Resolution Enhancement in Photolithography via STED Inspired Process and Novel Mask Development

by

David B. Miller

B.S. University of Colorado at Boulder, 2010

M.S., University of Colorado at Boulder, 2017

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Electrical Engineering

2020

This thesis entitled:
Resolution Enhancement in Photolithography via STED Inspired Process and Novel Mask Development
written by David B. Miller
has been approved for the Department of Electrical Engineering

____

Prof. Robert R. McLeod

____

Adam M. Jones

Date ____________________

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Miller, David B. (Ph.D., Electrical Engineering)

Resolution Enhancement in Photolithography via STED Inspired Process and Novel Mask Development

Thesis directed by Prof. Robert R. McLeod

Photolithography is a process that transfers a two dimensional pattern onto the surface of a substrate via optical exposure, and is a key enabling technology for IC fabrication. Being an optical patterning process, the resolution is diffraction limited. A constant push for decreasing transistor size requires corresponding improvements in resolution. Typically, resolution improvements have come from reduced exposure wavelength, improved optics, improved photoresist chemistry, multiple patterning, and resolution enhancement techniques.

This thesis develops a resolution enhancement technique, originally inspired by a technique used for super-resolution microscopy, capable of extending the critical dimension resolution of lithographic tools beyond the diffraction limit. Due to the non-linear response of positive tone photoresist to optical dose, it is possible to super-localize features, much like in the case of STED microscopy. Unlike other lithographic super-resolution techniques, this is accomplished with a single wavelength, over a broad area in the far-field, with i-line (near UV) illumination, and is compatible with commercial i-line steppers.

By saturating the response of photoresist, features can be localized to the center of deep intensity nulls. These nulls are produced via optical interference. Models are developed to explore the optical requirements and limits of the resolution enhancement. Resist processing is optimized and these models are validated using laser interference lithography. While periodic patterns can be produced through traditional multi-beam interference, arbitrary two dimensional patterns cannot. To enable arbitrary patterning, novel lithographic masks, pixelated polarization phase shifting masks (P3SM), with both local polarization and phase control are developed. Simulations are written to explore the mask design space, and P3SMs are designed and fabricated. These masks

are demonstrated by projection lithography using a lab-built i-line stepper.

Modeling shows that the ultimate limit to sub-diffraction feature size is set by both optical contrast and resist contrast. Additionally, the scaling of size with dose is contrast dependent. When optical contrast is perfect, size scales as dose1/2, as is the case for STED. Experiments validated these limits and scaling rates, and demonstrated features as small as 50 nm written at i-line. Deterministic linewidth control is demonstrated, as is low line edge roughness. Phase masks that incorporate polarization control solve key issues in traditional phase masks. Not only is interference contrast maximized in multiple simultaneous orientations, but also the added degree of freedom enables line end termination and resolution of phase conflicts. A strategy for polarization mask design is proposed based on a solution to the Laplace equation. Initial mask experiments demonstrate the line end termination enabled by polarization control.

These dimensions are similar to what may be achieved using scanning near-field, DUV, or e-beam lithography, yet achieved with far-field near UV exposures over a large area. Deterministic linewidth control and low LER make this process viable for fabrication at length scales well below those typically achieved with i-line tools. Finally, the addition of pixelated polarization control to photomasks solves the key issues preventing broader application of phase masks in lithography.

Dedication

First, to my grandfather, who challenged every statement I ever made, and required that all of his grandchildren become good writers. And, to Dr Thomas Schibli, who funded what very well may be one of the most expensive undergraduate honor thesis. Both my career and education in this field are thanks to him.

Acknowledgements

This work was possible thanks to the support of many professors, staff, and students at the university and Sandia National Labs. My work was supported by a generous grant from Sandia, as well as NSF MRSEC Grant DMR-1420736. Thank you to my committee members: Dr. Robert McLeod, Dr. Adam Jones, Dr. Alan Mickelson, Dr. Won Park, Dr. Shu-Wei Huang for your support and guidance. I also would like to thank my professors Dr. Juliet Gopinath, Dr. A.J. Gasiewski, Dr. Kelvin Wagner for providing the foundations I needed to build upon to complete my doctorate. Thank you Adam Sadoff for your support throughout this process. Thank you both past and present members of the McLeod group who have provided various support including Dr. David Glugla, Dr. Marvin Alim, Dr. Jacob Friedlein, Dr. Darren Forman, Megan Renny, Archish Muralidharan, and Jon Hergert, and Charlie Rackson. Thank you to both the Physics Instrument Shop and Jila Instrument Shop, in particular Hans Green and Charlie Bowen, for all of your assistance in the design and fabrication of hardware and the JILA electronics shop for providing assistance in designing and troubleshooting electronics. Finally, thank you Alex Denton and Dr. Tomoko Borsa for your work in running and maintaining the CNL and NCF facilities, without which this work wouldn't be possible.

Contents

Chapter

1 Introduction **1**

2 Background **6**

2.1 Lithography 6

2.2 Resolution 10

2.3 Throughput 12

2.4 Pattern Fidelity 12

2.5 Photoresists 13

2.6 Standing Waves and Anti-Reflection Coatings 16

2.7 Photomasks 18

2.8 Projection Lithography 18

2.9 Projection Tools 19

2.10 Resolution Enhancement Techniques 20

2.11 Phase Masks 21

3 Super-Resolution Lithography **24**

3.1 Ideal Super Resolution lithography 24

3.2 Super-Localized, Single Wavelength, Broad Area Lithography 24

3.3 Process Modeling 25

3.3.1 Aerial image contrast 26

3.3.2 Simplified Photoresist Model 27

3.3.3 Linewidth Model 28
3.3.4 Linewidth Scaling 31
3.4 Overexposure or Underexposure?
LER and NILS 33

4 Laser Interference Lithography **36**
4.1 Interferometer Configurations 37
4.2 Interference Contrast 39
4.2.1 Contrast in Two Beam Interference Patterns 39
4.2.2 Temporal Coherence 40
4.2.3 Beam Pointing Instability 47
4.2.4 Polarization 54
4.2.5 Intensity Difference 59
4.2.6 Multi-Factor Analysis 59
4.2.7 Component Selection and Implementation 64
4.2.8 Summary of Contrast Analysis 64
4.3 Intensity Distribution 65
4.4 Final LIL System Design and Implementation 65

5 Demonstration and Model Validation **71**
5.1 Interference Contrast Estimation 72
5.2 Resist Patterns 72
5.3 Uniformity Improvement by Beam Shaping 80
5.4 Beam Scanning 82

6 i-Line Stepper System **84**
6.1 System Specifications 84
6.2 System Design 85

6.2.1 Illumination 85

6.2.2 Projection Optics 85

6.2.3 Autofocus 88

6.2.4 Mask and Wafer Handling 88

6.2.5 Electronics and Control 90

6.3 System Performance 90

6.3.1 Air Force Resolution Test Mask 91

6.3.2 Contrast measured 93

6.3.3 Developed Resist Patterns 95

7 Pixelated Polarization Phase Masks **97**

7.1 Projection Lithography Simulations 102

7.1.1 Full Vector Simulations 102

7.1.2 Polarized Illumination 104

7.1.3 Coherence 105

7.1.4 Mask File and Mask Fields 105

7.1.5 Resist Simulation 105

7.2 Mask Implementation 105

7.3 Mask Layout 106

7.4 Example Lithographic Patterns 107

7.4.1 Line Ends 107

7.4.2 Circle 118

7.4.3 3 Way intersection 120

7.4.4 Alternating Lines 124

7.4.5 Sub-Contrast Assist Features (SCAFs) 129

7.5 Mask Fab 133

7.6 Experimental Results 133

8 Future Work and Summary 138

Bibliography 140

Appendix

A Appendix 144

A.1 Photoresist Contrast Curves . . . 144

A.2 Methods . . . 145

A.2.1 Substrate and Resist Preparation . . . 145

A.2.2 Aerial Image Generation, Resist Exposure and Development . . . 146

A.2.3 Pattern Inspection, Image Processing and Measurement . . . 146

A.2.4 SuMMIT Image Processing Details . . . 146

A.3 Photoresist Contrast Optimization . . . 147

B LIL Control Code 148

B.1 User_Set_Up.m . . . 148

B.2 Interference_Tool_Control.m . . . 150

B.3 PulseGen_init.m . . . 153

B.4 PulseGen_close.m . . . 153

B.5 PulseGenOut.m . . . 154

B.6 Shutter_Open.m . . . 155

B.7 Shutter_Close.m . . . 155

B.8 PM_init.m . . . 156

B.9 PM_close.m . . . 156

B.10 HomeAxis.m . . . 157

B.11 MotionStatus.m . . . 158

B.12 Laser_init.m . . . 159

B.13 Laser_close.m 160

B.14 Laser_close_on.m 160

B.15 SetLaserPower.m 161

C Stepper Control Code **162**

C.1 User_Set_Up.m 162

C.2 Coordinate_Map.m 164

C.3 Stepper_Tool_Control.m 165

C.4 chuck_data.m 168

C.5 XPS_init.m 169

C.6 XPS_close.m 170

C.7 HomeAxis.m 170

C.8 MotionStatus.m 171

C.9 XPS_abs_move_X.m 171

C.10 XPS_abs_move_Y.m 172

C.11 XPS_abs_move_Z.m 172

C.12 XPS_inc_move_X.m 173

C.13 XPS_inc_move_Y.m 173

C.14 XPS_inc_move_Z.m 174

C.15 XPS_position_move_X.m 174

C.16 XPS_position_move_Y.m 174

C.17 XPS_position_move_Z.m 174

C.18 PulseGen_init.m 175

C.19 PulseGen_close.m 175

C.20 PulseGenOut.m 176

C.21 Shutter_Open.m 177

C.22 Shutter_Close.m 177

C.23 Laser_init.m . . . 178

C.24 Laser_close.m . . . 178

C.25 Laser_close_on.m . . . 179

C.26 SetLaserPower.m . . . 179

C.27 auto_focus.m . . . 180

C.28 signal_read_in_manual.m . . . 181

D Vector EM Mask Simulation **184**

D.1 UserSetUp.m . . . 184

D.2 ErrorCheck.m . . . 188

D.3 ProjectionCode.m . . . 190

D.4 CoordSetUp.m . . . 193

D.5 PolSetUp.m . . . 194

D.6 MaskField.m . . . 197

D.7 FourierProp.m . . . 199

D.8 PupilFilter.m . . . 202

D.9 Defocus.m . . . 203

D.10 Aberrations.m . . . 203

D.11 RadiometricCorrection.m . . . 205

D.12 MyPcolor.m . . . 206

D.13 mask_function.m . . . 206

D.14 mask_function2.m . . . 207

D.15 mask_function3.m . . . 208

D.16 mask_function4.m . . . 209

D.17 mask_function5.m . . . 210

D.18 mask_function6.m . . . 211

D.19 mask_custom.m . . . 212

D.20 waveplate_matrix.m 212

D.21 OpticalContrast.m 213

D.22 Resist.m 213

D.23 MeasureLinewidth.m 214

D.24 SpatialCoherence.m 215

D.25 mask_solver.m 216

Tables

Table

5.1 **Interferometer alignment/component tolerances** 72

6.1 Summary of measured image contrasts 93

A.1 **Spin Coat Parameters for AZ4210 and Ultra-I** 145

A.2 **Soft Bake Parameters for AZ4210 and Ultra-I on NG4 Substrates** 145

Figures

Figure

1.1 Illustration of photolithography. A two-dimensional pattern (upper left) is transferred onto the surface of a substrate (lower left). This is accomplished by fabricating a mask with the desired pattern, then projecting a demagnified image of the mask into a light sensitive resin, photoresist (right). This photoresist is then developed leaving the desired pattern. The remaining photoresist acts as a stencil for subsequent processing of the substrate surface. 2

2.1 A generic lithography process. First, a wafer surface is prepared by cleaning, drying, and applying adhesion promoter. Second, a resist is spin coated onto the wafer, and baked. Third, the resist is exposed to a desired pattern, leaving a latent image in the resist. Fourth, the resist is developed, revealing the latent image. Fifth, the wafer surface is processed using the resist pattern as a stencil. Adapted from [33] 7

2.2 Resolution and throughput for various lithographic processes. Throughput for serial write lithography (blue) slows as critical dimension is reduced, as described by Tennant's Law. Several types of serial write are identified at their typical critical dimension length scales. Throughput for parallel write lithography (red) accelerates as critical dimension is reduced, as described by Moore's Law. Several generations of optical lithography are identified along the curve, labeled by the wavelengths used. 9

2.3 In lithography, there are two distinct resolutions: pitch resolution and critical dimensions (CD). Pitch resolution is the minimum distance between uniquely resolvable features. CD is the size, or width, of individual features. 11

2.4 Chemical structures for i-line photoresist components. a) Novolak resin b) Diazonapthoquinone (DNQ) a photoacid generator c) photogenerated acid reaction, DNQ to a carboxylic acid 15

2.5 Typical response curve for positive tone photoresist. Below dose threshold D_1, no resist is removed after development. Above dose threshold D_2, all resist is removed upon development. The slope of the response curve, γ, is the resist contrast. Large γ implies a strong non-linear response, and results in steep sidewalls. 17

2.6 Comparison of chrome masks with phase shifting masks. Chrome masks modulate the electric field amplitude only. As pitch is reduced, images from adjacent apertures overlap, reducing the final image intensity. Phase shifting masks modulate electric field phase with π phase steps, and can incorporate chrome patterns as well to modulate amplitude. Shifting the phase in alternating mask regions forces a zero crossing of the electric field, thereby improving image contrast. 22

2.7 Two significant problems encountered with phase shifting masks. A) A finite line segment (dark black line) cannot be terminated when alternating phases span the line. Dashed red circles indicate regions where a dark null will still form. B) A typical phase conflict problem. A phase cannot be determined for the region circled in red. A π phase will not give a null adjacent to another π phase. Zero phase step has the same problem with adjacent zero phase regions. 23

3.1 A plot of both the aerial image intensity and resulting developed photoresist cross sections. The projected images are all of finite image contrast (M=0.9). A nominal exposure produces a linewidth equal to half of the period. Maximum exposure dose leaves full thickness at the center of the feature. In the insert, a piecewise continuous model for the non-linear transfer function, plotted against Log(dose). 25

3.2 Plot of normalized linewidth as a function of normalized exposure dose. A range of practical image contrasts from $M = 0.9$ to $M = 0.99$ are shown. For all image contrasts, the nominal exposure ($\bar{D} = 1$) produces a diffraction-limited linewidth equal to half the period. Over-exposure given by $\bar{D} > 1$ reduces the linewidth below the diffraction limit. Feature size drops to zero at a dose determined by image contrast M, although, finite resist contrast γ prevents operation at this extreme. . . 30

3.3 Plot of scaling factor, S, for different aerial image contrasts, M. As M approaches 1, the scaling factor drops to 0.5. Decreasing the image contrast increases the scaling factor monotonically. This scaling provides a convenient measurement for M. 32

3.4 Normalized Image Log Slope (NILS) calculations near intensity maxima (peak) and minima (trough) as a function of feature width (CD) and image contrast. Around an intensity maxima, NILS decreases with decreasing CD (left plot). However, around an intensity minima, NILS increases with decreasing CD (right plot). It is for this reason that overexposure is preferred to underexposure. 34

3.5 Examples of pattern fidelity for various exposure conditions. All three patterns are on a 500 nm pitch. Underexposure can yield the same size negative features as overexposure can yield positive features. The key difference is, underexposure leads to poor LER while overexposure leads to improved LER. 35

4.1 Two-beam interferometer utilizing a plate beam splitter. 38

4.2 Two-beam interferometer utilizing a grating beam splitter. 38

4.3 Exact and approximate coherence functions, Γ_{12}, for a laser with 20 longitudinal modes, 150 MHz free spectral range, and linewidth much narrower than the free spectral range. The exact coherence function (blue) is well approximated by a cosine (orange) up to the first null. 42

4.4 Optical path length difference (OPD) across field. Regardless of how carefully aligned each arm of the interferometer is, there will always be an OPD across the field due to the non-zero width of the interfering beams. Beam 1 (blue) and Beam 2 (orange) have equal path length on center. However, Beam 1 travels a shorter distance to the upper edge of the field, while Beam 2 travels a longer distance. The OPD can be found as a function of distance, **R**, from center. 43

4.5 Wave vectors and coordinate system for two beam interference. $\vec{K}_1$ and $\vec{K}_2$ are the wave vectors for the first and second beams respectively. Both lie in the XZ plane, are symmetric about the z-axis, and have angle of incidence θ. 45

4.6 Parameterized plot of Equation 4.14 with $\Delta\phi = 0$. Minimum contrast is plotted as a function of angle of incidence and field radius normalized by coherence length. As angle of incidence increases, field size relative to source coherence length must decrease to maintain constant contrast. 46

4.7 Beam path deviations due to beam pointing instability in an interferometer built with a plate beam splitter. Blue beams represent nominal beam paths, red beams represent perturbed beam paths resulting from beam pointing error δ. Each beam walks off in opposing directions and both beams have an equal change in path length. At the edges of the exposure field, the beams do not remain overlapped and there is no interference. 48

4.8 Beam path deviations due to beam pointing instability in an interferometer built with a grating beam splitter. Blue beams represent nominal beam paths while red beams represent perturbed beam paths resulting from beam pointing error δ. Both beams walk off in the same direction in the plane of interference. One beam gains path length while the other beam loses path length. At the edges of the exposure field, both beams remain overlapped. 48

4.9 Wave vectors with beam pointing errors for interferometers built with: A) a plate beam splitter and B) a grating beam splitter. Nominal wave vectors $\vec{K_1}$ and $\vec{K_2}$ are displaced by the beam pointing error δ resulting in $\vec{K_1'}$ and $\vec{K_2'}$. For a plate beam splitter, the angle of incidence for both wave vectors increases by δ. For a grating beam splitter, the angle of incidence increases for one beam, while decreasing for the other. 50

4.10 Contrast loss across field due to beam wander. A plate beam splitter (blue) is compared with a grating beam splitter (orange). Both have 0.5 rad angle of incidence and 5 μrad total deviation. Contrast is determined by integrating over the intensity at each beam angle. Contrast loss in an interferometer built with a plate beam splitter is roughly equivalent to that of a grating configuration with 10^3 greater beam pointing errors. 52

4.11 Contrast loss across field due to beam wander. A plate beam splitter (blue) is compared with a grating beam splitter (orange). Both have 0.5 rad angle of incidence and 5 mrad total deviation. Contrast is determined by integrating over the intensity at each beam angle. Grating contrast drops to about 0.94 while plate contrast drops to zero. An interferometer built with a grating loses far less contrast with beam wander than does an interferometer built with a plate. 53

4.12 Polarization vectors after beam splitters with rotated linear incident polarization. A) With a plate beam splitter, the polarizations are mirrored, rotating in opposite directions in each arm. B) With a grating beam splitter, the polarizations rotate in the same direction in both arms. 57
4.13 Contrast loss due to polarization misalignment. Angle of incidence $\theta = 0.5$ rad. Interferometers built with a grating beam splitter (orange) are less sensitive to polarization angle misalignment than interferometers built with a plate beam splitter (blue). 58
4.14 Contrast loss due to intensity mismatch between interfering beams. 60
4.15 Contrast as a function of polarization misalignment, ϕ, and degree of polarization. . 61
4.16 Contrast as a function of polarization misalignment, ϕ, and intensity imbalance I_1/I_2. 62
4.17 Contrast as a function of total beam wander, δ, and coherence length, L_c. Calculated for fixed field size of $R = 4$ mm and fixed $\theta = 0.5$. 63
4.18 Schematic of the final design of the LIL tool. An Ar Ion laser ($\lambda = 363.8$ nm) is polarized with a polarizing beam splitter (PBS), then spatially filtered (S.F.) and split by a holographic grating. A resist coated wafer is positioned by a 2 axis motorized stage. Both beams interfere in the plane of the resist. Mirror positions can be adjusted to tune the interference period. Exposure time is controlled by a mechanical shutter. An optional refractive beam shaper (RBS) is inserted between the spatial filter and the grating beam splitter. 67
4.19 Beam path of an early implementation of the laser interference lithography tool. . . 68
4.20 Final implementation of laser interference lithography tool. Beam conditioning optics are not shown. 69
4.21 Wafer Handling. PM500 Stages position each wafer held by a vacuum chuck This eliminates wafer distortion and obscuration of the front side. The vacuum chuck is connected to a small vacuum pump (not shown) via tygon tubing. 70

5.1 In AZ4210, the smallest CD achievable is 80 nm, with a thickness of 150 nm. These lines are on a 620 nm period, with a line edge roughness of 10 nm. 75

5.2 In Ultra-I, the smallest CD achievable is 50 nm, with a thickness of 150 nm. These lines are on a 500 nm period, with a line edge roughness of 5.5 nm. 76

5.3 width= 77

5.4 Higher pattern densities are possible. In Ultra-I exposed with a 250 nm pitch, the smallest CD achievable is again 50 nm, with a thickness of 150 nm. The lower bound on CD is set by the photoresist. 78

5.5 Linewidth dose scaling in both AZ 4210 and Ultra-i resist. Linewidth is normalized to period, and dose is normalized to the dose which produces half pitch patterns. Error bars represent 3σ variation in measured width. 79

5.6 Comparison of gratings recorded with and without beam shaping. Both sets of grating were recorded on a 500 nm pitch, with a range of doses increasing from right to left, bottom to top. Left image contains gratings written with a Gaussian intensity distribution. The right image contains gratings written with a square, flat top intensity distribution. 81

5.7 Measured intensity distribution in the wafer plane using a beam profiler. Left image shows the intensity of a Gaussian beam and cross section. Right image shows the intensity after a refractive beam shaper, as well as a cross section. 81

5.8 Comparison of wafer plane intensity with and without beam scanning. On the left, the stationary intensity distribution measured with a beam profiler, and a cross section (yellow line). On the right, an image of the wafer plane intensity with horizontal and vertical scanning and a cross section (yellow line). Using imageJ, the image was shifted one pixel at a time and summed for a total of ± 20 pixels in X and Y. Each pixel corresponds to 5 μm. 83

6.1 Optical schematic for a tabletop i-line stepper. The key systems are illumination, mask projection, and autofocus. Illumination is provided by either laser or collimated LED. Mask projection optics are composed of microscope objectives. The autofocus system is modeled after those found in optical disk drives. 86

