, i) = 0
        End If
      End If

      If identity_value2 <> "" Then
        If identity_matrix(j, identity_column1) = identity_value1 And identity_matrix(i, identity_column1) =
identity_value1 And identity_matrix(j, identity_column2) = identity_value2 And identity_matrix(i,
identity_column2) = identity_value2 Then
matrix(j, i) = matrix2(j, i)
        Else
matrix(j, i) = 0
        End If
      End If
   Next j
Next i

zero_undesirable_matrix_columns_and_find_minimum = find_nonzero_minimum_of_matrix_columns(matrix)

End Function
Function find_nonzero_minimum_of_matrix_columns(matrix2 As Variant) As Variant

ReDim matrix(UBound(matrix2, 1), 1)
Min = 10 ^ 6
For i = 1 To UBound(matrix2, 1)
   For j = 1 To UBound(matrix2, 1)
      If matrix2(i, j) > 0 And matrix2(i, j) < Min Then Min = matrix2(i, j)
   Next j
   If Min = 10 ^ 6 Then Min = Empty
matrix(i, 1) = Min
   Min = 10 ^ 6
Next i

find_nonzero_minimum_of_matrix_columns = matrix
End Function
Function dist_between_points_matrix(matrix2 As Variant, column As Variant) As Variant

scale1 = Cells(2, 14).Value
ReDim matrix(UBound(matrix2, 1), UBound(matrix2, 1))
For i = 1 To UBound(matrix2, 1)
   For j = 1 To UBound(matrix2, 1)
matrix(i, j) = 1000 * Sqr((matrix2(i, column - 1) - matrix2(j, column - 1)) ^ 2 + (matrix2(i, column) - matrix2(j,
column)) ^ 2) / scale1
   Next j
Next i

dist_between_points_matrix = matrix
End Function
```

```
Function read_sheet_and_load_matrix(row As Variant, column As Variant) As Variant

ReDim matrix(row, column)
For i = 1 To row
  For j = 1 To column
matrix(i, j) = Cells(i + 1, j).Value
  Next j
Next i

read_sheet_and_load_matrix = matrix

End Function
Function count_rows() As Integer

i = 0
While (IsEmpty(Cells(i + 2, 1).Value) = False)
i = i + 1
Wend

count_rows = i
End Function
Function test_matrix_values(matrix As Variant)

For i = 1 To UBound(matrix, 1)
  For j = 1 To UBound(matrix, 2)
Cells(i + 1, j + 20).Value = matrix(i, j)
  Next j
Next i

End Function

Function remove_matrix_doubles_in_columns(matrix As Variant) As Variant

For i = 1 To UBound(matrix, 2)
  For j = 1 To UBound(matrix, 1)
    For k = j + 1 To UBound(matrix, 1)
      If matrix(j, i) = matrix(k, i) Then matrix(k, i) = Empty
    Next k
  Next j
Next i

remove_matrix_doubles_in_columns = matrix

End Function
```

TRUNK DIAMETER INTERSECTION VBA MACRO 2010

```
Sub Calculating_Average_Dia_From_Raw()
```

```
'this program calculates an intersection of two lines, and uses that as an XY center based on several sheets of standard bounding rectangle measurements taken with imagej.

Worksheets(Worksheets.Count).Activate
Columns("B:J").ColumnWidth = 13#
Columns("A:A").ColumnWidth = 25#
Rows("1:1").RowHeight = 100#
Range(Cells(2, 1), Cells(Worksheets.Count, 1)).NumberFormat = "@"
Range(Cells(2, 2), Cells(Worksheets.Count, 5)).NumberFormat = "0.0"

Range("A1").Select
  ActiveCell.FormulaR1C1 = "Sample Name"
  Selection.WrapText = True

Range("B1").Select
  ActiveCell.FormulaR1C1 = "Average Diameter [micrometers]"
  Selection.WrapText = True

Range("C1").Select
  ActiveCell.FormulaR1C1 = "Standard Deviation [micrometers]"
  Selection.WrapText = True