6.2 Lab-built i-line stepper. The system is constructed on a vertical breadboard to allow the mask and wafer to lay flat and level. Mask alignment, including focus, is performed manually with a camera for feedback. Wafer alignment, including focus, is automated. 87

6.3 Autofocus detection. **A)** Through focus spot size after passing through a pair of cylindrical lenses. At best focus, the spot is circular. Before and after best focus, the spot becomes elliptical with different orientations. **B)** Illustration of backreflection from wafer (gold) and the resulting spots on the quad cell detector. **C)** The labeled quadrants of the detector with equations for focus error signal (FES) and position error signals (PES). 89

6.4 The through focus error signal as measured with a 10x 0.25 NA objective. 92

6.5 AFRT illuminated with LED (coherence $\sigma = 0.1$). Imaged with 5x objective and CMOS camera with a 150mm tube lens 94

6.6 AFRT illuminated with laser (spatially coherent). Imaged with 10x objective and CMOS camera with a 150mm tube lens. The largest spatial frequencies (element 6, group 7) are better resolved than with partially coherent illumination, but coherent artifacts are clearly visible. 94

6.7 Optical micrograph of developed resist patterns exposed with fully coherent (laser) illumination. While the resolution is improved over partially coherent illumination, line edge roughness is poor, and the pattern often suffers from coherent artifacts. . . 96

6.8 Optical micrograph of developed resist patterns exposed with partially coherent (collimated LED) illumination. While the resolution is reduced due to diminished contrast, line edge roughness is greatly improved, and the pattern lacks any coherent artifacts. 96

7.1 Imaging an alternating phase shifting mask illuminated with P-polarization. When interfering polarizations are orthogonal, image contrast is zero. In the case of S-polarization, orthogonal polarizations never occur. 99

7.2 Image contrast of phase grating as a function of illuminating polarizations: P-polarization (blue lines) and S-polarization (orange lines). A) Image contrast in air. The P-polarization contrast drops to zero for NA = 0.707 (45 degrees). B) Image contrast in resist (N = 1.6). The P-polarization contrast never drops to zero, but reaches a minimum at M = 0.22. 100

7.3 Example of multiple orientations of phase steps within single pattern enabled by pixelated polarization control. The dark lines indicate the resulting intensity nulls. Arrows indicate polarization orientation, and direction indicates phase. For polarizations π out of phase, arrows differ by 180 degrees. 101

7.4 Simplified imaging system model used for vector Fourier simulation. The electric fields are computed for the mask plane, the pupil plane, and the image plane. The pupil is the transform of the mask, and the image is the scaled transform of the pupil. 103

7.5 A) Ordinary and B) Extraordinary Polarization basis set used in the pupil plane . . 104

7.6 Two potential strategies for terminating line ends with polarization. 108

7.7 Polarization states for a finite line segment. This mask was manually laid out using simple orthogonal polarizations to terminate the line ends. 109

7.8 Aerial intensity of the projected image of the mask in Figure 7.7. The orthogonal polarizations lead to some polarization diffraction, and shallow horizontal intensity nulls. The spurious nulls are undesired, but have poor contrast, and don't print in resist. 110

7.9 Log scale intensity cross section through the center of the image in Figure 7.8. The intensity follows a Sine Integral function as expected. 111

7.10 Developed resist pattern resulting from a nominal exposure of the intensity distribution shown in Figure 7.8. The spurious nulls are too shallow to print, and the line ends are bulbous. 112

7.11 Polarization states for a finite line segment. This mask was laid out using polarizations that satisfy the Laplace equation. 113

7.12 Aerial intensity of the projected image of the mask in Figure 7.11. Slowly varying polarization states eliminate spurious intensity nulls. 114

7.13 Log scale intensity cross section through the center of the image in Figure 7.12. The null depth is reduced compared to the prior mask design. 115

7.14 Developed resist pattern resulting from a nominal exposure of the intensity distribution shown in Figure 7.12. The line ends are rounded and slightly bent. 116

7.15 Comparison of simulated linewidth scaling for an isolated interference null with the model derived in Chapter 3. The difference is likely due to iso-dense bias. 117

7.16 Polarization vectors to generate a circular phase null. 118

7.17 Projected intensity distribution of the mask in Figure 7.16. 119

7.18 Developed photoresist pattern for the projected intensity in Figure 7.17 119

7.19 Polarization states to generate a three way intersection ("T") pattern. 121

7.20 Projected intensity of the mask pattern from Figure 7.19 122

7.21 Developed resist patterns after exposure with the intensity of Figure 7.20. The lines all connect at the central intersection, although the optical contrast is poor, and with overexposure the lines terminate without connecting. 123

7.22 Polarization after mask to generate three alternating lines. 125

7.23 Projected intensity distribution for the mask shown in Figure 7.22. 126

7.24 Log scale intensity cross section through the center of the aerial intensity shown in Figure 7.23. The nulls each approximately follow the Sine Integral function. 127

7.25 Developed resist pattern resulting from a nominal dose delivered by the projected intensity in Figure 7.23 128

7.26 Polarization grating used as sub-contrast assist feature (SCAF) around a phase null. 130

7.27 Projected intensity of isolated line with SCAF 131

7.28 Intensity cross-section of Figure 7.27. The SCAF null contrast is only M = 0.25, and does not print in the resist. This allows an isolated line to image as a periodic structure. 132

7.29 Optical micrograph of grating lines in resist. Period is approximately 10 μm. 134

7.30 Optical micrograph of lines ends in resist. Period is approximately 10 μm. 135

7.31 Optical micrograph of connected lines (inverted pattern) in resist. Period is approximately 10 μm. 136

7.32 SEM image of overexposed grating pattern. 137

8.1 Schematic of proposed pixelated wiregrid polarizer phase masks. 139

A.1 An analytic model for resist thickness is overplotted on the piecewise approximation used in the main text. As shown, both models yield similar contrast. A key difference is that in the analytic model, the resist thickness begins to decrease at doses below D_1. 144

Chapter 1

Introduction

Photolithography is a process that transfers a two dimensional pattern onto the surface of a substrate via optical exposure (Figure 1.1). In many ways, photolithography is analogous to print making in film photography. A photographic print starts as a negative image on film. The film image is magnified and projected onto a paper coated in photosensitive emulsion, which reacts to the incident light. After exposure, the emulsion is chemically developed, revealing the projected image. Likewise, photolithography starts with an image on a mask. This mask image is *demagnified* and projected onto a substrate coated in photosensitive resin, called photoresist. The exposed photoresist is then chemically developed, revealing the latent image by removing resin. Remaining resist acts as a stencil on the substrate for subsequent processing steps.

Photolithography is among the most important industrial processes in use today as it is a key enabling process for fabricating integrated circuits; all modern electronics rely on photolithography. Semiconductor fabrication is driven by two scaling laws: Moore's Law and Dennard Scaling. Moore's law states that the transistor count on a dense integrated circuit doubles every two years [39]. As transistor density, and count, increases, total cost remains fixed. A more recent formulation of Moore's law says the the cost per function cuts in half every 18 months [29]. Not a true law, Moore's law is the road map used by in industry. Dennard scaling relates MOSFET performance characteristics to MOSFET size. Decreasing MOSFET gate length improves key performance metrics. Smaller MOSFETs have reduced gate voltage, power consumption, gate capacitance, and switching time [10][3]. Smaller transistors enable a continuation of Moore's law. Chasing smaller

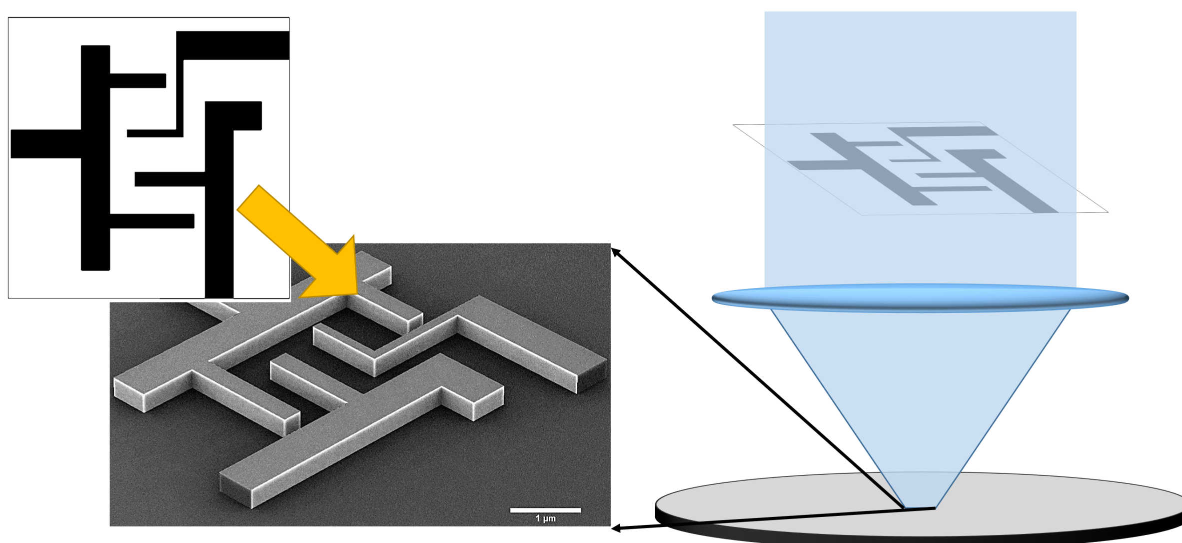

Figure 1.1: Illustration of photolithography. A two-dimensional pattern (upper left) is transferred onto the surface of a substrate (lower left). This is accomplished by fabricating a mask with the desired pattern, then projecting a demagnified image of the mask into a light sensitive resin, photoresist (right). This photoresist is then developed leaving the desired pattern. The remaining photoresist acts as a stencil for subsequent processing of the substrate surface.

features in an effort to improve performance and bring down device cost, leads to an endless pursuit of improvements in lithographic resolution. Fabricating smaller transistors requires improving both minimum feature size and aerial density, while maintaining high areal throughput in order to be commercially viable.

Lithographic techniques can be broadly classified as either serial write or parallel write processes. Serial write processes enable patterning down to single atoms, however, throughput for these processes is too low to produce meaningful quantities of complete devices. Parallel write processes lack the resolution of serial write, however, parallel write does have sufficient throughput to enable high volume manufacturing. In particular, photolithography has sufficient throughput for economic viability, but the write resolution is limited by diffraction. There is thus strong motivation to bring the arbitrarily small feature size of serial techniques to high speed parallel write optical lithography by resolving features with dimensions far below the half-pitch diffraction limit. Doing so would enable a "best of both worlds" method with high critical dimension resolution typical of serial write tools, such as e-beam, along with high throughput and low process costs typical of parallel photolithography.

Techniques previously investigated to accomplish this goal can generally be classified as optical near-field or multi-wavelength. Near-field approaches, including evanescent lithography [36], near-field scanning optical microscopy (NSOM) [47] and plasmonic masks [58], transmit light through a nano-scale aperture to generate optical patterns below the diffraction limit that are then evanescently coupled to the target resist. Since both the transmission and coupling processes suffer from very low efficiency, these methods do not retain the throughput advantage, even when used in parallel [36].

Multiple wavelength techniques [11] [48] [25] [2] overcome the near-field efficiency limit by using far-field focused beams to create a nano-scale mask in a custom, nonlinear resist. Inspired by STED microscopy [18], a first wavelength sensitizes the film in a diffraction limited focus, while a second wavelength shaped with a dark null inhibits response outside of the null [48]. The second wavelength thus reduces the feature size below the diffraction limit as the photopolymer is

overexposed, while the first wavelength locally restores resist sensitivity to enable resolution below the diffraction limit, effectively performing multiple patterning in situ. Unfortunately, this serial write process suffers from low throughput. Another drawback is that these custom negative-tone photopolymer resists have all shown low pattern fidelity [2].

This thesis develops photlithographic resolution enhancements, enabling sub-diffraction limited feature sizes over a large area, in the far field, using single wavelength, i-line (365 nm wavelength) illumination. This is accomplished both with a novel mask technology as well as necessary accompanying processes and models. The result is a resolution enhancement compatible with off-the-shelf resists, guaranteeing compatibility with existing CMOS foundries. I show that non-linear optical response inherent to positive photoresist enables a large reduction of critical dimensions, similar to those demonstrated with STED-lithography, while maintaining high throughput and high pattern fidelity. A high contrast pattern can be projected and the resist overexposed such that, after development, resist only remains in sub-diffraction sized regions confined within the dark nulls. Critical dimension scales approximately as the square root of exposure dose, as is the case for STED [16], enabling deterministic dimensional control. The one constraint is that feature spacing within a single exposure is limited by diffraction. Through the use of standard multiple patterning methods, such as litho-etch-litho-etch (LELE) [17], this limit on feature spacing can be violated, enabling both critical dimension and feature spacing below the diffraction limit while retaining the advantages of large field size exposed in seconds.

To generate arbitrary high contrast images, a novel phase mask, the pixelated polarization phase shifting mask (P^3SM), is developed. This is a lithographic mask enabling local polarization control at the mask plane. Projection lithography simulations with P^3SM show that these masks enable the arbitrary two-dimensional patterns with sufficient contrast for writing sub-diffraction limited critical dimensions. At the same time, P^3SM solves two key challenges in phase mask implementation; the line end problem, and the phase conflict problem. I show how patterns not previously possible in alternating phase masks can now be implemented, enabled by local polarization control, and propose a simple design methodology for P^3SM.

Process models are developed for quantitative predictions of linewidth scaling, minimum linewidth, and an important aerial image fidelity metric. These models are experimentally verified using i-line ($\lambda = 364$ nm) interference lithography and two different commercial NUV photoresists. These experiments confirm the models while demonstrating sub-wavelength parallel-write lithography over large areas. Critical dimensions as small as 50 nm are written with am aspect ratio of 3:1 and a line edge roughness of 5 nm. These results are comparable with both 193 nm ArF immersion steppers, at a fraction of the cost, and EBL, at a fraction of the write time. Finally, P^3SM are fabricated and are demonstrated using an i-line stepper.

This thesis is organized as follows: Chapter 2 provides a brief introduction to photolithography. Chapter 3 introduces the super-resolution technique and develops models to predict and understand performance limits. Chapter 4 introduces laser interference lithography, and details the LIL tool developed for process development and model validation. Chapter 5 presents the experimental results from LIL and model validation. Chapter 6 details the stepper design used for mask projection experiments. Chapter 7 introduces the pixelated polarization phase masks, design methodology, simulation and experimental results. Chapter 8 provides direction for ongoing and future work.

Chapter 2

Background

2.1 Lithography

Lithography, derived from ancient Greek, means "to write on stones." Micro- and nanolithography involve controlled patterning on, a surface at micron or nanometer length scales. Within this thesis, both micro- and nanolithography will simply be referred to as lithography.

A complete, generic lithographic process is illustrated in Figure 2.1, and described as follows: The first step prepares the substrate surface for coating with a resist. This surface preparation includes cleaning and drying the surface, as well as application of adhesion promoters. The second step deposits resist onto the substrate via spin coating. After spin coating, the resist is baked to remove excess solvent. The third step exposes the resist coated surface with the desired pattern, altering the resist and leaving a latent image. Typically, another bake cycle is applied after exposure. The fourth step develops the exposed resist to remove undesired regions, revealing the latent image. Finally, the last step uses the patterned resist as a mask to process the exposed substrate. Typical processes that follow resist patterning include wet or dry etch, deposition, and implantation.

Many lithographic processes have been developed, and this vast array of lithographic patterning methods can be broadly categorized either as serial write or parallel write processes. Lithographic processes classified as serial write, write one single pixel at a time. Lithographic processes classified as parallel write, write a full field simultaneously.

Serial write lithography spans a wide range of technologies and patterning scales. Representative approaches include μm-scale dimensions written with laser direct write, 10's of nm dimensions

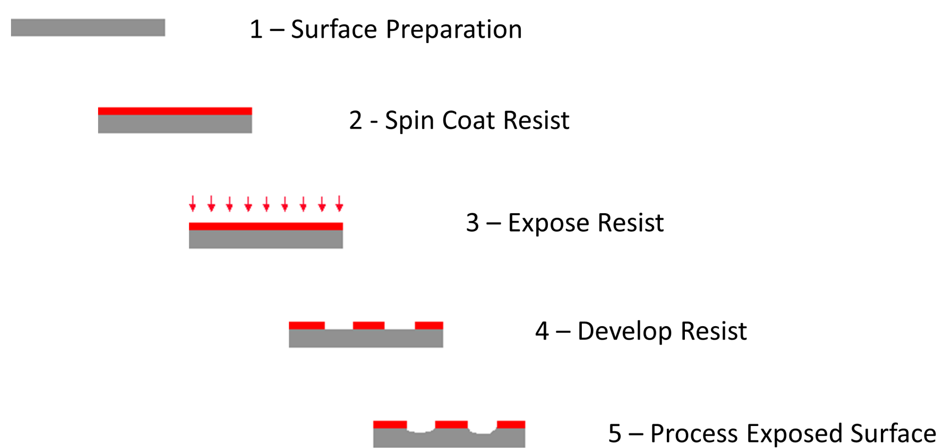

Figure 2.1: A generic lithography process. First, a wafer surface is prepared by cleaning, drying, and applying adhesion promoter. Second, a resist is spin coated onto the wafer, and baked. Third, the resist is exposed to a desired pattern, leaving a latent image in the resist. Fourth, the resist is developed, revealing the latent image. Fifth, the wafer surface is processed using the resist pattern as a stencil. Adapted from [33]

written with near field optical methods, nm dimensions written with e-beam lithography (EBL), and single angstrom features written with scanning tunneling microscopy (STM). Unfortunately, this resolution comes at the expense of field size and areal throughput; regardless of the technology used, serial write areal throughput scales as the fifth power of resolution, known as Tennant's Law [52]. Figure 2.2 plots resolution and pixel throughput for serial write processes. Pixel throughput differs from areal throughput by a factor of d^2. On this same plot, several representative serial lithographic processes are identified at their respective pattern length scales. Given this, 10 nm resolution EBL of a 1 mm^2 field area requires approximately 8 hours to complete [42]. Patterns that are larger than this scannable field area must be stitched via stages which introduce inevitable pattern distortion. The tool cost is usually significantly lower and the process cost per device is significantly higher than parallel methods, motivating the use of serial write lithography for research applications where a small number of devices are typically needed.

Parallel write lithography is dominated by mask-based photolithography. Parallel write methods contrast with serial write methods in that pitch resolution typically has a fixed lower limit, but over large write areas and high areal throughput. Amazingly, improvements in parallel write resolution are beneficial to throughput. Photolithography has followed Moore's Law which states that the areal throughput *increases* as the fifth power of resolution. This improvement has been driven by considerable innovation and investment by the semiconductor industry. Figure 2.2 illustrates the relation between resolution and pixel throughput for parallel write, with several generations of wavelength identified at their respective resolution length scales.

Although technically diverse, all of these lithographic processes can be compared using four key metrics [7]. The first metric is resolution, which encompasses both minimum feature size (critical dimensions) as well as minimum feature spacing (pattern pitch). The second metric is throughput, which is conveniently described by the total field size and the field write rate. The third metric is pattern fidelity, measured by local metrics, such as line edge roughness (LER), as well as global metrics, such as pattern distortion. The final metric is tool cost and process, which vary widely, driven by both resolution and throughput. Each metric's importance depends on the

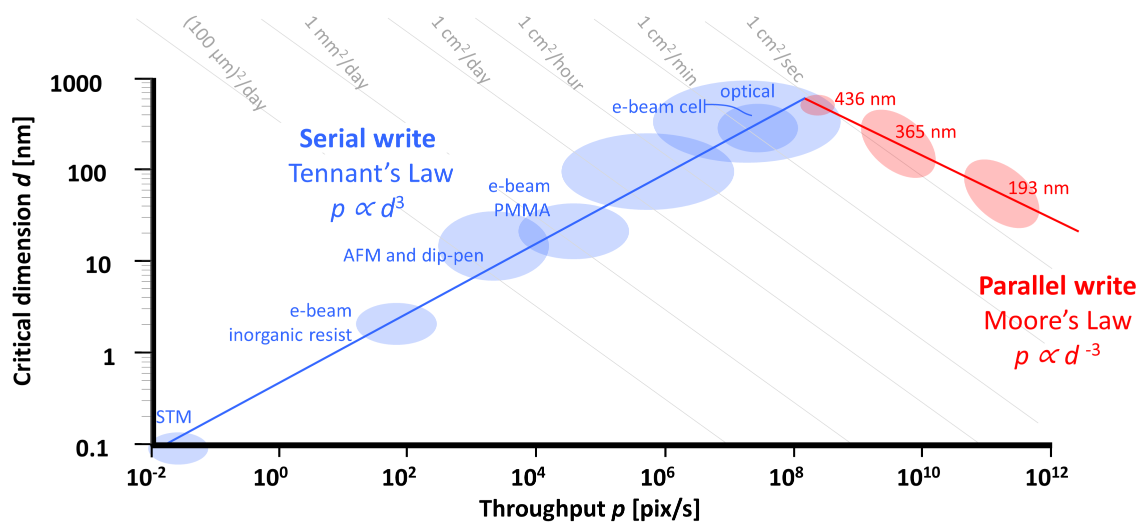

Figure 2.2: Resolution and throughput for various lithographic processes. Throughput for serial write lithography (blue) slows as critical dimension is reduced, as described by Tennant's Law. Several types of serial write are identified at their typical critical dimension length scales. Throughput for parallel write lithography (red) accelerates as critical dimension is reduced, as described by Moore's Law. Several generations of optical lithography are identified along the curve, labeled by the wavelengths used.

application; for example, commercial manufacturing tools are typically high cost in order to reach high throughput. In the following sections, these metrics are considered in detail with respect to photolithography.

2.2 Resolution

There are two resolutions in lithography - pitch spacing and critical dimensions (CDs) illustrated in Figure 2.3 [34]. Pitch spacing is the minimum spacing between two uniquely resolvable features. CD is the size of individual features. While related, pitch resolution and CD resolution are bounded by different limits. Pitch resolution for photolithography is enforced by diffraction [42] with the lower bound given by the Rayleigh diffraction limit

$$\Lambda = \frac{\lambda}{2NA}. \tag{2.1}$$

Here, Λ is the minimum pitch spacing, set by optical wavelength λ, and numerical aperture NA. However, the minimum CD is ultimately determined by process capability. This process capability can be expressed in a coefficient k_1 [26] [24]. Using this process coefficient, CD resolution can be expressed as

$$CD_{min} = k_1 \frac{\lambda}{NA}. \tag{2.2}$$

While wavelength and NA still play large roles, the process related term, k_1 also limits the lower bound. A process able to print equal width line-space patterns, will have $k_1 = \frac{1}{4}$.

Resolution improvements for both pitch and CD have typically been achieved by reducing exposure wavelength, by developing higher NA projection optics, and by improving photoresist chemistry [7]. Diffraction-limited pitch resolutions have dropped from several μm using near ultraviolet (NUV) and contact masks down to 45 nm using 193 nm deep UV (DUV) ArF sources and projection with high NA immersion optics [42]. Minimum CD resolutions have decreased from μms to 10 nm as k_1 has decreased [40] from ≈ 1 to ≈ 0.1. Commercial foundries have improved

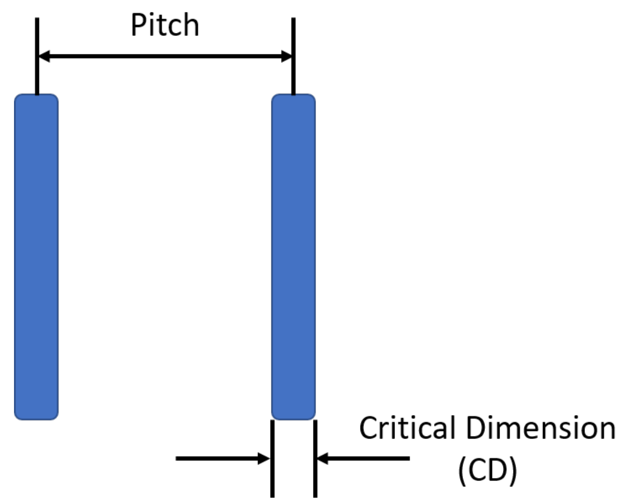

Figure 2.3: In lithography, there are two distinct resolutions: pitch resolution and critical dimensions (CD). Pitch resolution is the minimum distance between uniquely resolvable features. CD is the size, or width, of individual features.

pitch resolution beyond the diffraction limit via multiple patterning methods, albeit at the cost of throughput. Multiple patterning techniques now enable foundries to reach $\approx$ 10nm resolution with ArF immersion tools [17].

2.3 Throughput

The throughput of modern photolithographic tools is without equal: commercial steppers routinely write field sizes as large as 8 cm^2 in under one second [6]. Photolithographic tooling costs are high, but the very high throughput offsets this cost as foundries can amortize these costs over many devices. Unfortunately, this amortization is not possible outside of large commercial semiconductor applications. There are many use cases requiring high resolution, but not high throughput. In these cases, resolution enhancement at the cost of throughput is still beneficial.

2.4 Pattern Fidelity

Fidelity of developed resist patterns is an incredibly broad metric, encompassing both local and global metrics. Global metrics include measures of pattern distortion and overlay error. Local metrics include line edge roughness (LER) and linewidth roughness (LWR). In this thesis, the analysis of pattern fidelity is focused on LER for the developed lithographic patterns. LER is defined as the high spatial frequency deviation of an edge from an ideal line. LER is measured from a top-down image of developed features. The deviation of edge position relative to an ideal line is computed for every point along the edge, and LER is reported as the 3σ standard deviation of edge deviations. LER is problematic when the edge roughness approaches the length scale of the CDs. In this case, the line width varies considerably as a result of edge variations. Typically, lithographers look to achieve LER $\leq$ CD/10. Many factors can influence LER: resist composition, developer chemistry, resist processing, and aerial image all impact the final LER.

The focus in this thesis is on the impact of aerial image on LER. The impact of aerial image on LER can be understood using an image quality metric known as Normalized Image Log Slope (NILS). NILS can be used to predict both CD uniformity and LER, and is defined as

$$NILS = w\frac{d}{dx}\left(Log\left[I\left(\frac{w}{2}\right)\right]\right) = w\frac{\frac{dI}{dx}}{I(\frac{w}{2})}. \tag{2.3}$$

Here, I is the aerial image intensity and w is local linewidth. NILS is the ratio of the image intensity slope over the image intensity, evaluated at the line edge, further normalized by the linewidth. Another way to think about NILS is as a ratio of fractional change in intensity to fractional change in linewidth

$$NILS = \frac{dI}{I}\left(\frac{dx}{w}\right)^{-1}. \tag{2.4}$$

Thought of in this way, NILS is simply feature size sensitivity to intensity variations. Well defined edges within an aerial image result in small changes in CD with respect to a change of intensity (or dose). A large value for NILS indicates that the aerial image of the line is sufficiently sharp for good recording. Large NILS also imply better uniformity, lower LER, and larger process windows. As a result, NILS is one of the most important image quality metrics in photolithography. Despite this importance, NILS has not yet been addressed in studies of STED lithography.

2.5 Photoresists

Photoresists can be broadly categorized as either positive tone or negative tone. Positive tone resists, when developed, are removed from exposed regions and remain in unexposed regions. Dark regions on a mask correspond to the developed resist pattern. Negative tone resists, when developed, are removed from unexposed regions and remain in exposed regions. Clear regions on a mask correspond to the developed resist pattern, hence the negative classification.

This thesis is focused on positive tone i-line resists; negative tone resists are not typically used for high resolution lithography. Typical i-line positive tone resist is composed of a resin and a photoacid generator (PAG). Most positive tone resists designed for use in the near-UV use a Novolak resin and a diazonaphthoquinone (DNQ) PAG, shown in Figure 2.4a and Figure 2.4b respectively. Novolak dissolves in an aqueous base. Novolak containing DNQ dissolves even more

slowly in an aqueous base. Novolak containing an acid dissolves much faster when submerged in an aqueous base. DNQ (PAG) inhibits dissolution, and photo-generated acid accelerates dissolution. The large difference in dissolution rates between exposed regions containing a photogenerated acid and unexposed regions containing DNQ enables spatial patterning via optical exposure.

Resist is typically applied via spin coating. Spin coating spreads a viscous fluid via centrifugal force over a flat surface to produce a thin, uniform film. Film thickness is controlled by rotations per minute (RPM) and fluid viscosity. In order to spin coat, photoresist is dissolved into a solvent which acts as a delivery vehicle for the resin. After spin coating, photoresist is baked to drive out excess solvent, although this bake also removes water from the film. To produce a photogenerated acid, DNQ undergoes a two step reaction ending in a carboxylic acid, shown in Figure 2.4c. The first step involves absorption of a UV photon, cleaving off N_2. The second step involves dissolution in water to form carboxylic acid. Therefore, to properly expose, positive tone resists must be properly re-hydrated.

After baking, resist re-hydrates by absorbing water from the air, eventually reaching an equilibrium with the surrounding atmosphere. Too low of a water content inhibits development while too high of a water content reduces adhesion with the substrate. This makes humidity control important. For proper exposure, without loss of adhesion, appropriate humidity levels are typically between 40% - 50% RH [57]. During this work, I have found that humidity levels as low as 35% RH are acceptable. Rehydration time depends on film thickness; the absorption is diffusion limited [57]. Thin films, less than a micron, may require less than one minute, while thick films, e.g. 100 microns, may require days [57].

Typically, a post-exposure bake is applied before developing [24]. In a Novolak based resist, this bake step leads to diffusion of photogenerated acid, smoothing out features and reducing line edge roughness [34]. This is especially useful in minimizing standing wave patterns due to reflections during exposure [24][34].

Novolak resists are developed in aqueous bases. Older processes used potassium hydroxide (KOH), although the current standard is tetra methyl ammonium hydroxide (TMAH). TMAH is

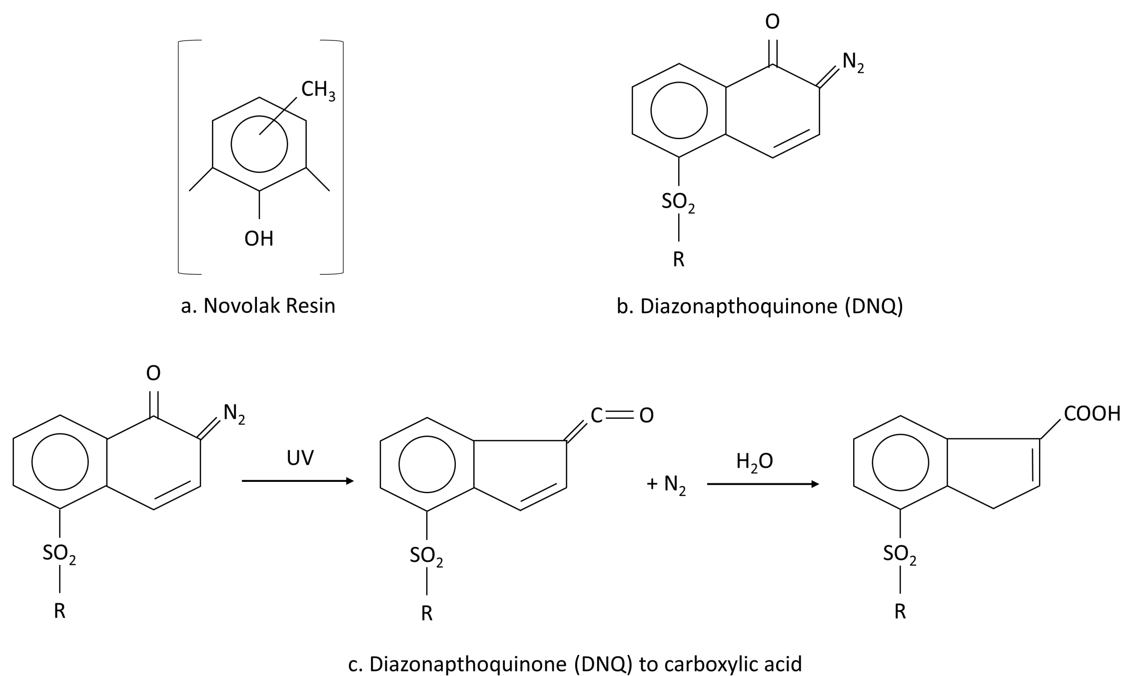

Figure 2.4: Chemical structures for i-line photoresist components. a) Novolak resin b) Diazonapthoquinone (DNQ) a photoacid generator c) photogenerated acid reaction, DNQ to a carboxylic acid

preferred because it is metal-ion-free and yields lower line edge roughness [24]. Development rate is temperature dependent, so developer temperature should be well controlled [24]. Typically, when tuning a lithographic process, development time and temperature are held constant while other process parameters such as soft-bake, exposure dose, and post-exposure bake are varied.

A transfer function can be used to describe the response of positive tone photoresist to optical dose. Normalized remaining resist thickness after development is given as a function of exposure dose. This response curve is determined empirically for each resist formulation. A typical response curve plot, with three key parameters marked, is given in Figure 2.5. The first parameter is lower dose threshold D_1. D_1 is the largest exposure dose for which development doesn't initiate. The second parameter is upper dose threshold D_2. D_2 is the smallest dose for which development will complete. The third parameter is resist contrast γ. γ is the log slope between thresholds. Low contrast indicates near-linear optical response, while high contrast indicates highly non-linear optical response. Additionally, contrast is related to sidewall angle of the developed patterns [33].

2.6 Standing Waves and Anti-Reflection Coatings

Photoresist records all present light, both incident illumination as well as back reflections from the substrate. These reflections lead to standing wave patterns in the resist which manifest as scalloped side walls [34]. As feature size shrinks, these scalloped sidewalls become increasingly problematic. To eliminate standing waves, a bottom anti-reflection coating (BARCs) can be applied prior to applying photoresist [34][24]. BARCs are polymers that are index matched to resist and contain non-bleachable absorbers. Incorporating a BARC eliminates backside reflections and standing waves within the resist. In this thesis, instead of applying a BARC, absorptive neutral density filters are used as substrates. These glass substrates are well matched to the resist index in the NUV, and provide excellent reflection suppression.

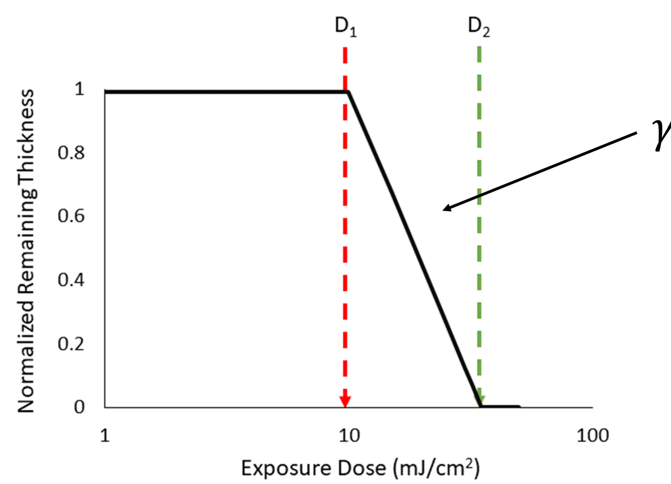

Figure 2.5: Typical response curve for positive tone photoresist. Below dose threshold D_1, no resist is removed after development. Above dose threshold D_2, all resist is removed upon development. The slope of the response curve, γ, is the resist contrast. Large γ implies a strong non-linear response, and results in steep sidewalls.

2.7 Photomasks

Master lithographic patterns are called photomasks. Classic photomasks are made of chrome patterns on glass plates, modulating transmitted electric field amplitude. An early lithographic process, contact lithography, placed the masks in direct contact with the wafer, then illuminated with a UV lamp. Contact lithography requires mask patterns to be at the intended final size as there is no demagnification. While simple to implement, contact mask lifetime is short due to wear caused by repeated wafer contact. To solve this, proximity lithography adds a small gap between the mask and the wafer. Both contact lithography and proximity lithography write in the near field, requiring the near field to reach the bottom of the photoresist film. Optical near-fields span a distance given by

$$L = \frac{w^2}{\lambda}, \tag{2.5}$$

where L is the distance to the far-field, w is the feature (aperture) size, and λ is the wavelength. The mask features (apertures) must be large enough so that the Fresnel zone reaches the bottom of the photoresist. Simple rearrangement of Equation 2.5 gives the mask resolution limit of

$$w \geq \sqrt{L\lambda}. \tag{2.6}$$

Here, w is the smallest critical dimension on the mask. Resolution for proximity lithography is lower than resolution for contact lithography due to the added gap distance.

2.8 Projection Lithography

Instead of placing a mask in contact with a wafer, an image of the mask can be projected onto the wafer. Imaging eliminates mask deterioration, enables demagnification, reduces process time, and enables Rayleigh limited resolution. Demagnification eases mask fabrication as mask features can be many times larger than the final printed features. Other key imaging parameters

are depth of focus (DOF) and spatial coherence. Depth of focus is given by

$$DOF = k_2 \frac{\lambda}{NA^2}. \tag{2.7}$$

Here, k_2 is a process dependent term. The nominal value is $k_2 = 2$ for a perfect process. Lithographers describe spatial coherence in terms of a coherence factor, σ. This factor

$$\sigma = \frac{NA_{illum}}{NA_{imaging}} \tag{2.8}$$

is the ratio of the illumination NA to the entrance pupil NA of the projection optics. Perfect coherence, $\sigma = 0$, requires a perfectly collimated illuminating beam, a point source in Fourier space. Totally incoherent illumination $\sigma \geq 1$ fills the pupil in Fourier space.

2.9 Projection Tools

Imaging systems that expose a full field at once, then step-and-repeat across a wafer are called steppers. Imaging systems that expose a narrow slit while scanning both mask and wafer, then repeat across a wafer (scan-and-repeat) are called scanners. The most advanced projection tools today are scanners. Steppers, though older, are still in use today and typically have a demagnification of 2-4x, limited by the mask size. For an equivalent exposure field area, a larger demagnification requires a larger mask. Practical mask sizes are limited by issues such as thermal expansion and mechanical deformation. All steppers consist of at least four major subsystems: Illumination, projection optics, reticle and wafer mechanics, and control system.

The optical subsystems are analogous to those found in microscopes. An illumination system consists of a light source, conditioning optics, and a condenser. Conditioning optics homogenize the intensity, control polarization, control illumination NA, and create patterned illumination. Projection optics project a demagnified image of the reticle onto the wafer. These lenses are designed for high resolution, low distortion imaging. In addition, optical systems are used for focus detection and mask-to-wafer alignment.

Key mechanical systems are responsible for holding and positioning the reticle and the wafer. The mechanics must move with high precision in at least three axis for focus and alignment and hold position without vibration. Wafer stages must also move fast for high throughput. A control system monitors and drives all systems of the lithographic tool.

2.10 Resolution Enhancement Techniques

To extend the capability of lithographic tools, a number of resolution enhancement techniques (RETs) have been developed. Resolution enhancements can serve several purposes. First, they can be used to improve developed pattern accuracy and fidelity. Second, they can be used to improve critical dimension resolution. Third, they can be used to improve pitch resolution.

Optical proximity correction (OPC) is used to improve final pattern fidelity [34][24]. Adjustments made to mask patterns can improve the geometry of developed resist patterns to better match the nominal design. For example, developed pattern dimensions are compensated by biasing feature sizes on the mask [34][24]. Square corners and line ends are improved by adding serifs to the mask [34][24]. Sub-resolution assist features are added around isolated features to approximate dense features [34][24].

Off-Axis Illumination (OAI) is used to improve pitch resolution [34][24]. Illuminating the mask at non-normal incidence, different diffraction orders from the mask enter the projection system pupil [34][24]. This enables two-beam imaging, effectively doubling pitch resolution. A drawback to OAI is that it may require pattern decomposition into multiple masks to maximize image contrast. OAI may also include shaped illumination optimized for each mask [34][24]. Polarization control is used in conjunction with OAI and shaped illumination [49][1]. For example, dipole illumination requires s-polarization to maximize image contrast. More recently, retarders have been incorporated into shaped illumination masks, allowing arbitrary polarized shaped illumination patterns to further improve image contrast [27][15][5].

Finally, phase shifting masks (PSM) improve feature size and contrast for all pattern densities. PSMs are the most promising RET, and are covered in detail in Section 2.11.

2.11 Phase Masks

Adding phase discontinuity to masks creates localized interference that can be exploited to improve image contrast[34][23][28][22]. A comparison of traditional chrome masks and phase shifting masks is given in Figure 2.11. As feature pitch on a traditional chrome mask decreases, the point spread function of each feature begins to overlap. Decreasing feature pitch leads to decreasing image contrast. However, incorporating alternating π phase steps forces the image electric field to cross zero, maximizing image contrast, regardless of feature pitch. Phase masks may incorporate chrome patterns, or may be chrome-less [34][53][12][4].

While PSM is the most promising RET, there are severe limitations in implementation. One such limitation is that to maximize image contrast, the polarization must be locally aligned to the phase step and is detailed in Chapter 7. To maximize contrast in multiple orientations requires mask decomposition and multiple patterning. A second limitation is that traditional phase masks suffer from a line end problem. This problem is illustrated in Figure 2.7A. It is not clear how to end a line segment with alternating phase to either end. A third imitation is that traditional phase masks have severe phase conflict problems. This is analogous to the map coloring problem with only two colors. A typical phase conflict is illustrated in Figure 2.7B. To implement phase shifting, design rules and chip layouts must be redesigned to accommodate the restrictions of phase shifting masks [34].

A novel pixelated polarization phase shifting mask (P^3SM) is developed in this thesis that overcomes all three of these limitations.

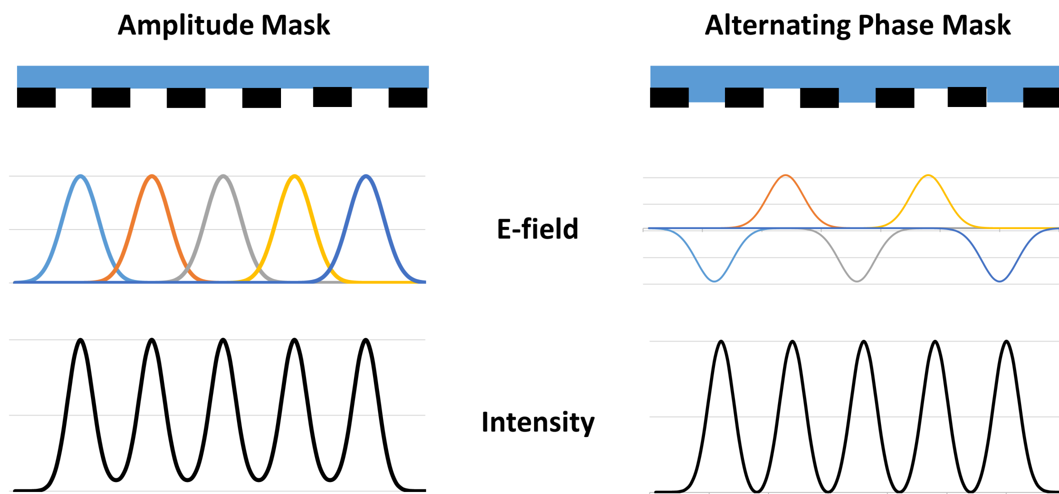

Figure 2.6: Comparison of chrome masks with phase shifting masks. Chrome masks modulate the electric field amplitude only. As pitch is reduced, images from adjacent apertures overlap, reducing the final image intensity. Phase shifting masks modulate electric field phase with π phase steps, and can incorporate chrome patterns as well to modulate amplitude. Shifting the phase in alternating mask regions forces a zero crossing of the electric field, thereby improving image contrast.