'need final sheet labels

'define variables
Dim dia1, dia2, scale1, xbounding1, ybounding1, boundwidth1, boundheight1, angle1 As Double
Dim xbounding2, ybounding2, boundwidth2, boundheight2, angle2 As Double
Dim xcbounding1, ycbounding1, xcbounding2, ycbounding2, yintercept1, yintercept2, slope1, slope2, xcentroid, ycentroid As Double
Dim dividexbound, divideybound, divideboundwidth, divideboundheight, divideangle, dividexcbound, divideycbound, divideslope, divideyint, xofdivideline As Double

'define arrays
Dim corearray() As Double
ReDim corearray(0) As Double

Pi = 3.14159

counter = 1

'NOTE: counter must stop short of count if extra non-raw sheets have been added to the end, NOT including graphs.
'the following loop cycles through the already created 'raw' data sheets and 'calculation' sheets and performs calculations, then writes to 'calc' sheets

Do While counter < Worksheets.Count


  'set column widths and row height
```

```
Worksheets(counter).Activate

Columns("A:J").ColumnWidth = 13#
Rows("1:1").RowHeight = 100#

   'need to create labels


Range("A1").Select
   ActiveCell.FormulaR1C1 = "Measurement count"
   Selection.WrapText = True

Range("B1").Select
   ActiveCell.FormulaR1C1 = "bounding box upper-left x-coord for dendrite diameter"
   Selection.WrapText = True

Range("C1").Select
   ActiveCell.FormulaR1C1 = "bounding box upper-left y-coord for dendrite diameter"
   Selection.WrapText = True

Range("D1").Select
   ActiveCell.FormulaR1C1 = "bounding box width[px]"
   Selection.WrapText = True

Range("E1").Select
   ActiveCell.FormulaR1C1 = "bounding box height[px]"
   Selection.WrapText = True

Range("F1").Select
   ActiveCell.FormulaR1C1 = "Angle of diameter: starting point to ending point"
   Selection.WrapText = True

Range("G1").Select
   ActiveCell.FormulaR1C1 = "Diameter of dendrite[px]"
   Selection.WrapText = True

Range("H1").Select
   ActiveCell.FormulaR1C1 = "Core[true] or not-core[blank]"
   Selection.WrapText = True

Range("I1").Select
   ActiveCell.FormulaR1C1 = Null

Range("J1").Select
   ActiveCell.FormulaR1C1 = "bounding box for dividing line, if applicable"
   Selection.WrapText = True

Range("J4").Select
   ActiveCell.FormulaR1C1 = "scale:[px] per [mm]"

Range("K1:O1").Select
```

```
    Selection.FormulaR1C1 = Null
Worksheets(counter + 1).Activate

Columns("A:J").ColumnWidth = 13#
Rows("1:1").RowHeight = 100#

Range("A1").Select
    ActiveCell.FormulaR1C1 = "Dendrite number"
    Selection.WrapText = True

Range("B1").Select
    ActiveCell.FormulaR1C1 = "Avg Diameter[px]"
    Selection.WrapText = True

Range("C1").Select
    ActiveCell.FormulaR1C1 = "Avg Diameter[micrometer]"
    Selection.WrapText = True

Range("D1").Select
    ActiveCell.FormulaR1C1 = "x-coord bounding box 1 center"
    Selection.WrapText = True

Range("E1").Select
    ActiveCell.FormulaR1C1 = "x-coord bounding box 2 center"
    Selection.WrapText = True

Range("F1").Select
    ActiveCell.FormulaR1C1 = "y-coord bounding box 1 center"
    Selection.WrapText = True

Range("G1").Select
    ActiveCell.FormulaR1C1 = "y-coord bounding box 2 center"
    Selection.WrapText = True

Range("H1").Select
    ActiveCell.FormulaR1C1 = "slope line 1"
    Selection.WrapText = True

Range("I1").Select
    ActiveCell.FormulaR1C1 = "slope line 2"
    Selection.WrapText = True

Range("J1").Select
    ActiveCell.FormulaR1C1 = "y-intercept line 1"
    Selection.WrapText = True

Range("K1").Select
    ActiveCell.FormulaR1C1 = "y-intercept line 2"
    Selection.WrapText = True