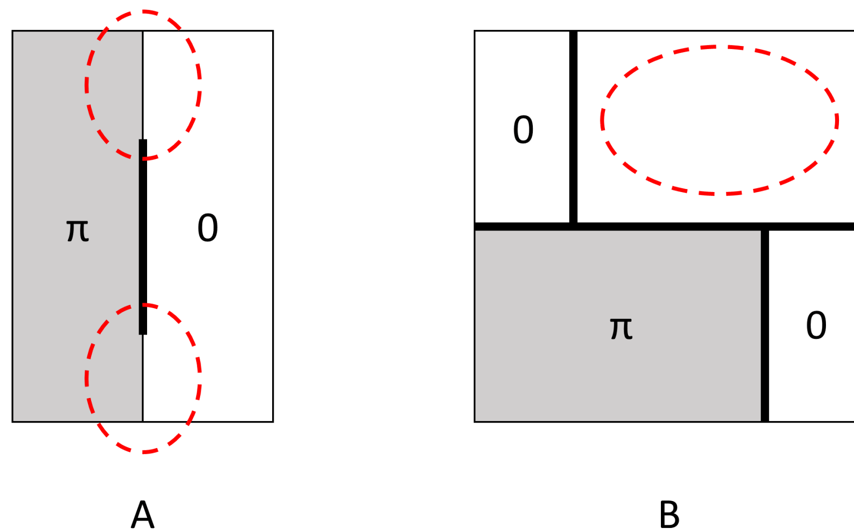

Figure 2.7: Two significant problems encountered with phase shifting masks. A) A finite line segment (dark black line) cannot be terminated when alternating phases span the line. Dashed red circles indicate regions where a dark null will still form. B) A typical phase conflict problem. A phase cannot be determined for the region circled in red. A π phase will not give a null adjacent to another π phase. Zero phase step has the same problem with adjacent zero phase regions.

Chapter 3

Super-Resolution Lithography

3.1 Ideal Super Resolution lithography

An ideal super-resolution lithography process overcomes the limitations of STED lithography by implementing two major constraints. First, ideal SR lithography uses only a single wavelength to record patterns making the process compatible with existing lithographic tools and resists. Second, ideal SR lithography uses only commercial off-the-shelf materials. Together, these constraints eliminate novel resist development and guarantee compatibility with CMOS fabs. Ideal SR lithography should be capable of producing deep sub-wavelength CDs, with deterministic control and high fidelity. Super-resolution lithography as developed within this thesis satisfies these requirements.

3.2 Super-Localized, Single Wavelength, Broad Area Lithography

The process presented here relies on the non-linear optical response of photoresist in order to super-localize pattern dimensions. This is analogous to STED microscopy, which relies on the non-linear optical response of fluorophores in order to localize response [44]. After optical exposure, photoresist is chemically developed to reveal the latent image, and the remaining thickness is a non-linear function of exposure dose, shown in the inset Figure 3.1. The inset plots a piecewise continuous non-linear model for resist thickness against dose on a log scale. This model is described in equation 3.4. Regions where exposure dose falls short of the lower threshold are left undeveloped. While some dark erosion may occur in unexposed regions, dark erosion is assumed to be negligible for simplicity of discussion. Where exposure doses exceed an upper threshold,

photoresist is completely removed. Intermediate doses cause a reduction in thickness that scales as Log(dose). Overexposure results in decreasing linewidth, as a result of the dose threshold required for development as illustrated in Figure 3.1. Also shown in Figure 3.1 are sloped sidewalls for each resist cross section. To remove ambiguity, in this thesis, linewidth is defined as the width at the base (widest point) of each feature.

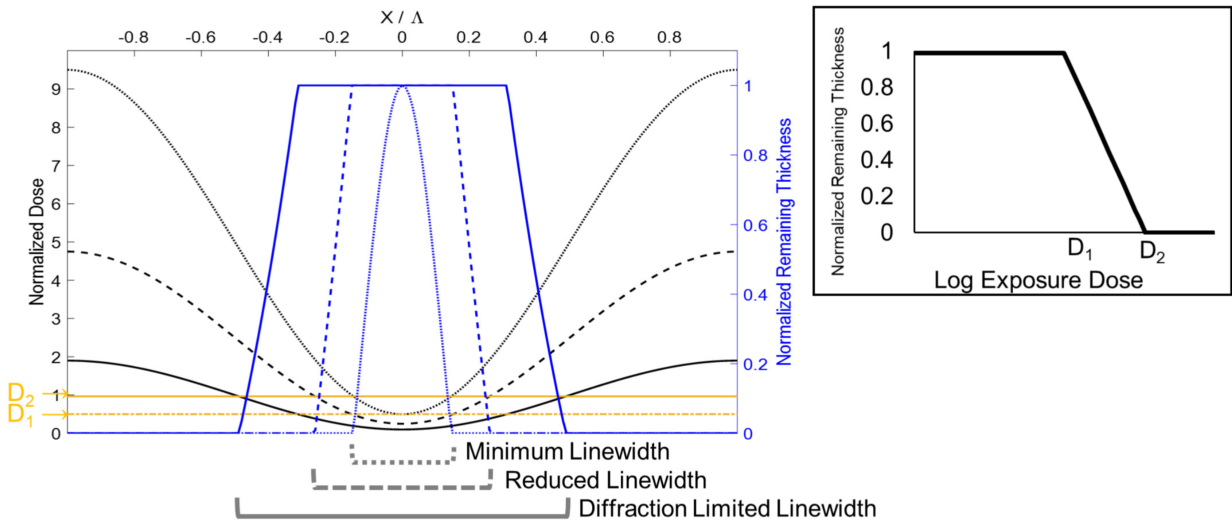

Figure 3.1: A plot of both the aerial image intensity and resulting developed photoresist cross sections. The projected images are all of finite image contrast (M=0.9). A nominal exposure produces a linewidth equal to half of the period. Maximum exposure dose leaves full thickness at the center of the feature. In the insert, a piecewise continuous model for the non-linear transfer function, plotted against Log(dose).

3.3 Process Modeling

A benefit to using well-characterized photoresist as a recording material is the potential to precisely predict sub-diffraction feature size and pattern fidelity. Using a two-beam interference model, I establish the dominant limits on required contrast of the areal image. Then, I derive how linewidth is expected to scale when exposed with finite contrast and as a function of over-exposure. With this model, I show that easily attainable resist and optical attributes enable five-fold violation of the diffraction limit. Finally, using a common lithography metric, normalized image log slope

(NILS), I predict that LER will be small and the process window large.

3.3.1 Aerial image contrast

The aerial pattern of a periodic line pattern imaged at the diffraction limit of a projection lens can be locally described by the interference of two plane waves emerging from opposite edges of the pupil. Minimum feature pitch (resolution), Λ, is then limited by exposure wavelength, λ, and the numerical aperture, NA, by $\Lambda = \frac{\lambda}{2\sin(\theta)}$. The areal image intensity is then given by

$$I = I_1 + I_2 + 2G_{12}\sqrt{I_1 I_2}\cos((\vec{k}_1 - \vec{k}_2)\vec{r}) < \hat{e}_1 \cdot \hat{e}_2 > \tag{3.1}$$

where I_i, $\vec{k}_i$, and $\hat{e}_i$, are the intensities, wave numbers, and polarizations of the two plane waves $i = 1, 2$. G_{12} is the mutual coherence [46] given by

$$G_{12} = \frac{< E_1^* E_2 >}{\sqrt{I_1 I_2}}. \tag{3.2}$$

One important quality metric of this aerial pattern is the interference contrast, or fringe visibility M, defined as the peak-to-valley intensity difference over the peak-to-valley intensity sum,

$$M = \frac{I_{max} - I_{min}}{I_{max} + I_{min}}, \tag{3.3}$$

where maximum and minimum intensities are given by I_{max} and I_{min} respectively. Perfect contrast of $M = 1$ occurs only when the nulls of the interference pattern are completely dark, while $M = 0$ corresponds to a uniform intensity exposure. Perfect contrast can only occur for a particular polarization state, in this situation, s-polarization. In general, the desired polarization depends on the shape of the null; for example, isolated point nulls, as found in Laguerre-Gauss beams, require circular polarization consistent with the sign of the orbital angular momentum [54].

Analogous to STED, features are localized through over-exposure with a high-contrast inten-

sity pattern. Exposure and thus minimum feature size is limited by the accumulation of dose in the dark nulls of finite contrast. Equation 3.1 shows that contrast is linearly sensitive to the degree of polarization and the mutual coherence of the two beams. Conversely, imperfect intensity balance and polarization misalignment are comparatively weak effects as contrast decreases quadratically for both. These trends indicate that the proposed technique is compatible with existing photolithographic tools whose light sources have high polarization and spectral purity and whose projection optics provide at least modest levels of amplitude and polarization control.

Direct experimental validation of image contrast requires high dynamic range intensity measurements at the nm scale. One method, which has been used by STED microscopists, is to scan over a nanoparticle [16]. While effective, it is difficult to implement in a large area projection lithography tool. However, the analysis below reveals that the scaling of linewidth with exposure time (total dose) provides a robust, indirect measure of aerial pattern contrast.

3.3.2 Simplified Photoresist Model

Developed image size is constrained by both the finite contrast of the aerial image and the fact that photoresist response is not binary. For exposure doses less than a lower threshold, D_1, no photoresist is removed during development. Conversely for doses greater than an upper threshold, D_2, all photoresist is removed during development. Both threshold values are readily determined by experiment. Intermediate doses lead to partial resist removal during development. The slope of this intermediate dose region is conveniently defined as the photoresist contrast, $\gamma = \frac{1}{\log_{10}(\frac{D_2}{D_1})}$. For typical commercial resists, γ is in the range of 2 to 4 ($D_2/D_1 \approx 3.2$ to $D_2/D_1 \approx 1.8$) [8]. While analytic models for the resist thickness exist [30], given that there are three distinct dose-response regions, I approximate with a piecewise continuous model. The remaining photoresist thickness than given as

$$T(D)/T_0 = \begin{cases} 1 & D \leq D_1 \\ -\gamma Log_{10}(D) + \gamma Log_{10}(D_2) & D_1 < D < D_2 \\ 0 & D \geq D_2. \end{cases} \tag{3.4}$$

where $T(D)$ is the developed thickness after applied dose D, and T_0 is the initial resist thickness. This model is illustrated in the insert in Figure 3.1. A comparison of an analytic model with this approximation is given in Appendix A.1.

3.3.3 Linewidth Model

Taking into account both finite aerial image contrast as well as photoresist contrast, it is possible to derive a relation between expected linewidth and exposure dose. The equation for dose with period Λ, contrast M, and average dose D_A is given by

$$D(x) = D_A[1 - M\cos(\frac{2\pi}{\Lambda}x)]. \tag{3.5}$$

The average dose which yields a diffraction limited linewidth equal to half the image period is then found to be simply $D_A = D_2$. For convenience, I define a normalized average dose relative to this diffraction limited dose as $\bar{D} = D_A \,/\, D_2$. Experimentally, $\bar{D}$ is simply the ratio of the exposure time to the time that results in a diffraction limited pattern. Solving for normalized linewidth with this definition yields

$$\frac{L}{\Lambda} = \frac{1}{\pi}\arccos[\frac{1-\frac{1}{\bar{D}}}{M}]. \tag{3.6}$$

As shown in Figure 3.2, this equation reveals that linewidth depends on both applied dose and aerial image contrast. Linewidth drops to zero at a maximum dose determined only by image contrast, $\bar{D} \leq \frac{1}{1-M}$. As dose approaches this limit, linewidth falls infinitely fast and thus dose tolerance becomes infinitely tight. Thus, while dose near the image contrast limit appears to

provide arbitrarily small feature size, it is impractical to form a large area intensity pattern with arbitrarily uniform intensity.

The resist transfer function also prevents operation at this extreme. As illustrated in Figure 3.1, finite resist contrast will cause the center of the line to lose thickness for sufficiently large dose. Here, minimum linewidth is defined by the linewidth at the largest dose before thickness at line center begins to decay. The maximum dose enforced by the resist contrast is then found by setting dose in the null to the lower threshold D_1, yielding

$$\bar{D} \leq \frac{D_1}{D_2(1-M)} = \frac{1}{10^{1/\gamma}(1-M)}. \tag{3.7}$$

This equation reveals that maximum dose to maintain full resist thickness is limited by both resist contrast, γ, and aerial image contrast, M. Therefore, the lower limit on feature size is

$$\frac{L}{\Lambda} = \frac{1}{\pi} \arccos\left(\frac{1}{M} - \frac{1-M}{M} 10^{1/\gamma}\right). \tag{3.8}$$

These trends are illustrated in Figure 3.2. With perfect image contrast ($M = 1$), linewidth falls like $\sqrt{\bar{D}}$. Again there is a parallel with STED microscopy, where resolution also scales as $\sqrt{\bar{D}}$ [16]. For finite image contrast ($M < 1$), the linewidth falls faster than square root of dose, approaching infinitely fast reduction at a maximum normalized dose of $\frac{1}{1-M}$. Not only would operation here require impractical tolerances, the resist thickness also approaches zero which would impair subsequent process steps. The resist contrast thus enforces a stricter dose limit to maintain full thickness at line center. As shown, for typical resists with $\gamma \leq 3.5$, the maximum resist-limited exposure dose falls close to the knee of the linewidth curve for all practical image contrast.

This analysis shows that the nonlinearity inherent in conventional photoresist transfer functions enables significant reduction of critical dimensions below the half-pitch diffraction limit. Moderate aerial image contrast of $0.9 < \mathrm{M} < 0.99$ and photoresist contrast of $\gamma < 3.5$ enable violation of the diffraction limit by a factor of 3 to 11. The penalty is a loss of throughput due to greater exposure times by a factor of between 9 and 120. It is worth noting that photoresist contrast greater

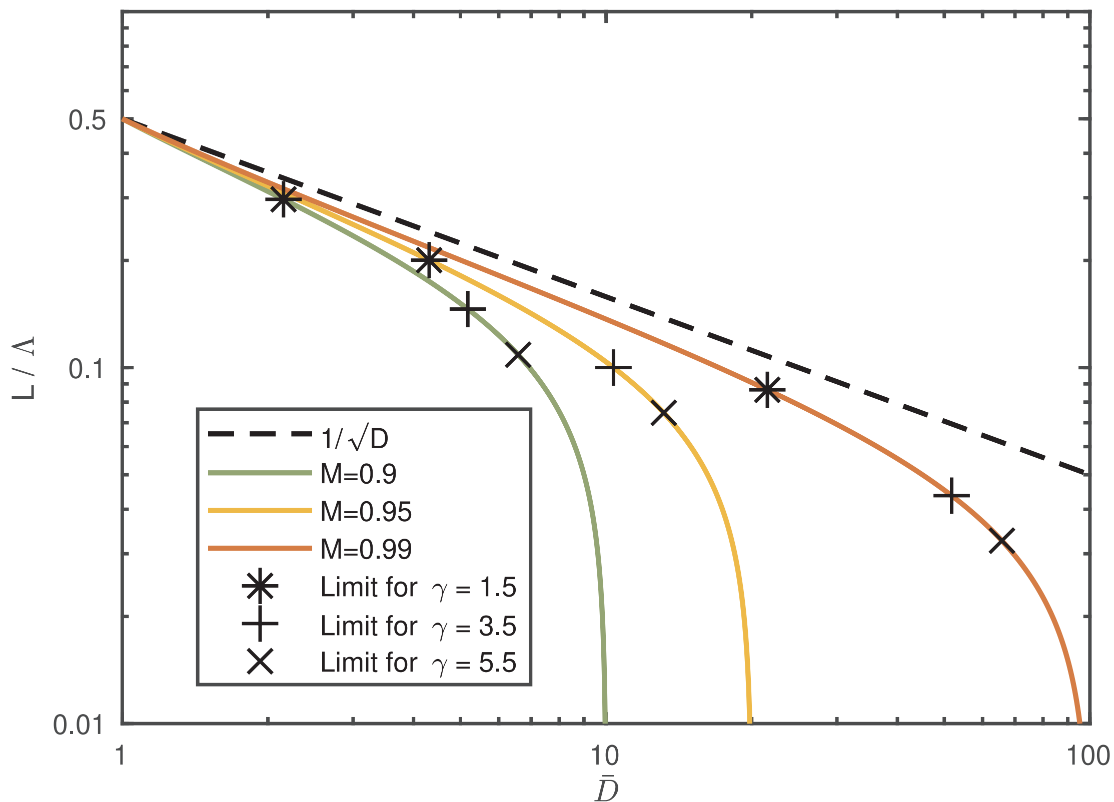

Figure 3.2: Plot of normalized linewidth as a function of normalized exposure dose. A range of practical image contrasts from $M = 0.9$ to $M = 0.99$ are shown. For all image contrasts, the nominal exposure ($\bar{D} = 1$) produces a diffraction-limited linewidth equal to half the period. Over-exposure given by $\bar{D} > 1$ reduces the linewidth below the diffraction limit. Feature size drops to zero at a dose determined by image contrast M, although, finite resist contrast γ prevents operation at this extreme.

than approximately 3.5 is unnecessary, as the image contrast then becomes the limiting factor in linewidth reduction.

3.3.4 Linewidth Scaling

Measuring the aerial image contrast is critical for linewidth control using the proposed process. From Figure 3.2, the dependence of linewidth on dose approximately follows a power law

$$\frac{L}{\Lambda} \propto \bar{D}^{-S} \tag{3.9}$$

$$S = \lim_{\alpha \to 1} log \frac{\frac{\bar{L}(\bar{D})}{\bar{L}(\alpha\bar{D})}}{log[\alpha]} = \frac{1}{\sqrt{M^2\bar{D}^2 - (\bar{D}-1)^2} \arccos\left(\frac{1-1/\bar{D}}{M}\right)}. \tag{3.10}$$

This deterministic scaling provides not only a means of experimental linewidth control, but also, as will be shown, a convenient method to measure image contrast. The theoretical scaling factor, S, is weakly dependent on normalized dose $\bar{D}$. To simplify the model so that S depends only on image contrast, a range of linewidths were computed as a function of dose and image contrast, then a power law was fit to the computed values. This result is plotted in Figure 3.3. As can be seen, for the case of ideal image contrast, $M = 1$, linewidth scaling rate $S = 0.5$, as expected. Decreasing image contrast below the ideal case, increases the scaling factor. Therefore, experimental linewidth as a function of dose can be simply fit to a power law, revealing the aerial image contrast via Equation 3.9. Spatial variations of S provide a map of M over the exposure area, independent of local average dose, D_a.

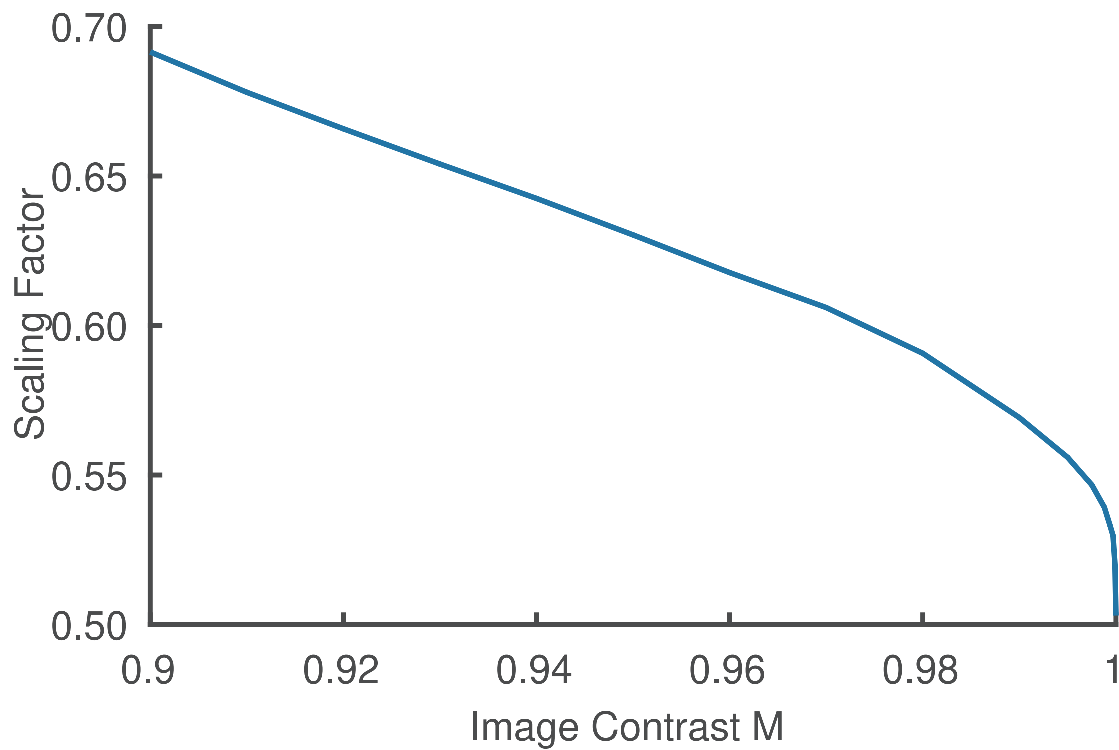

Figure 3.3: Plot of scaling factor, S, for different aerial image contrasts, M. As M approaches 1, the scaling factor drops to 0.5. Decreasing the image contrast increases the scaling factor monotonically. This scaling provides a convenient measurement for M.

3.4 Overexposure or Underexposure? LER and NILS

Based on the resist response curve, similar sized negative features could be obtained through underexposure. It is worth asking why should the pattern be overexposed, not underexposed. To answer requires consideration of pattern fidelity. Up to this point, the discussion has focused entirely on feature size, but pattern fidelity is equally important. Line edge roughness (LER) is a critical metric in evaluating pattern fidelity in any lithographic process. Not only does LER reduce pattern fidelity, but also it contributes to non-uniformity in critical dimensions. Both CD uniformity and LER can be predicted using a single image metric: normalized image log slope (NILS) [31] [19] as previously discussed in Chapter 2. Again, large NILS indicates that the aerial image of the line is sufficiently sharp for good recording. Large NILS values also imply better linewidth uniformity, lower LER, and larger process windows.

Calculating NILS near an intensity maxima yields very different values than near an intensity minima. Looking at Figure 3.1, it is not clear that line edge position is better defined near an intensity minima than near an intensity maxima. For both cases, NILS is calculated as a function of feature size and optical contrast, and plotted in Figure 3.4. At an intensity maxima, NILS decreases with decreasing feature size. At an intensity minima, NILS *increases* with decreasing feature size. Decreasing optical contrast reduces NILS in both cases.

Experimental results confirm that compared to nominal exposure, LER is significantly worse for underexposure, and significantly improved for overexposure. Examples of this are shown in Figure 3.5.

The minimum acceptable value for NILS depends on how large a process window is needed, but in general NILS greater than 2 are often sufficient [32]. From Figure 3.4, we see that when $M > 0.95$, $NILS > 2$ for all overexposure. This implies that it is desirable to have an aerial image contrast greater than 0.95 not only to achieve small feature size, but also to achieve low LER.

To emphasize the importance of NILS and why it enables STED lithography of high pat-

tern quality, consider the analogous approach using a negative tone resist and under-exposing at the peaks of the aerial image. Here one finds the NILS approaches zero for significant linewidth reduction and thus LER and pattern uniformity would be extremely poor, even if adequate dose control were possible. Writing with overexposure is key to this method for achieving features with sub-wavelength dimensions. Overexposure yields controllable and reproducible feature size, while underexposure can not.

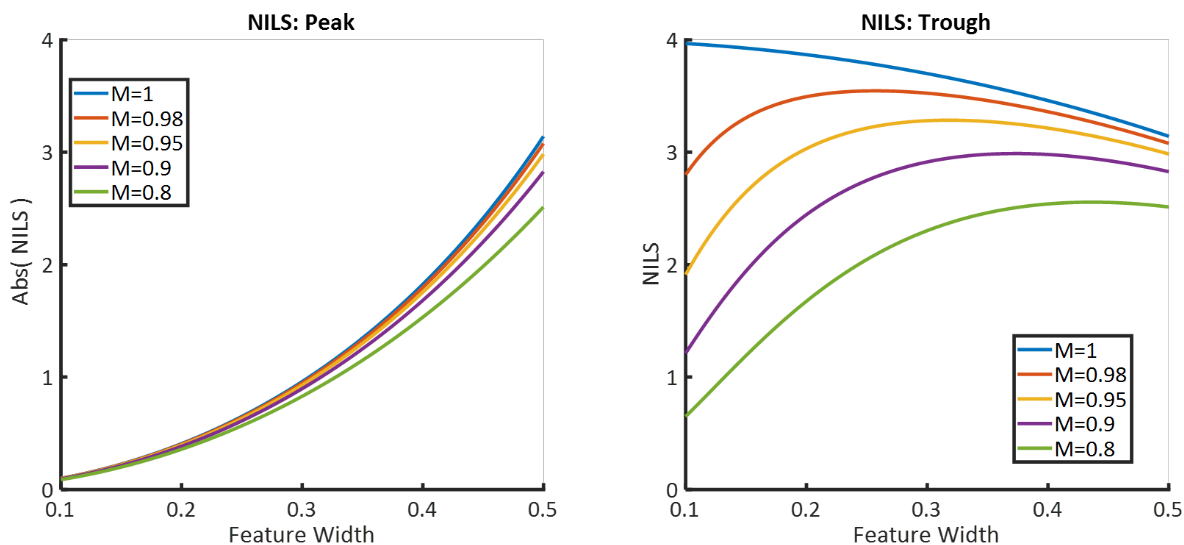

Figure 3.4: Normalized Image Log Slope (NILS) calculations near intensity maxima (peak) and minima (trough) as a function of feature width (CD) and image contrast. Around an intensity maxima, NILS decreases with decreasing CD (left plot). However, around an intensity minima, NILS increases with decreasing CD (right plot). It is for this reason that overexposure is preferred to underexposure.

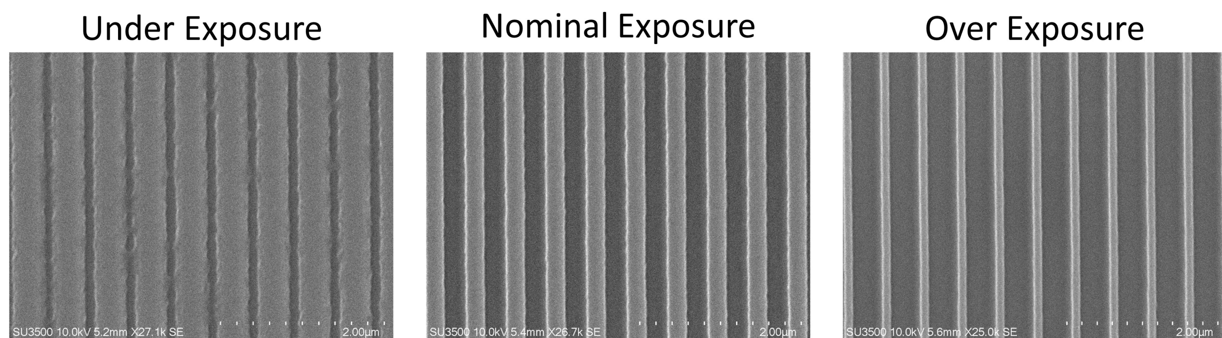

Figure 3.5: Examples of pattern fidelity for various exposure conditions. All three patterns are on a 500 nm pitch. Underexposure can yield the same size negative features as overexposure can yield positive features. The key difference is, underexposure leads to poor LER while overexposure leads to improved LER.

Chapter 4

Laser Interference Lithography

Laser interference lithography (LIL) is a maskless optical patterning process where periodic patterns are generated by optical interference. Interference between multiple coherent beams can generate a wide range of periodic patterns readily tailored via geometric alignment of interfering beams. When two or more mutually coherent beams of light coincide, the electric fields add, resulting in an intensity distribution containing alternating bright and dark regions corresponding to where the fields are in and out of phase. Typically, multiple beams are mutually coherent when all come from the same narrow linewidth source. Two beam interference yields a one dimensional line-space intensity distribution throughout the volume in which the beams overlap and for collimated beams, this intensity distribution is periodic. Spatial frequency of the interference pattern is set by the angle between interfering beams; pattern pitch is readily changed by adjusting angle of incidence. With the addition of a third, or fourth beam, two dimensional lattice interference patterns can be generated, including all two-dimensional Bravais lattice structures [38]. Throughout the overlapping volume of many interfering beams, it is also possible to achieve all 14 three-dimensional Bravais lattice structures [38].

Coherent interference patterns have found use in applications such as lithography[6] [59], holography [51] [43], structured illumination microscopy [14], and optical trapping [9]. Though this thesis focuses on application to lithography, the same analysis presented here is applicable for all of these applications.

Unlike mask-based optical lithography, LIL is capable of patterning large areas (> 1 m) within

a single exposure [55]. Also in sharp contrast to mask-based optical lithography, the depth of focus for LIL is as large as the region over which the beams overlap, increasing process latitude, and allowing for thick resist films that would otherwise be larger than the depth of focus [56]. Placing a resist coated wafer within the overlapping region of all beams records the resulting intensity pattern as a latent image in the resist. Compared to contact mask lithography, LIL does not require separate masks for each unique pattern and achieves greater resolution [56]. Compared to mask based projection lithography, LIL systems are far simpler to design, operate, maintain, and lower cost. All things considered, LIL is well suited for use both in research and in some manufacturing applications [45].

Interference patterns are used in this work for process development as well as lithographic model validation. High contrast interference patterns can be used to write sub-diffraction limited features in photoresist, as described in Chapter 3. Limitations on linewidth are set by both optical contrast and photoresist contrast [37]. To reach the smallest feature sizes, very high optical contrast, $M \approx 1$, is required. To ensure sufficient interference contrast, interferometer design and component selection, are crucial to achieve desired critical dimensions. This chapter covers the analysis used for LIL system design as well as the final system design.

4.1 Interferometer Configurations

Both single-beam (wavefront splitting) and multiple-beam (amplitude splitting) interferometer configurations have been used for LIL [56]. Of these, the most common single-beam configuration is the Lloyd's mirror interferometer [56]. Simple to construct, the Lloyd's mirror typically uses a single diverging spherical wave to illuminate a mirror and resist covered wafer. Multiple-beam interferometer designs require more components and alignment, but offer pattern flexibility not found in a single-beam interferometer. In this chapter, only multiple-beam interferometers with two beams are analyzed.

When analyzing a two-beam interferometer, the type of beam splitter impacts the sensitivity of interference contrast to beam pointing (wander) as well as polarization misalignment [21]. In-

terferometers built with two different beam splitters are analyzed. First, a plate beam splitter, as shown in Figure 4.1. Second, a grating beam splitter, as shown in Figure 4.2. Although the grating shown here is a transmission grating, a reflection grating may also be used, with similar analysis.

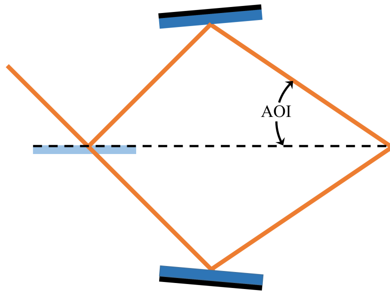

Figure 4.1: Two-beam interferometer utilizing a plate beam splitter.

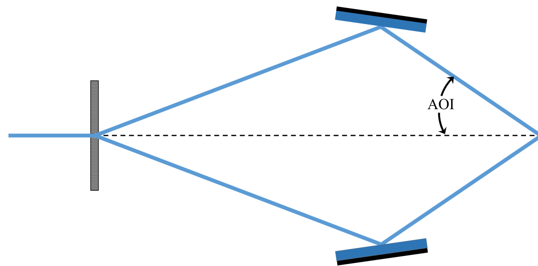

Figure 4.2: Two-beam interferometer utilizing a grating beam splitter.

4.2 Interference Contrast

Sources of contrast loss in interference patterns include: temporal coherence, beam pointing instability, polarization state, and the ratio of intensity for the interfering beams. Temporal coherence is determined by laser selection. Beam pointing stability is determined by stability of the laser, as well as the mechanical stability of the beam conditioning optics. Polarization purity and polarization angle are set by a polarizer and its angular alignment. The intensity split between beams is set by a beam splitter and, interestingly, the type of beam splitter impacts overall sensitivity to beam pointing and polarization misalignment. In this chapter, I show how selection of and specifications for these components control contrast in LIL. Much of this work is readily extended to optical imaging systems when used for two beam imaging with phase masks.

4.2.1 Contrast in Two Beam Interference Patterns

Interference contrast, sometimes called fringe visibility, is defined as

$$M = \frac{I_{max} - I_{min}}{I_{max} + I_{min}} \tag{4.1}$$

with I_{max} and I_{min} representing the intensity maxima and minima within the aerial image. Two stable, coherent beams interfering yield a fringe pattern given by

$$I = I_1 + I_2 + 2|\Gamma_{12}|\sqrt{I_1 I_2}\cos[(\vec{k}_1 - \vec{k}_2)\cdot\vec{r}] < \hat{e}_1 \cdot \hat{e}_2 > . \tag{4.2}$$

Here, I_n, $\vec{k}_n$, and $\hat{e}_n$ are the intensity, wave vector, and polarization vector corresponding to the nth beam. Brackets, $< >$, represent a time average. Finally, Γ_{12} represents the mutual coherence between the two interfering beams [46]. Substituting Equation 4.2 into Equation 3.3 yields

$$M = |\Gamma_{12}| < \hat{e}_1 \cdot \hat{e}_2 > \frac{2\sqrt{I_1 I_2}}{I_1 + I_2}. \tag{4.3}$$

Equation 4.3 states that contrast is impacted by mutual coherence between beams, polarization

states for each beam and intensity mismatch. Equation 4.2 assumes that the wave vectors are time invariant, although that is often not the case, and time varying beam pointing must also be considered. These factors are analyzed in the following sections.

4.2.2 Temporal Coherence

Interference contrast is directly proportional to the mutual coherence between interfering beams [50]. Mutual coherence, Γ_{12}, depends on the linewidth of the source, and the path length difference between the interfering beams [46]. Temporal coherence, τ, of a source with frequency bandwidth $\Delta\nu$, or equivalently center wavelength λ and spectral bandwidth $\Delta\lambda$, is given by

$$\tau = \frac{1}{\Delta\nu} = \frac{\lambda^2}{c\Delta\lambda}. \tag{4.4}$$

Multiplying the coherence time by the speed of light gives the coherence length

$$L_{coherence} = \frac{c}{\Delta\nu} = \frac{\lambda^2}{\Delta\lambda}. \tag{4.5}$$

This is the optical path length difference (OPD) over which mutual coherence is lost. Coherence length alone is not enough to determine contrast loss due to OPD. A coherence function, $\Gamma_{12}(\Delta L)$, of OPD is needed. Using the Wiener-Kinchin theorem, the coherence function can be determined given the output power spectrum of the laser [13][50]. Wiener-Kinchin relates frequency spectrum to coherence via Fourier transform [50]. Interference contrast is then given by $|\Gamma_{12}(\Delta L)|$ [50]. Assuming a laser's frequency spectra is composed of multiple lines, whose linewidth is much narrower than the free spectral range, coherence as a function of path length difference ΔL, up to the first null, is approximately a cosine function

$$\Gamma_{12}(\Delta L) \approx cos(\pi \frac{\Delta L}{L_{coherence}}). \tag{4.6}$$

For a laser with two lines, this equation is exact. The approximation is demonstrated in Figure 4.3

for a laser with 20 lines and a free spectral range of 150 MHz. The exact coherence function, plotted in blue, is calculated via Fourier transform per the Wiener-Kinchin theorem. The approximation, given by Equation 4.6, is plotted in orange. With so many laser lines, this is an extreme example which serves to illustrate the validity of the approximation.

The most obvious source of OPD is due to path length differences between arms of the interferometer. However, even if the two arms are exactly matched, the OPD will still vary over the width of the exposure field. This can be seen by considering beams of non-zero width as shown in Figure 4.4. Although both Beam 1 and Beam 2 have nominally equal path lengths on center, at the edges of the field this isn't true. At the left edge, Beam 1 has traveled a shorter distance than Beam 2. Likewise, at the right edge of the field, Beam 2 has traveled a shorter distance than Beam 1. Regardless of the care taken in aligning the arms of the interferometer, OPD will always vary across the transverse direction of the exposure field.

The analysis for path length difference in a symmetric interferometer with collimated beams, is as follows: let k_1 and k_2 be the wave numbers of each beam

$$k_1 = k_2 = k = \frac{2\pi}{\lambda}. \tag{4.7}$$

Let $\overrightarrow{K_1}$ and $\overrightarrow{K_2}$ be the wave vectors

$$\overrightarrow{K_1} = k \begin{bmatrix} sin(\theta) \\ 0 \\ cos(\theta) \end{bmatrix} \tag{4.8}$$

$$\overrightarrow{K_2} = k \begin{bmatrix} -sin(\theta) \\ 0 \\ cos(\theta) \end{bmatrix} \tag{4.9}$$

with θ representing angle of incidence, as shown in Figure 4.5. Phase delay, $\Delta\Phi$, is given by

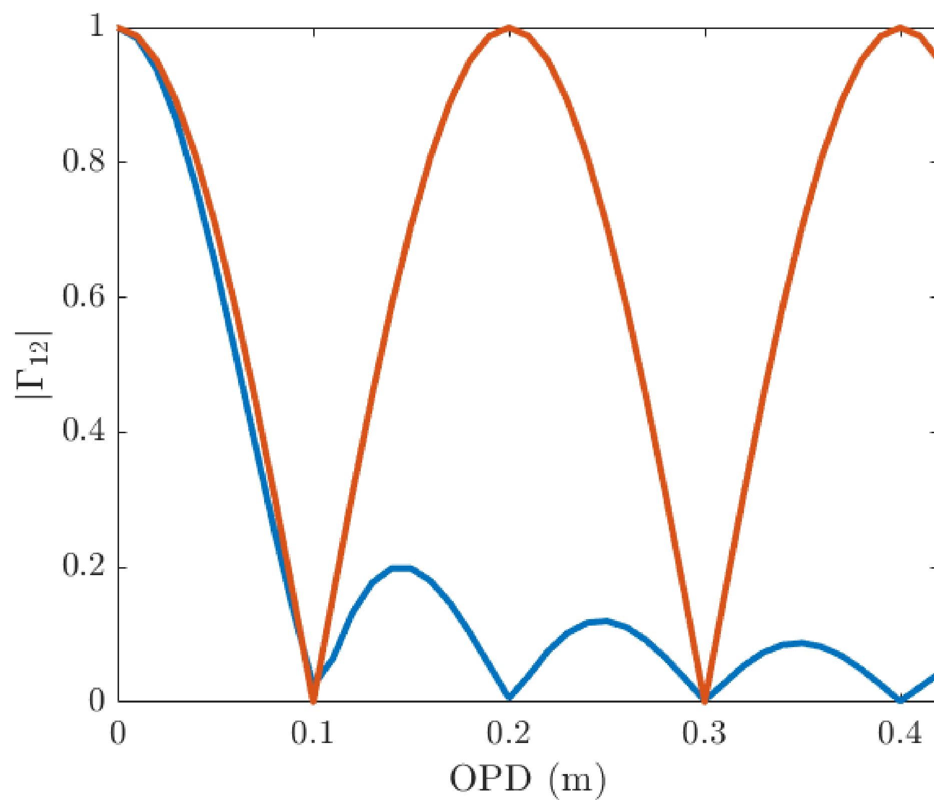

Figure 4.3: Exact and approximate coherence functions, Γ_{12}, for a laser with 20 longitudinal modes, 150 MHz free spectral range, and linewidth much narrower than the free spectral range. The exact coherence function (blue) is well approximated by a cosine (orange) up to the first null.

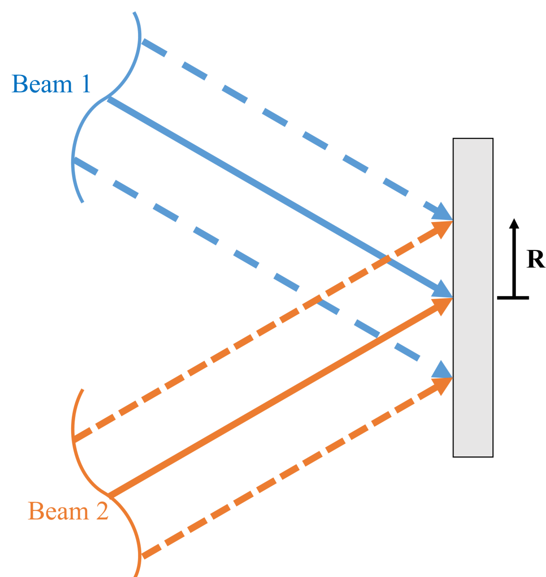

Figure 4.4: Optical path length difference (OPD) across field. Regardless of how carefully aligned each arm of the interferometer is, there will always be an OPD across the field due to the non-zero width of the interfering beams. Beam 1 (blue) and Beam 2 (orange) have equal path length on center. However, Beam 1 travels a shorter distance to the upper edge of the field, while Beam 2 travels a longer distance. The OPD can be found as a function of distance, **R**, from center.

$$\Delta\Phi = \Phi_1 - \Phi_2 = (\vec{K_1} \cdot \vec{r} + \phi_1) - (\vec{K_2} \cdot \vec{r} + \phi_2). \tag{4.10}$$

The term $\Delta\phi$ accounts for any OPD on center, such as that due to a plate beam splitter. Moving away from center, in the plane of incidence

$$\vec{r} = R\hat{x} \tag{4.11}$$

introduces a phase delay

$$\Delta\Phi = 2kRsin(\theta) + \Delta\phi \tag{4.12}$$

proportional to field size R (Figure 4.4). Along the direction normal to the plane of incidence, $\hat{y}$, phase delay does not change. A phase delay of $\Delta\Phi$ corresponds to an OPL ΔL of

$$\Delta L = \frac{\Delta\Phi}{k} = 2Rsin(\theta) + \frac{\Delta\phi}{k}. \tag{4.13}$$

Using Equations 4.13 and 4.6, the contrast for distances less than R_{beat} is approximately

$$M \approx 1 - \frac{2\pi R^2 \left(ksin(\theta)\right)^2}{(kL_{coherence} - \Delta\phi)^2}. \tag{4.14}$$

Limits on maximum exposure field width come from requirements for minimum acceptable contrast, temporal coherence of the source, and angle of incidence. A parameterized plot of Equation 4.14 is given in Figure 4.6 showing that at fixed contrast, field size relative to coherence length decreases with increasing angle of incidence. For periods near the diffraction limit, the field size is a small fraction of the laser coherence length - with a steep penalty for high contrast.

Laser interference lithography typically relies on near-UV or deep-UV lasers. Near-UV lasers, such as argon-ion lasers, may operate with either multi-longitudinal modes or single frequency. These lasers have coherence lengths ranging from several cm (multi-longitudinal) to over 100 m

(single frequency). Deep-UV excimer lasers may have a coherence length as short as 100 μm (free-running), or as long as several cm (line narrowed).

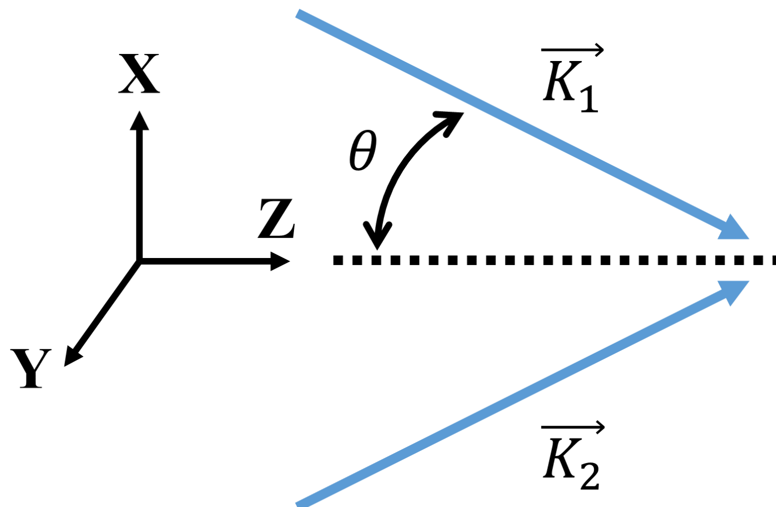

Figure 4.5: Wave vectors and coordinate system for two beam interference. $\vec{K}_1$ and $\vec{K}_2$ are the wave vectors for the first and second beams respectively. Both lie in the XZ plane, are symmetric about the z-axis, and have angle of incidence θ.