Range("L1").Select
```

```
    ActiveCell.FormulaR1C1 = "x-coord diameter intersection"
    Selection.WrapText = True

Range("M1").Select
    ActiveCell.FormulaR1C1 = "y-coord diameter intersection"
    Selection.WrapText = True

Range("N1").Select
    ActiveCell.FormulaR1C1 = "Left or Right of dividing line (if applicable)"
    Selection.WrapText = True

Range("O1").Select
    ActiveCell.FormulaR1C1 = "Core or non-Core"
    Selection.WrapText = True


    'now need to loop through cells and create formulas
    counter2 = 0
    Do While IsEmpty(Worksheets(counter).Cells((counter2 * 2 + 2), 7)) = False

        'read variables from raw data
        dia1 = Worksheets(counter).Cells(counter2 * 2 + 2, 7)
        dia2 = Worksheets(counter).Cells(counter2 * 2 + 3, 7)

        scale1 = Worksheets(counter).Range("J5")
        xbounding1 = Worksheets(counter).Cells(counter2 * 2 + 2, 2)
        ybounding1 = Worksheets(counter).Cells(counter2 * 2 + 2, 3)
        boundwidth1 = Worksheets(counter).Cells(counter2 * 2 + 2, 4)
        boundheight1 = Worksheets(counter).Cells(counter2 * 2 + 2, 5)
        angle1 = Worksheets(counter).Cells(counter2 * 2 + 2, 6) * Pi / 180

        xbounding2 = Worksheets(counter).Cells(counter2 * 2 + 3, 2)
        ybounding2 = Worksheets(counter).Cells(counter2 * 2 + 3, 3)
        boundwidth2 = Worksheets(counter).Cells(counter2 * 2 + 3, 4)
        boundheight2 = Worksheets(counter).Cells(counter2 * 2 + 3, 5)
        angle2 = Worksheets(counter).Cells(counter2 * 2 + 3, 6) * Pi / 180

    'MsgBox (dia1 & "A " & dia2 & "B " & scale1 & "C " & xbounding1 & "D " & ybounding1 & "E " & boundwidth1
& "F " & boundheight1 & "G " & angle1 & "H " & xbounding2 & "I " & ybounding2 & "J " & boundwidth2 & "K " &
boundheight2 & "L " & angle2)


        'calculate new values
avgDia = Application.Average(dia1, dia2)
avgDiamicrometer = 1000 * avgDia / scale1

        xcbounding1 = xbounding1 + boundwidth1 / 2
        ycbounding1 = ybounding1 + boundheight1 / 2

        xcbounding2 = xbounding2 + boundwidth2 / 2
        ycbounding2 = ybounding2 + boundheight2 / 2
```

```
    slope1 = (-1) * Sin(angle1) / Cos(angle1)
    slope2 = (-1) * Sin(angle2) / Cos(angle2)


    yintercept1 = ycbounding1 - slope1 * xcbounding1
    yintercept2 = ycbounding2 - slope2 * xcbounding2

xcentroid = (yintercept2 - yintercept1) / (slope1 - slope2)
ycentroid = slope1 * xcentroid + yintercept1

    'write variables to sheet

Worksheets(counter + 1).Cells(counter2 + 2, 1).Value = counter2 + 1
Worksheets(counter + 1).Cells(counter2 + 2, 2).Value = avgDia
Worksheets(counter + 1).Cells(counter2 + 2, 3).Value = avgDiamicrometer
Worksheets(counter + 1).Cells(counter2 + 2, 4).Value = xcbounding1
Worksheets(counter + 1).Cells(counter2 + 2, 5).Value = xcbounding2
Worksheets(counter + 1).Cells(counter2 + 2, 6).Value = ycbounding1
Worksheets(counter + 1).Cells(counter2 + 2, 7).Value = ycbounding2
Worksheets(counter + 1).Cells(counter2 + 2, 8).Value = slope1
Worksheets(counter + 1).Cells(counter2 + 2, 9).Value = slope2
Worksheets(counter + 1).Cells(counter2 + 2, 10).Value = yintercept1
Worksheets(counter + 1).Cells(counter2 + 2, 11).Value = yintercept2
Worksheets(counter + 1).Cells(counter2 + 2, 12).Value = xcentroid
Worksheets(counter + 1).Cells(counter2 + 2, 13).Value = ycentroid