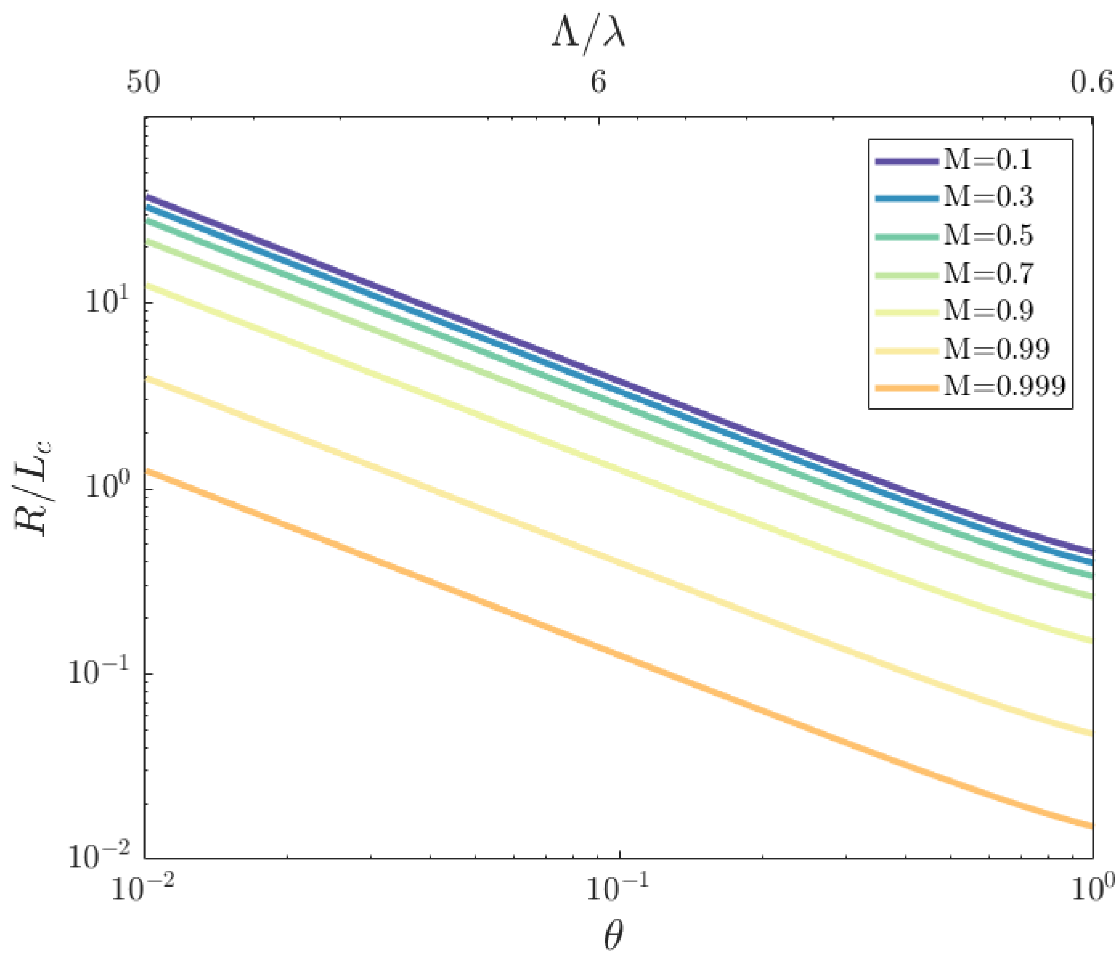

Figure 4.6: Parameterized plot of Equation 4.14 with $\Delta\phi = 0$. Minimum contrast is plotted as a function of angle of incidence and field radius normalized by coherence length. As angle of incidence increases, field size relative to source coherence length must decrease to maintain constant contrast.

4.2.3 Beam Pointing Instability

In any real interferometer, beam pointing varies slightly over time due to temperature fluctuations, mechanical vibrations, or may be inherent to the laser cavity itself [41]. Beam wander leads to a shift in the position of interference fringes. Photoresists, light sensitive media, and detectors, accumulate dose during exposure. Therefore, shifting fringe patterns will wash out the contrast. In considering beam pointing stability, there is a distinct difference between an interferometer built with a plate beam splitter and an interferometer built with a grating beam splitter. Using a plate beam splitter, shown in Figure 4.7, introducing a tilt to the incident beam in the plane of incidence causes the two interfering beams to walk off in opposite directions and the angle of incidence increases for both beams. In contrast, when using a grating beam splitter, shown in Figure 4.8, introducing a tilt to the incident beam causes the two interfering beams to walk off in the same direction. The total beam displacement will depend on the physical dimensions of the interferometer. The analysis here assumes that the width of the interfering beams is large compared to the total lateral displacement.

Analysis of beam pointing for a plate beam splitter configuration as follows: let $\overrightarrow{K_i}$ be the ith wave vector and $\overrightarrow{K_i'}$ be the ith perturbed wave vector due to beam pointing angle δ, shown in Figure 4.9A. Then, $\overrightarrow{K_1'}$ and $\overrightarrow{K_2'}$ are, to first order,

$$\overrightarrow{K_1'} \approx k \begin{bmatrix} sin(\theta+\delta) \\ 0 \\ cos(\theta+\delta) \end{bmatrix}, \tag{4.15}$$

$$\overrightarrow{K_2'} \approx k \begin{bmatrix} -sin(\theta+\delta) \\ 0 \\ cos(\theta+\delta) \end{bmatrix}. \tag{4.16}$$

where θ represents the angle of incidence and k the wave number. Total path delay, $\Delta\Phi$, is

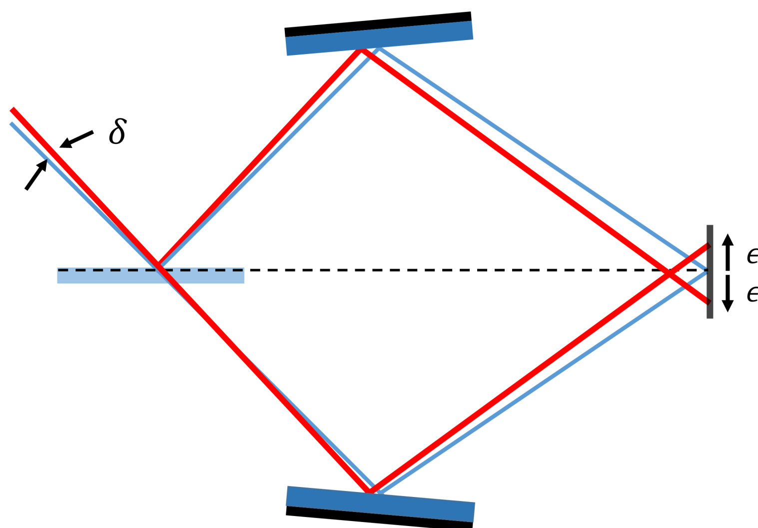

Figure 4.7: Beam path deviations due to beam pointing instability in an interferometer built with a plate beam splitter. Blue beams represent nominal beam paths, red beams represent perturbed beam paths resulting from beam pointing error δ. Each beam walks off in opposing directions and both beams have an equal change in path length. At the edges of the exposure field, the beams do not remain overlapped and there is no interference.

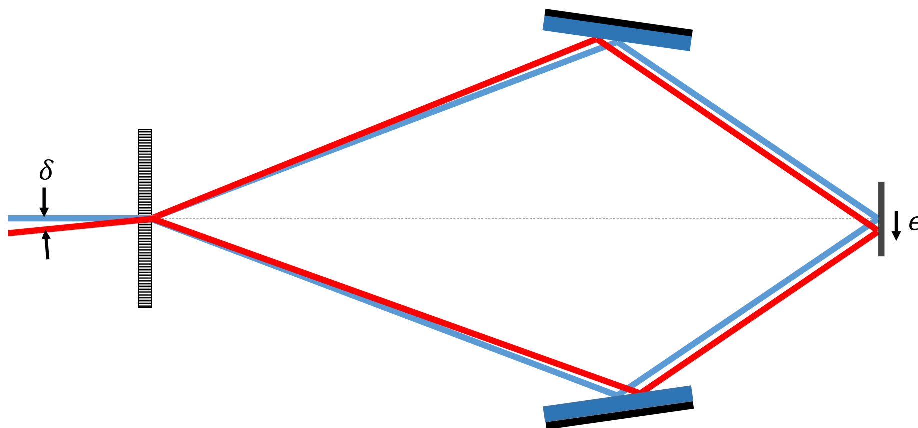

Figure 4.8: Beam path deviations due to beam pointing instability in an interferometer built with a grating beam splitter. Blue beams represent nominal beam paths while red beams represent perturbed beam paths resulting from beam pointing error δ. Both beams walk off in the same direction in the plane of interference. One beam gains path length while the other beam loses path length. At the edges of the exposure field, both beams remain overlapped.

$$\Delta\Phi = \Phi_1 - \Phi_2 = (\overrightarrow{K_1'} \cdot \overrightarrow{r} + \phi_1) - (\overrightarrow{K_2'} \cdot \overrightarrow{r} + \phi_2). \tag{4.17}$$

At position $\vec{r} = R\hat{x}$, $\overrightarrow{K_i'} \cdot \overrightarrow{r}$ can be expanded as

$$\overrightarrow{K_1'} \cdot \vec{r} = kRsin(\theta + \delta) \approx kR\left(sin(\theta) + \delta cos(\theta) - \frac{\delta^2}{2}sin(\theta) + \ldots\right) \tag{4.18}$$

and total phase delay is, approximately,

$$\Delta\Phi \approx 2kR\left(sin(\theta) + \delta cos(\theta) - \delta^2 sin(\theta)\right). \tag{4.19}$$

To first order, the fringe phase shifts by a factor of $2\delta kRcos(\theta)$. This shift is largest at the edge of the exposure field, but reduced by increasing angle of incidence.

Analysis of beam pointing for a grating beam splitter configuration is as follows: A perturbation in pointing angle increases the angle of incidence for one beam, while decreasing the angle for the other, as shown in Figure 4.9B. Perturbed wave vectors $\overrightarrow{K_1'}$ and $\overrightarrow{K_2'}$ are, to first order,

$$\overrightarrow{K_1'} \approx k\begin{bmatrix} sin(\theta+\delta) \\ 0 \\ cos(\theta+\delta) \end{bmatrix}, \tag{4.20}$$

$$\overrightarrow{K_2'} \approx k\begin{bmatrix} -sin(\theta-\delta) \\ 0 \\ cos(\theta-\delta) \end{bmatrix}. \tag{4.21}$$

Total path delay, $\Delta\Phi$, is again given by Equation 4.17. At position $\vec{r} = R\hat{x}$, $\overrightarrow{K_i'} \cdot \overrightarrow{r}$ can be expanded as

$$\overrightarrow{K_1'} \cdot \vec{r} = kRsin(\theta + \delta) \approx kR\left(sin(\theta) + \delta cos(\theta) - \frac{\delta^2}{2}sin(\theta) + \ldots\right) \tag{4.22}$$

and

$$\overrightarrow{K_2'} \cdot \vec{r} = -kRsin(\theta - \delta) \approx -kR\left(sin(\theta) - \delta cos(\theta) - \frac{\delta^2}{2} sin(\theta) + \ldots\right) \tag{4.23}$$

so that total phase delay is, approximately,

$$\Delta\Phi \approx 2kR\left(1 - \frac{\delta^2}{2}\right) sin(\theta). \tag{4.24}$$

To first order, there is no fringe phase shift with δ. This is in sharp contrast with the case of a plate beam splitter, Equation 4.19. A shift does occur to second order in a direction opposite that of the beam motion. As a result, a two-beam interferometer utilizing a grating beam splitter is far less sensitive to beam pointing instability. In both cases, fringe phase shift depends only on angle of incidence and pointing error, not the overall dimensions of the interferometer. The nominal fringe phase, however, does depend on the physical dimensions for each leg of the interferometer.

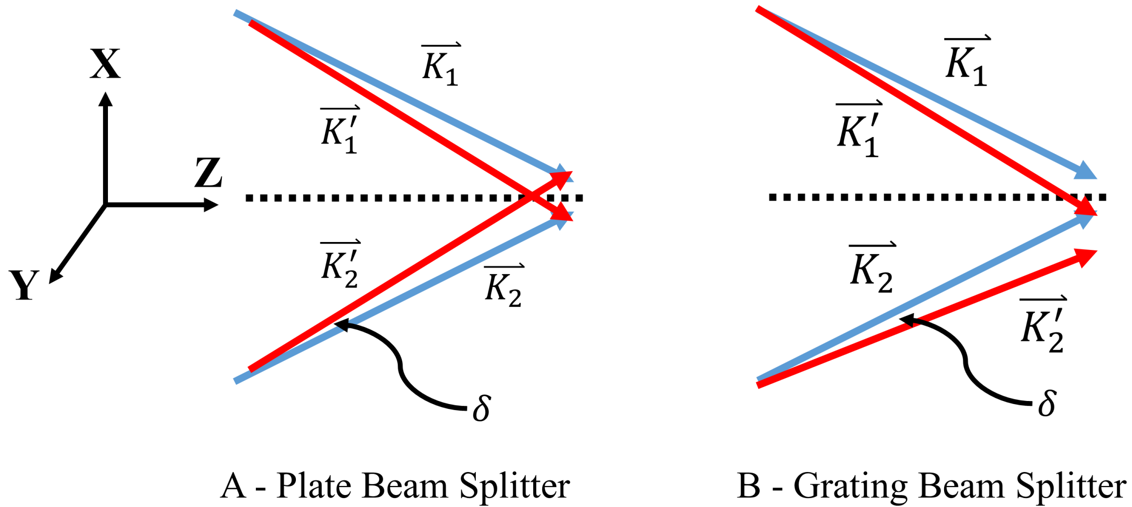

Figure 4.9: Wave vectors with beam pointing errors for interferometers built with: A) a plate beam splitter and B) a grating beam splitter. Nominal wave vectors $\vec{K_1}$ and $\vec{K_2}$ are displaced by the beam pointing error δ resulting in $\vec{K_1'}$ and $\vec{K_2'}$. For a plate beam splitter, the angle of incidence for both wave vectors increases by δ. For a grating beam splitter, the angle of incidence increases for one beam, while decreasing for the other.

Contrast loss due to beam pointing depends on the time-dependence of the instability. To illustrate the impact, I assume a single linear sweep of the beam from zero angle to an angle δ over the duration of recording. Intensity distributions are integrated over each angular position of the beams. Resulting contrast for each beam splitter type is plotted with δ= 5 μrad (Figure 4.10) and δ= 5 mrad (Figure 4.11). For a field size of $10^4\lambda$ and $\delta = 5$ μrad, contrast for a plate beam splitter configuration contrast drops to 0.95, while contrast for a grating configuration has no significant decrease. For the same field size and δ = 5mrad, contrast for a plate beam splitter configuration drops to zero, while contrast for a grating beam splitter configuration drops to 0.94. These examples illustrate that a plate beam splitter configuration is about three orders of magnitude more sensitive to beam pointing than is the grating beam splitter. For angular deviations greater than 1 mrad, the plate beam splitter contrast drops to zero for fields more than a hundred wavelengths across.

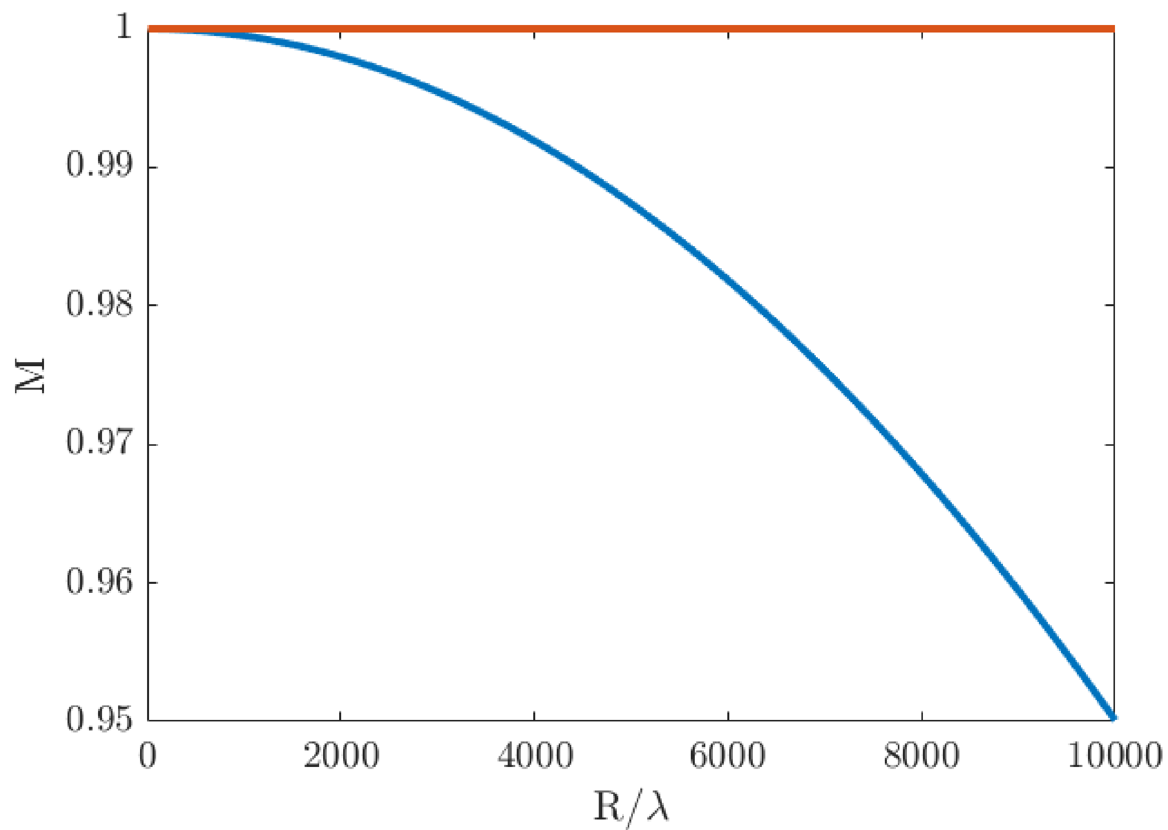

Figure 4.10: Contrast loss across field due to beam wander. A plate beam splitter (blue) is compared with a grating beam splitter (orange). Both have 0.5 rad angle of incidence and 5 μrad total deviation. Contrast is determined by integrating over the intensity at each beam angle. Contrast loss in an interferometer built with a plate beam splitter is roughly equivalent to that of a grating configuration with 10^3 greater beam pointing errors.

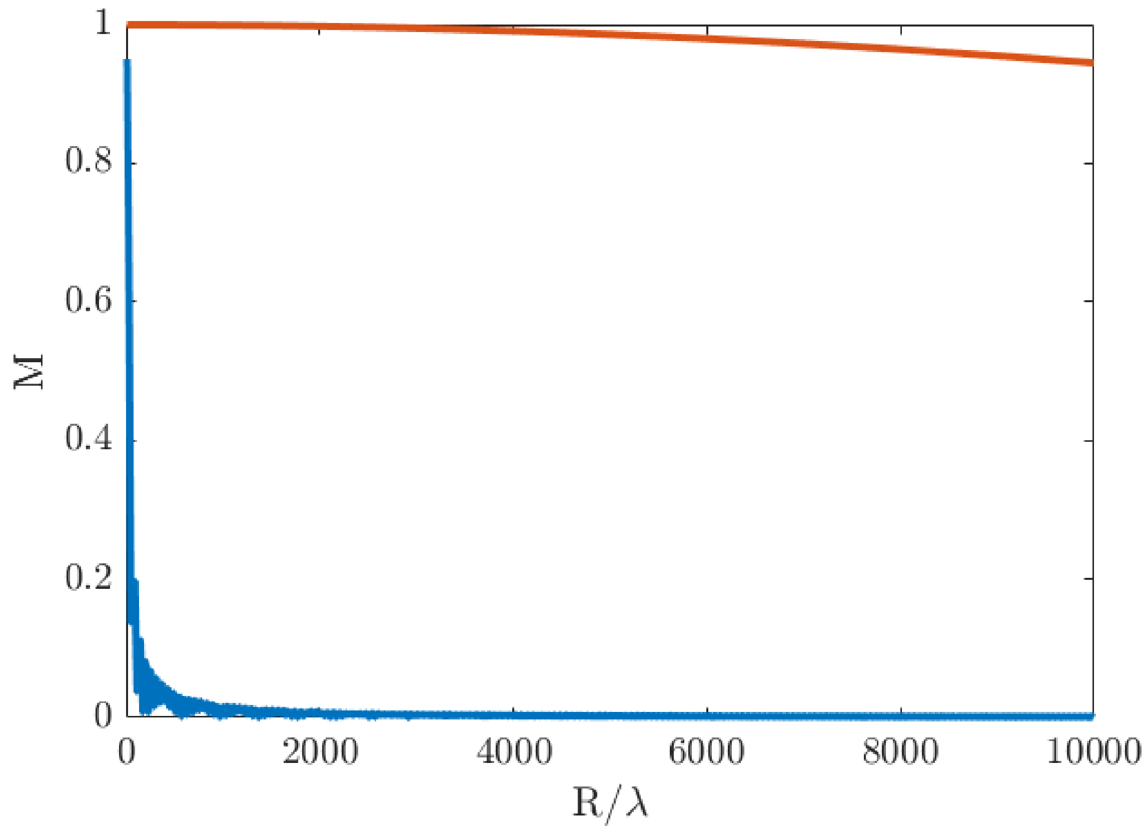

Figure 4.11: Contrast loss across field due to beam wander. A plate beam splitter (blue) is compared with a grating beam splitter (orange). Both have 0.5 rad angle of incidence and 5 mrad total deviation. Contrast is determined by integrating over the intensity at each beam angle. Grating contrast drops to about 0.94 while plate contrast drops to zero. An interferometer built with a grating loses far less contrast with beam wander than does an interferometer built with a plate.

4.2.4 Polarization

From Equation 4.3, contrast is reduced in linear proportion to the time averaged polarization dot product $< \hat{e}_1 \cdot \hat{e}_2 >$. With partially polarized linear polarization states, this term can be further reduced to

$$< \hat{e}_1 \cdot \hat{e}_2 >= DOP * cos(\Theta) \tag{4.25}$$

where DOP is the degree of polarization, and Θ is the angle between interfering linear polarization vectors. From Equation 4.25, contrast is maximized for s-polarized light, decreases linearly with degree of polarization (DOP) and, for small polarization misalignment, interference contrast decreases quadratically with angle Θ. While any misalignment from s-polarization reduces interference contrast, the magnitude, again, depends on the type of beam splitter.

In plate beam splitter configurations, each arm of the interferometer will have a mirrored polarization state. Rotating the incident polarization by angle ϕ, the polarization in each arm will rotate in opposite directions as shown in Figure 4.12A. Polarization, in the reference frame perpendicular to each beam, is given by the Jones vectors

$$\overrightarrow{J_1} = \begin{bmatrix} sin(\phi) \\ cos(\phi) \end{bmatrix}, \tag{4.26}$$

$$\overrightarrow{J_2} = \begin{bmatrix} -sin(\phi) \\ cos(\phi) \end{bmatrix}. \tag{4.27}$$

This polarization in the laboratory reference frame, $\vec{P}$, is given by

$$\vec{P} = \begin{bmatrix} cos(\theta) & 0 \\ 0 & 1 \\ sin(\theta) & 0 \end{bmatrix} \vec{J}. \tag{4.28}$$

From Equation 4.28, the polarization vectors in the lab reference frame are

$$\vec{P_1} = \begin{bmatrix} sin(\phi)cos(\theta) \\ cos(\phi) \\ cos(\phi)sin(\theta) \end{bmatrix}, \tag{4.29}$$

$$\vec{P_2} = \begin{bmatrix} -sin(\phi)cos(\theta) \\ cos(\phi) \\ -cos(\phi)sin(\theta) \end{bmatrix}. \tag{4.30}$$

Interference contrast scales as

$$M \propto \vec{P_1} \cdot \vec{P_2} = -sin^2(\phi)cos^2(\theta) + cos^2(\phi) - sin^2(\phi)sin^2(\theta) \tag{4.31}$$

and simplifies to

$$M \propto \cos^2(\phi) - \sin^2(\phi). \tag{4.32}$$

Interestingly, this contrast loss is independent of angle of incidence.

In grating beam splitter configurations, each arm of the interferometer will have the same polarization state. Rotating the incident polarization by angle ϕ, the polarizations in each arm of the interferometer rotate in the same direction as indicated in Figure 4.12B. Polarization, in the reference frame perpendicular to each beam, is given by the Jones vectors

$$\overrightarrow{J_1} = \overrightarrow{J_2} = \begin{bmatrix} sin(\phi) \\ cos(\phi) \end{bmatrix}. \tag{4.33}$$

This polarization in the laboratory reference frame is, again, computed by Equation 4.28, and the polarization vectors in the lab reference frame are

$$\vec{P_1} = \begin{bmatrix} sin(\phi)cos(\theta) \\ cos(\phi) \\ cos(\phi)sin(\theta) \end{bmatrix}, \tag{4.34}$$

$$\vec{P_2} = \begin{bmatrix} sin(\phi)cos(\theta) \\ cos(\phi) \\ -cos(\phi)sin(\theta) \end{bmatrix}. \tag{4.35}$$

In this case, the contrast reduction scales as

$$M \propto \vec{P_1} \cdot \vec{P_2} = \sin^2(\phi)\left(\cos^2(\theta) - \sin^2(\theta)\right) + \cos^2(\phi). \tag{4.36}$$

Contrast loss in a grating beam splitter configuration is sensitive not only to the polarization misalignment, but also to the angle of incidence; a significant difference from the plate beam splitter. Additionally, for $\theta < \pi/2$, the grating interferometer loses less contrast than the plate configuration and, as the angle of incidence approaches $\pi/2$ radians, Equation 4.36 converges to Equation 4.32; both interferometer configurations become equivalent. Equations 4.32 and 4.36 are plotted for comparison in Figure 4.13. For a given angle of incidence, contrast loss in an interferometer utilizing a grating beam splitter is less severe to polarization misalignment.

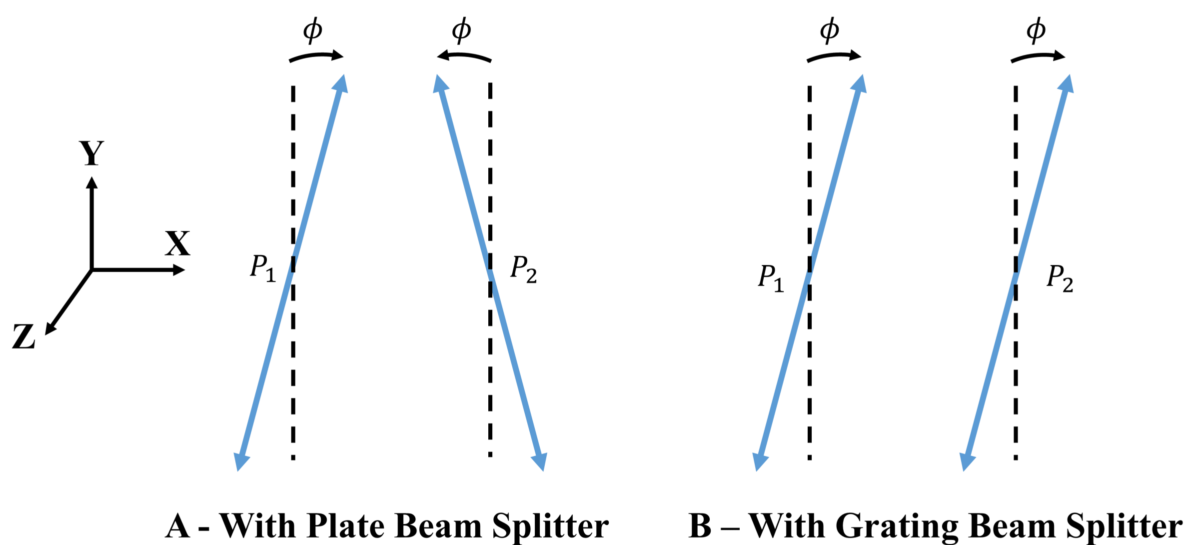

Figure 4.12: Polarization vectors after beam splitters with rotated linear incident polarization. A) With a plate beam splitter, the polarizations are mirrored, rotating in opposite directions in each arm. B) With a grating beam splitter, the polarizations rotate in the same direction in both arms.

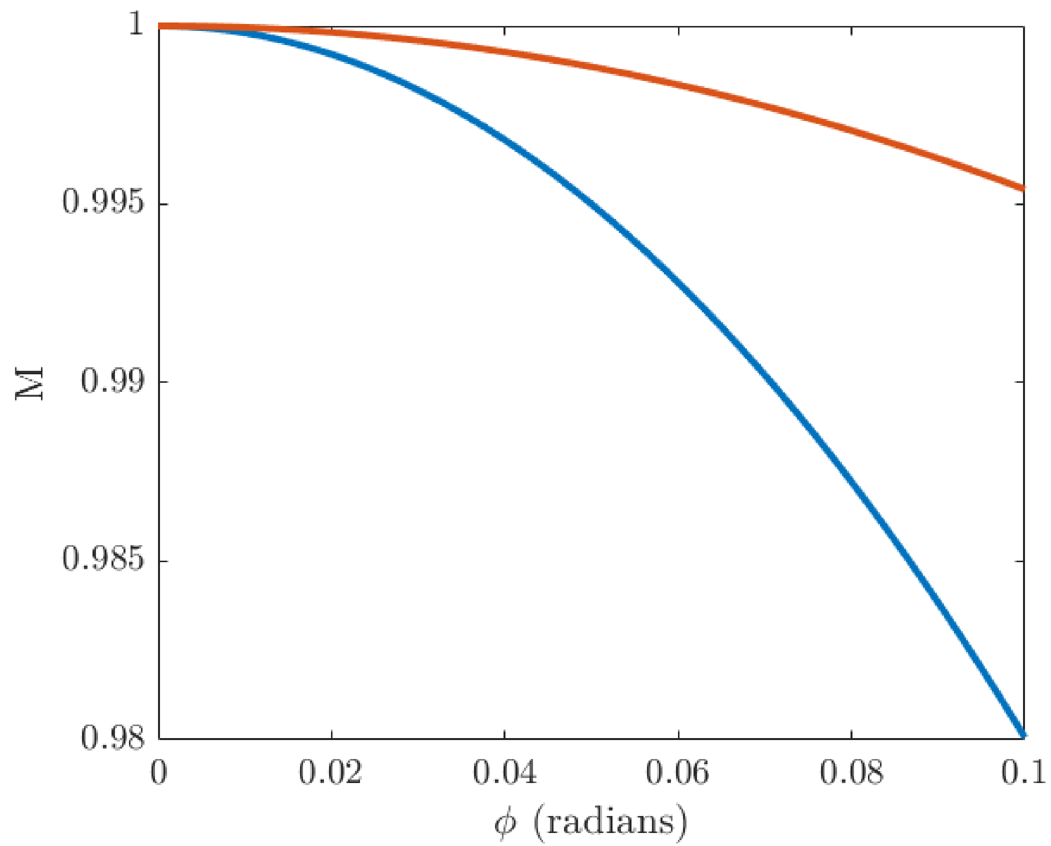

Figure 4.13: Contrast loss due to polarization misalignment. Angle of incidence $\theta = 0.5$ rad. Interferometers built with a grating beam splitter (orange) are less sensitive to polarization angle misalignment than interferometers built with a plate beam splitter (blue).

4.2.5 Intensity Difference

An intensity mismatch may be caused by either of two factors. The first is a mismatched intensity due to the beam splitter, leading to a constant intensity ratio across the plane of interference. When using plate beam splitters, the intensity split is close to 50:50. However, when using gratings as beam splitters this is often not the case as many commercially available gratings are blazed. As a result, one diffraction order will have greater intensity than the other. The second factor is a lateral misalignment of the interfering beams. In this case, the intensity mismatch will vary across the plane of interference.

The final term in Equation 4.3 shows whenever $I_1 \neq I_2$, contrast is reduced. For this reason, intensity difference is an important consideration both for beam splitter selection, as well as beam alignment. From Equation 4.3, contrast reduction due only to the intensity ratio $\frac{I_1}{I_2}$ is

$$M \propto \frac{2\sqrt{\frac{I_1}{I_2}}}{\frac{I_1}{I_2}+1} \tag{4.37}$$

and is plotted in Figure 4.14. This shows that contrast loss due to small intensity mismatch has sub-linear scaling with respect to I_1/I_2. As a result, even for a large mismatch in intensity, the contrast loss is modest. For example, when $I_1/I_2 = 1/2$, contrast is only reduced to 0.94. For many lithographic patterns, modest mismatch of beam intensity will not be a concern. Should intensity mismatch due to the beam splitter reduce contrast below acceptable values, a neutral density filter may be used on one arm to balance the intensity between the beams.

4.2.6 Multi-Factor Analysis

As each contrast loss mechanism given above contributes equally to final contrast, design trade-offs are often necessary. Considering two factors at a time, cross sections through the design space are given for DOP and ϕ , ϕ and I_1/I_2, as well as δ and L_c. Figures 4.15 through 4.17 plot contrast as a function of two contributing factors. These plots can be used to make trade-off decisions for system design and tolerancing.

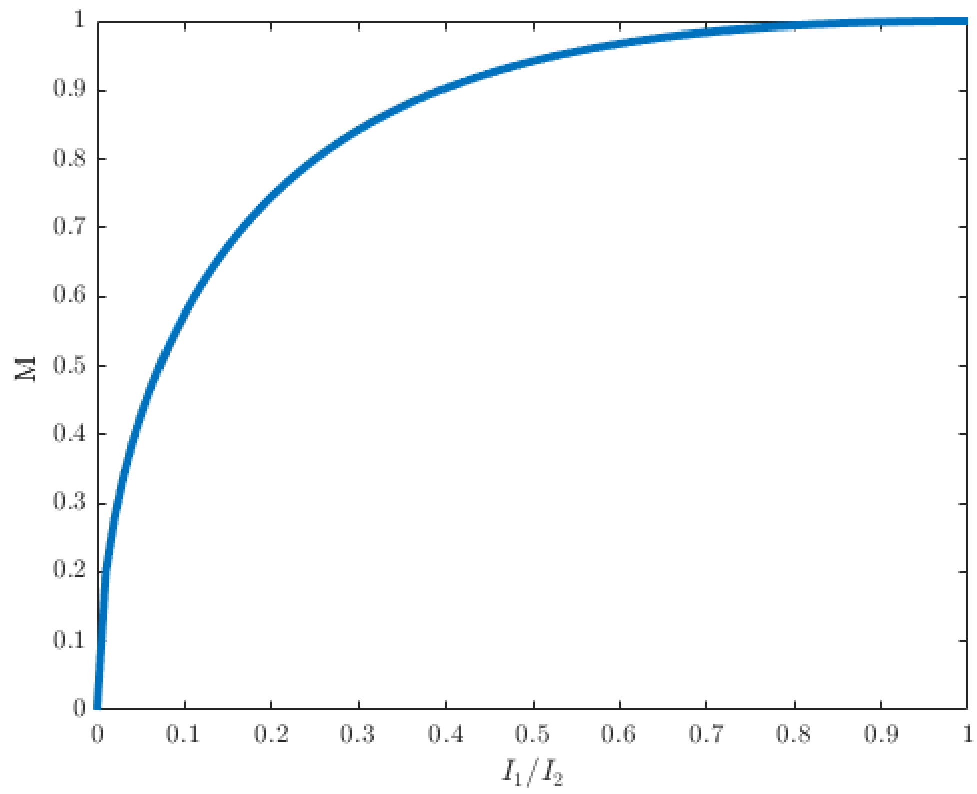

Figure 4.14: Contrast loss due to intensity mismatch between interfering beams.

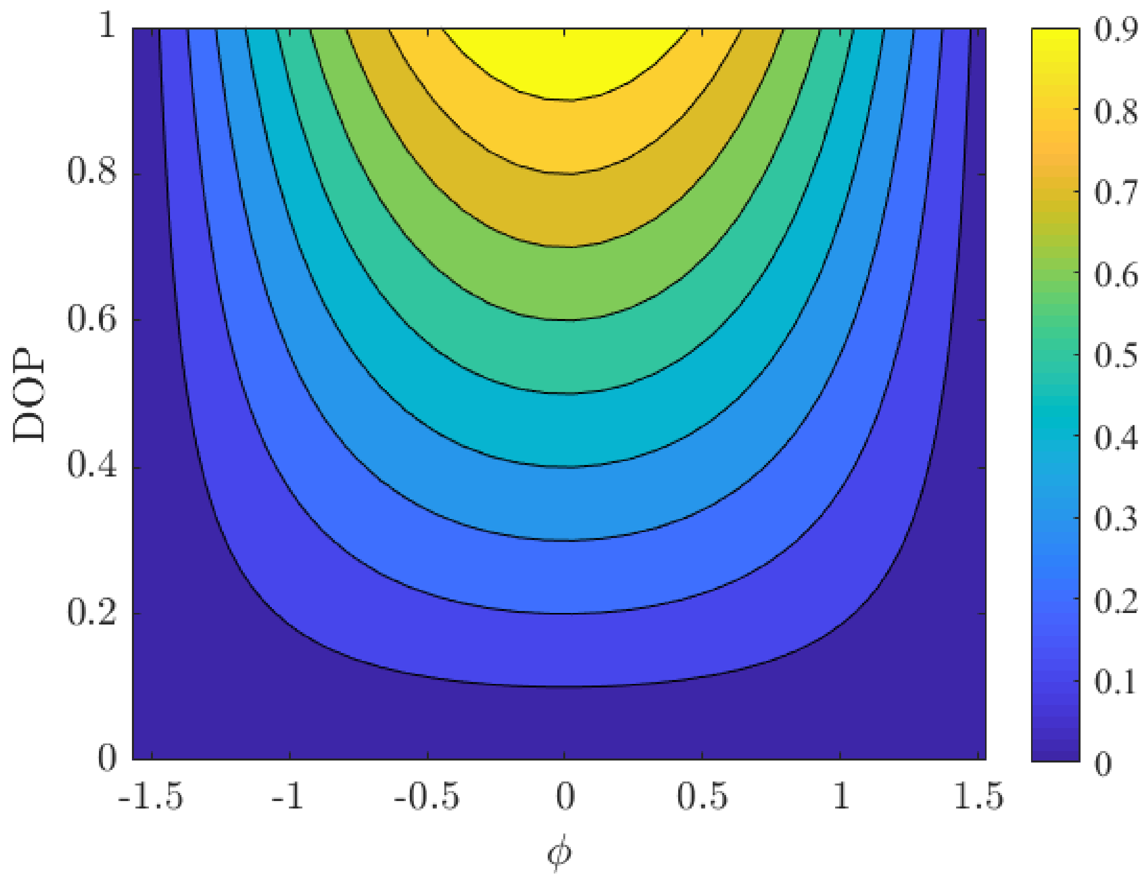

Figure 4.15: Contrast as a function of polarization misalignment, ϕ, and degree of polarization.

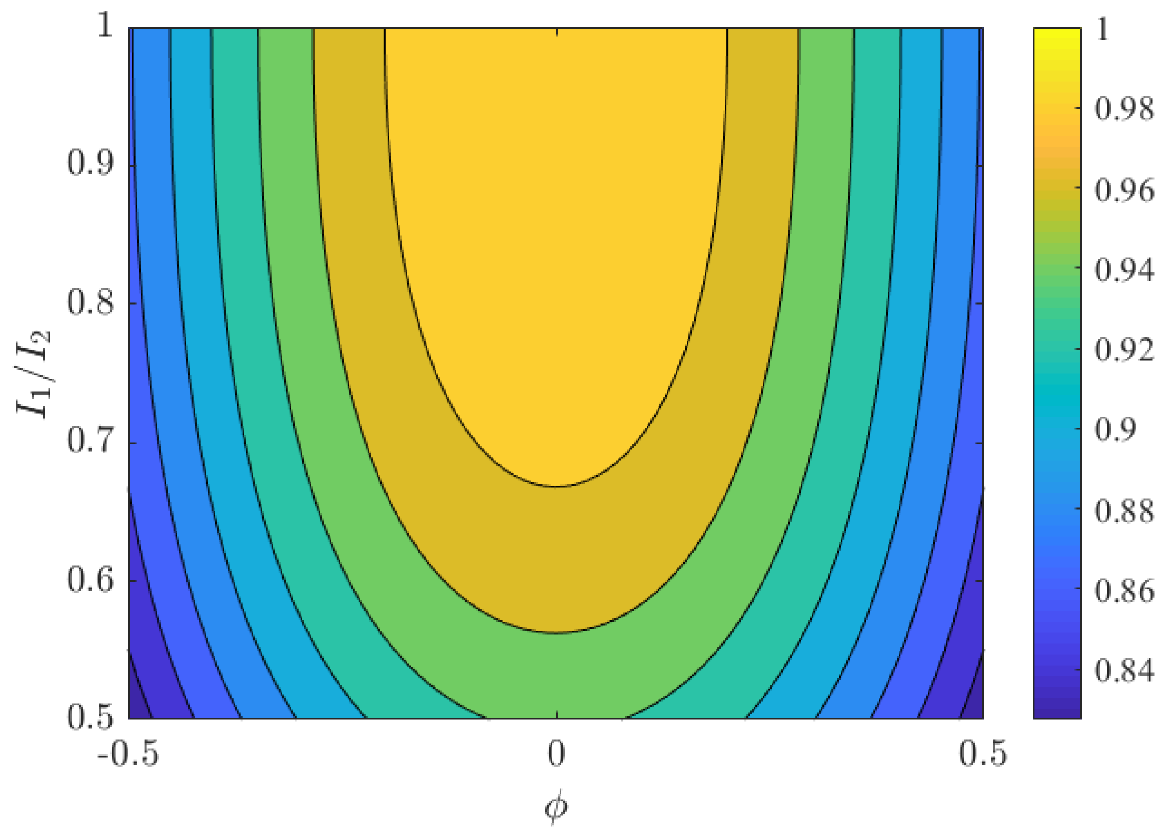

Figure 4.16: Contrast as a function of polarization misalignment, ϕ, and intensity imbalance I_1/I_2.

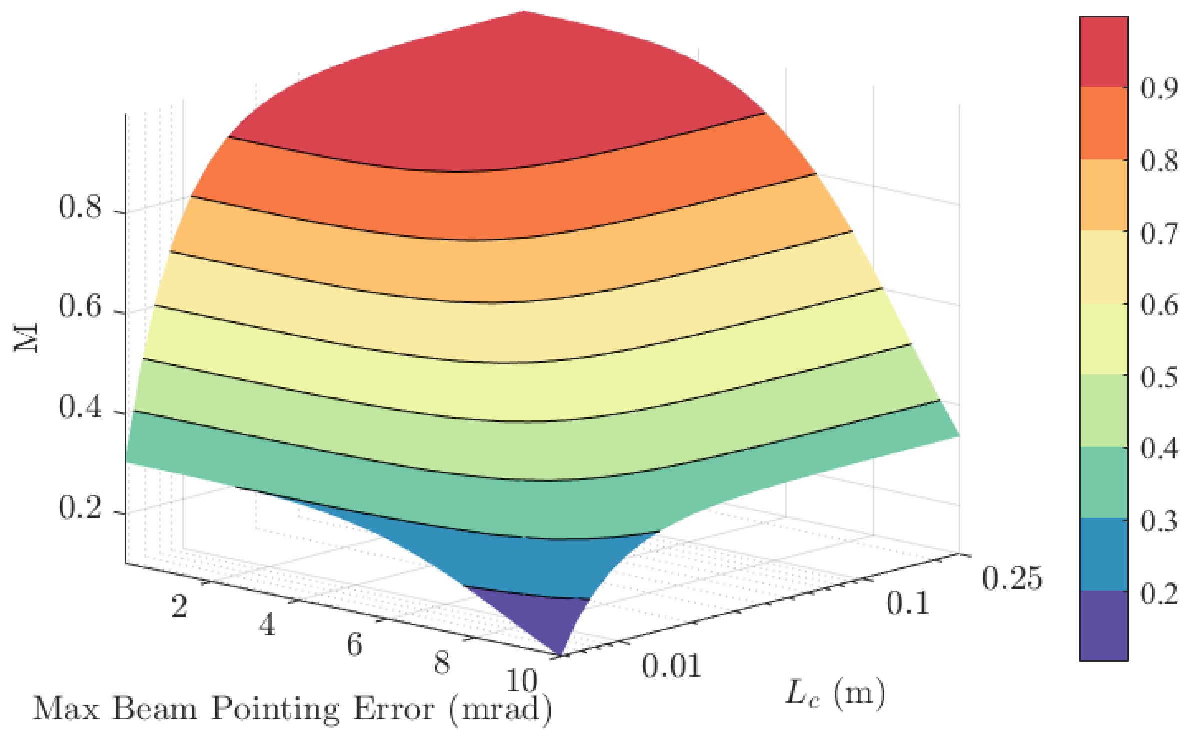

Figure 4.17: Contrast as a function of total beam wander, δ, and coherence length, L_c. Calculated for fixed field size of $R = 4$ mm and fixed $\theta = 0.5$.