    If IsEmpty(Worksheets(counter).Cells(counter2 * 2 + 2, 8)) = False Then

Worksheets(counter + 1).Cells(counter2 + 2, 15).Value = "Core"
       ReDim Preserve corearray(UBound(corearray) + 1) As Double
corearray(UBound(corearray) - 1) = Worksheets(counter + 1).Cells(counter2 + 2, 3).Value

    Else
Worksheets(counter + 1).Cells(counter2 + 2, 15).Value = "not-Core"
    End If

    If IsEmpty(Worksheets(counter).Range("J2")) = False Then

dividexbound = Worksheets(counter).Range("K2")
divideybound = Worksheets(counter).Range("L2")
divideboundwidth = Worksheets(counter).Range("M2")
divideboundheight = Worksheets(counter).Range("N2")
divideangle = Worksheets(counter).Range("O2") * Pi / 180

dividexcbound = dividexbound + divideboundwidth / 2
divideycbound = dividexbound + divideboundwidth / 2
divideslope = (-1) * Sin(divideangle) / Cos(divideangle)
divideyint = divideycbound - divideslope * dividexcbound

xofdivideline = (ycentroid - divideyint) / divideslope
```

```
        If xofdivideline > xcentroid Then
Worksheets(counter + 1).Cells(counter2 + 2, 14).Value = "Left"
        Else
Worksheets(counter + 1).Cells(counter2 + 2, 14).Value = "Right"
        End If

      Else
Worksheets(counter + 1).Cells(counter2 + 2, 14).Value = "N/A"

      End If


      counter2 = counter2 + 1
   Loop

  'add average and stdev for diameters on each sheet, then create new sheet to post numbers after loop

Worksheets(counter + 1).Cells(counter2 + 4, 2).Select
  ActiveCell.FormulaR1C1 = "Average Dia"

Worksheets(counter + 1).Cells(counter2 + 5, 2).Select
  ActiveCell.FormulaR1C1 = "STDEV"

  Set range1 = Worksheets(counter + 1).Range(Cells(2, 3), Cells(counter2 + 2, 3))
  average1 = Application.WorksheetFunction.Average(range1)
  stdev1 = Application.WorksheetFunction.StDev(range1)


  'adds values to current sheet
Worksheets(counter + 1).Cells(counter2 + 4, 3).Value = average1
Worksheets(counter + 1).Cells(counter2 + 5, 3).Value = stdev1

  'adds values to final sheet
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 1).Value = Worksheets(counter + 1).Name
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 2).Value = average1
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 3).Value = stdev1

  'can use redim to "reassign" a size to an array, makes the array dynamic
  'using Ubound always gives one extra array element of zero, so need to chop the last one off

  If UBound(corearray) > 1 Then
  ReDim Preserve corearray(UBound(corearray) - 1) As Double
coreaverage = Application.WorksheetFunction.Average(corearray)
corestdev = Application.WorksheetFunction.StDev(corearray)
  Else
coreaverage = corearray(0)
corestdev = 0
  End If
```

```
  'adding labels
Worksheets(counter + 1).Cells(counter2 + 7, 2).Select
  ActiveCell.FormulaR1C1 = "Avg Core Dia."

Worksheets(counter + 1).Cells(counter2 + 8, 2).Select
  ActiveCell.FormulaR1C1 = "Core Stdev"

  'adds values to current sheet
Worksheets(counter + 1).Cells(counter2 + 7, 3).Value = coreaverage
Worksheets(counter + 1).Cells(counter2 + 8, 3).Value = corestdev

  'adds values to final sheet
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 4) = Worksheets(counter + 1).Name + "-CORE"
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 5) = coreaverage
Worksheets(Worksheets.Count).Cells((counter + 1) / 2 + 1, 6) = corestdev
  ReDim corearray(0) As Double
  'NOTE: if creating worksheets for the first time counter = counter + 1

counter = counter + 2
Loop

End Sub
```