4.2.7 Component Selection and Implementation

After the field size, angle of incidence, and minimum contrast are set, other components can be selected. Polarizer selection is predominantly set by extinction ratio. The extinction ratio determines the degree of polarization, and must be sufficiently large. Cube polarizers are easily aligned to ensure s-polarization. Laser selection is driven by several key factors. First, the wavelength needs to be appropriate for the recording medium. Second, the coherence length needs to be sufficiently long given the desired field size. Third, the power of the laser needs to be sufficiently high to expose the light sensitive media.

When selecting a grating as a beam splitter for a two beam interferometer, the spatial frequency should be large enough that only the ± 1 orders propagate. This condition is met when grating pitch is less than twice wavelength. Ideally, the grating is not blazed so that the optical power is evenly split. However, it is my experience that the vast majority of high frequency gratings are blazed. In this case, a neutral density filter can be incorporated into one arm of the interferometer to balance intensities. For transmission gratings, a zero stop is often needed.

4.2.8 Summary of Contrast Analysis

Interference contrast in a two beam interferometer depends on angle of incidence, field size, source coherence, beam pointing stability, polarization state, intensity balance, and type of beam splitter. Coherence length of the source determines maximum field size for a given contrast and angle of incidence. Beam pointing instability leads to shifting interference fringes during recording, washing out contrast. Contrast is maximized for s-polarized light and contrast drops in proportion to degree of polarization. Misalignment from s-polarization also reduces contrast. Intensity balance between the arms of the interferometer has a small impact even for modest mismatch. Using a grating as a beam splitter in a two-beam interferometer gives two distinct advantages. The sensitivity to beam pointing is much lower and sensitivity to polarization misalignment is also lower compared to a plate beam splitter.

4.3 Intensity Distribution

Single transverse mode laser cavities produce beams with Gaussian intensity profiles. Using Gaussian beams for photolithography is problematic because different doses are delivered across the exposure field. As a result, linewidth varies dramatically within each exposure. This linewidth variation can be easily calculated, given interference contrast, as shown in Chapter 3. When interference contrast is perfect, $M = 1$, linewidth dependence on dose is

$$w \propto \frac{1}{\sqrt{dose}} \tag{4.38}$$

as given in Chapter 3. Across a Gaussian beam, the intensity from the center to the waist radius drops by e^2, increasing recorded linewidth by a factor of e. Even at a radius equal to 1/3 the of beam waist, the intensity drops by 20% and the linewidth increases by $\approx$ 12%. Large spatial variations in linewidth are problematic for many applications; typically lithographers look for <10% variation. One method to deal with this would be to aperture the beam [56], but ringing due to diffraction and the loss of otherwise useful photons is less than ideal. A better solution is to use a beam with a top-hat intensity profile. This can be achieved by adding a beam shaper to the optical train. Refractive beam shapers produce top-hat beams with low variation in intensity and nearly flat phase. Beam shapers are most conveniently inserted before the beam splitter.

4.4 Final LIL System Design and Implementation

The final system design is detailed in the schematic in Figure 4.18. The light source is an Ar-Ion laser (λ = 363.8 nm) with feedback loop to stabilize output power. This laser lacks an intracavity etalon, so coherence length depends on output power. For an output of 6 mW, the measured coherence length is >1.5 m. Output polarization is cleaned up using a polarizing beam splitter with an extinction ratio of 2000:1. The beam profile is cleaned with a spatial filter. After spatial filtering the beam is collimated to a diameter of 3.9 mm. A Shack-Hartman wavefront sensor is used to check the collimated beam. After tuning the optical alignment, the filtered beam

has a Strehl Ratio greater than 0.99. An optional refractive beam shaper assembly can be inserted after the spatial filter and before the beam splitter. This refractive beam shaper is sold by Eksma Optics (GTH-4-2.2FA) with a custom UV AR coating. After the beam shaper, a lens with 2 m focal length is placed 2 m from the wafer plane as measured along the optical path. The beam splitter is a holographic grating with spatial frequency of 1800 lines/mm. All mirrors are dielectric with high reflectivity in the near UV. Mirrors in each leg of the interferometer are mounted along optical rails to allow manual adjustment of the angle of incidence. Holding laser power constant to maintain high temporal coherence, optical dose is changed by changing the exposure time, controlled by a mechanical shutter.

Wafer positioning is achieved via high precision linear stages (PM500) in an XZ configuration. These stages are capable of highly stable positioning once a target position is reached. Wafers are held by vacuum chuck mounted on the stage assembly. A small vacuum pump is connected to the chuck via tygon tubing.

This LIL tool is controlled by a Matlab script, given in Appendix B.

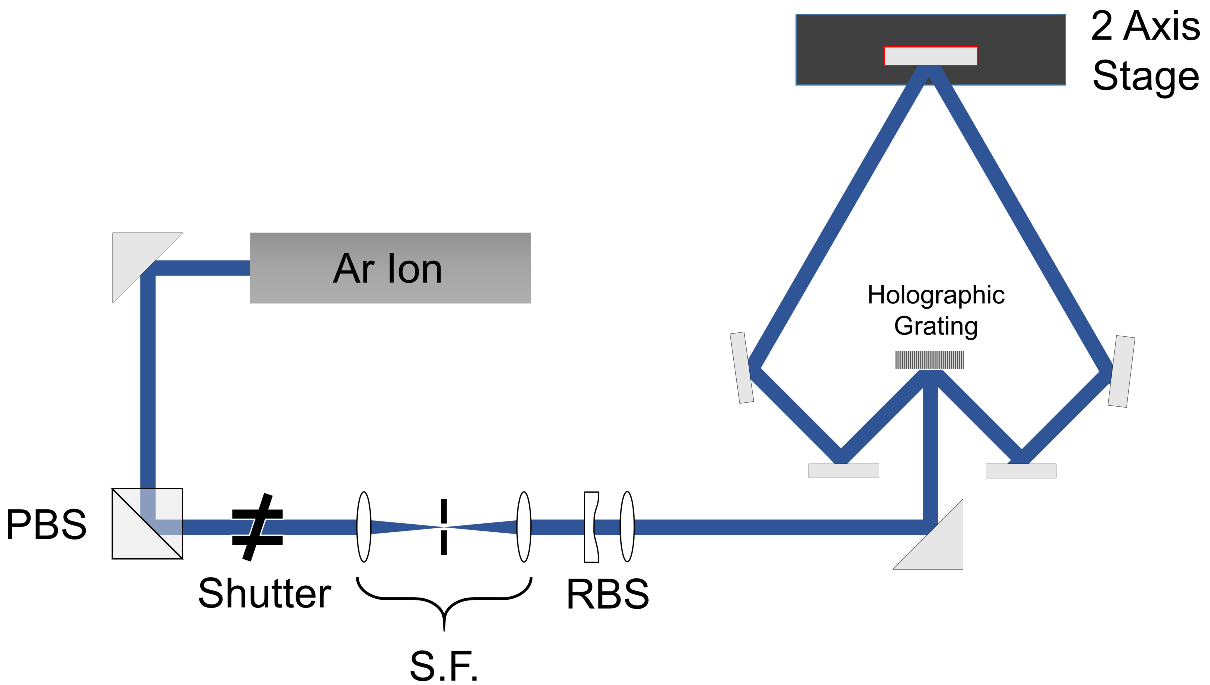

Figure 4.18: Schematic of the final design of the LIL tool. An Ar Ion laser (λ = 363.8 nm) is polarized with a polarizing beam splitter (PBS), then spatially filtered (S.F.) and split by a holographic grating. A resist coated wafer is positioned by a 2 axis motorized stage. Both beams interfere in the plane of the resist. Mirror positions can be adjusted to tune the interference period. Exposure time is controlled by a mechanical shutter. An optional refractive beam shaper (RBS) is inserted between the spatial filter and the grating beam splitter.

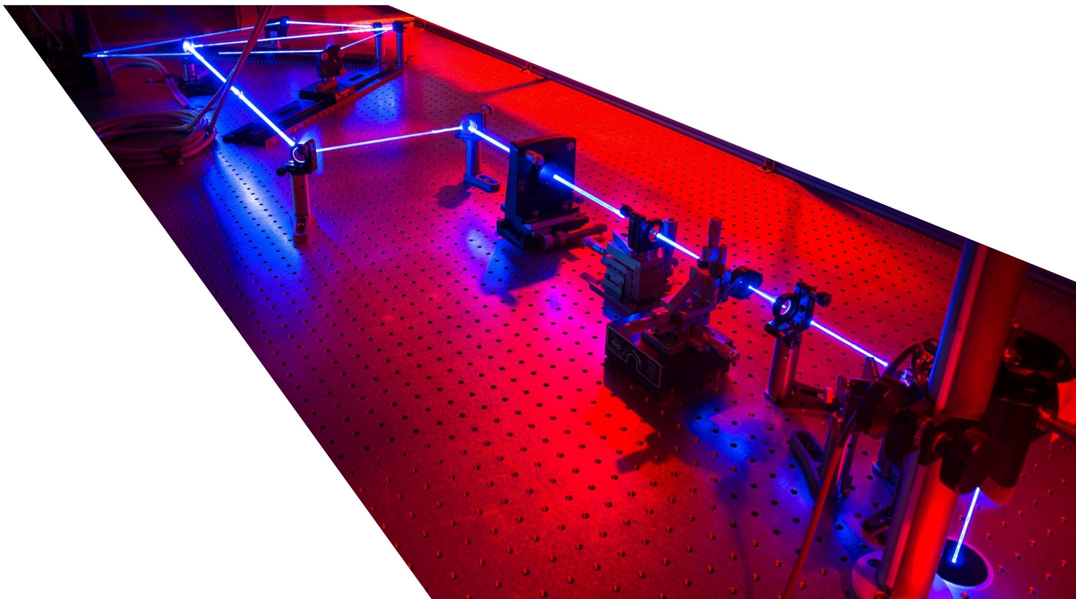

Figure 4.19: Beam path of an early implementation of the laser interference lithography tool.

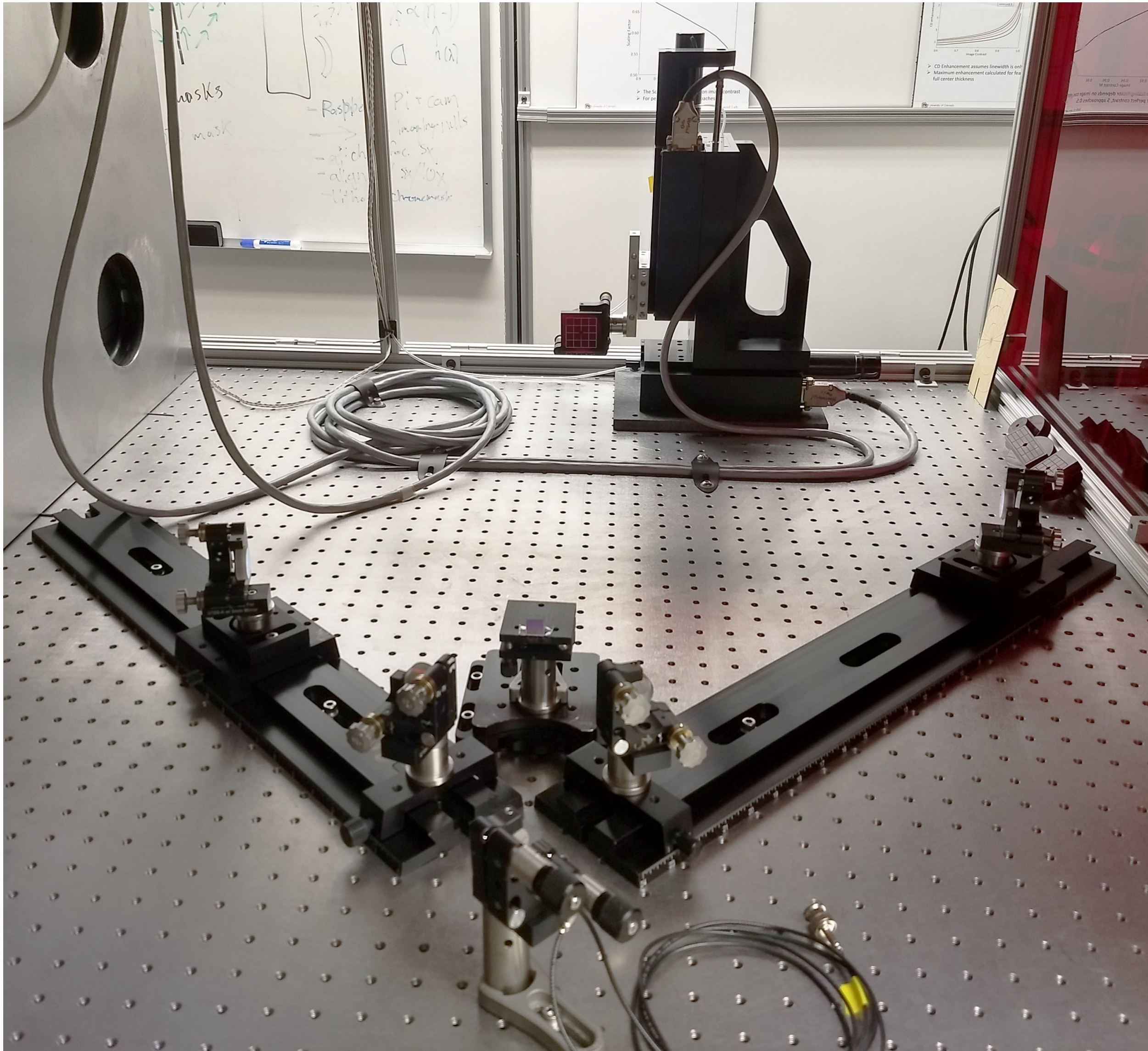

Figure 4.20: Final implementation of laser interference lithography tool. Beam conditioning optics are not shown.

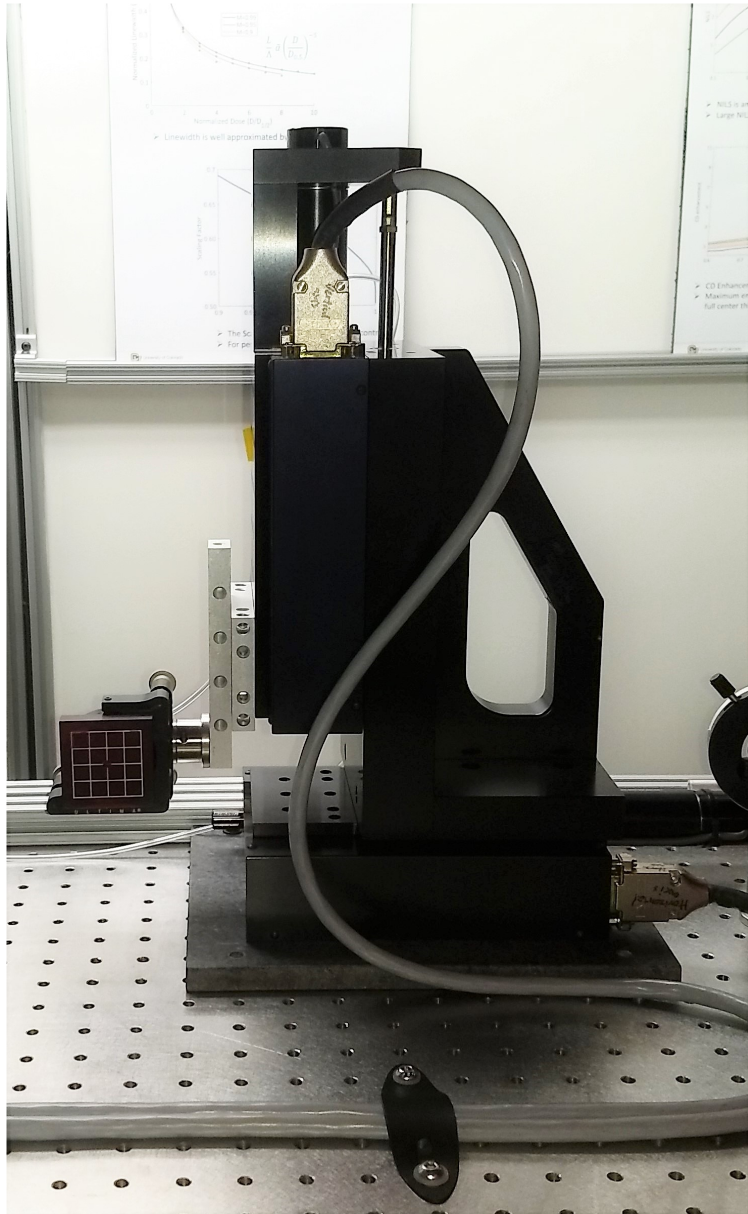

Figure 4.21: Wafer Handling. PM500 Stages position each wafer held by a vacuum chuck This eliminates wafer distortion and obscuration of the front side. The vacuum chuck is connected to a small vacuum pump (not shown) via tygon tubing.

Chapter 5

Demonstration and Model Validation

Laser interference lithography is used to demonstrate super-resolved critical dimensions and validate the models in Chapter 3. These experiments involved two different commercial, positive tone i-line photoresists. First, AZ 4210 (Merk) is a work-horse resist designed for 2-4 μm film thickness and a minimum linewidth of approximately 1 μm. Second, Ultra-i 123 1.0 (Dow) is a high resolution resist, designed for 0.6-1.1 μm film thickness and linewidths as small as 0.25 μm. While commercial resists were chosen for this work, a number of modifications are required to reach the desired feature sizes. First, the finite aspect ratio supported by the resists requires significantly thinner films. This requires dilution with propylene glycol monomethyl ether acetate (PGMEA), the typical delivery solvent for Novolak based resists. Second, soft bake time and temperature were optimized for repeatable processing, good adhesion, and minimal dark erosion. The common rule of one minute of bake per micron of film thickness did not apply to such thin films and instead optimal bake times and temperatures are similar to those for 1 μm thick films. Typical post-exposure bake parameters washed out the latent image in the photoresist. This is likely due to the diffusion length of carboxylic acid within the resist exceeding feature size. In the absence of post-exposure bake, the pattern fidelity was sufficient, so post-exposure bake was eliminated. Finally, developers free of metal ions, such as Microposit MF-319 (Shipley), yield higher pattern contrast with lower line-edge roughness than is typical with potassium hydroxide based developers, such as AZ400K. Simple immersion development proved adequate for this work. Details of the substrate preparation, and photoresist preparation, coating, and bake recipes are given in Appendix A.2.

As covered in Chapter 2, it is well known that back reflections off of the substrate are problematic for resist exposure. Typically, when working on a reflective substrate or substrates with large index mismatch such as silicon, a bottom anti-reflection coating (BARC) is applied between the substrate and photoresist. To isolate the proposed lithography technique from the impact of reflections, I used absorptive neutral density filters, Schott NG4 (1.9 mm thick), as a substrate. These filters are well index-matched at i-line to photoresist and generate very little back reflection – effectively acting as the ultimate BARC. Photoresist adhesion on a glass filter is similar to adhesion on an oxide film on silicon, so much of the work should translate. All threshold dose and contrast measurements are dependent on the substrate material, so proper BARC and soft bake optimization are required to replicate this work on silicon wafers.

5.1 Interference Contrast Estimation

Interference image contrast is estimated from measurements of the interferometer's beam intensities, polarizations, coherence length of the laser, and path length mismatch. A summary of these measurements is provided in Table 5.1. With the analysis outlined in Section 3 and values from Table 5.1, interference contrast is estimated to be 0.99 ± 0.005. Given these values along with optimized resist contrast, the expected minimum linewidth is close to $\Lambda/20$.

Table 5.1: **Interferometer alignment/component tolerances**

I_1	I_2	θ	DOP	Mutual Coherence
1	0.95 ± 0.01	$5° \pm 1°$	0.999 ± 0.001	0.995 ± 0.005

5.2 Resist Patterns

Keeping in mind a need to translate these results to steppers, spatial periods were chosen within the resolution of many i-line steppers. Interference patterns with period, Λ, of 250, 500 nm, 620 nm were used corresponding to NA = 0.73, NA = 0.36, and NA = 0.3 respectively. For patterns with Λ = 500 nm and 620 exposures with a range of doses were used to investigate critical

dimension scaling with dose as well as to find the minimum developable critical dimension.

Measured linewidth as a function of dose is plotted in Figure 5.5 for both resists. The measured scaling factor for AZ4210 was $S = 0.58 \pm 0.05$, while for Ultra-I $S = 0.57 \pm 0.03$. As predicted, linewidth scaling is the same for both resists, within experimental error. This result serves to validate the model which predicts the scaling factor should only depend on image contrast. Given these scaling factors and using the results of the theory section above, we find $M = 0.99 \pm 0.01$. Again, the estimated and measured image contrast agree within experimental error, further validating the model presented.

The minimum linewidth achieved for AZ4210 was 80 nm on a 620 nm period ($L/\Lambda \approx 0.13$) Figure 5.1. The minimum linewidth achieved for Ultra-i was 50 nm on a 500 nm period ($L/\Lambda = 0.1$) Figure 5.2. These correspond to minimum k_1 values of 0.065 and 0.05 respectively. Maximum achievable aspect ratios were 2:1 in AZ 4210, and 3:1 in Ultra-i. A fortuitous case of pattern collapse, shown in Figure 5.3, in Ultra-i allowed the sidewalls and feature depth to be directly observed alongside pattern dimensions. Prior to development, initial resist thickness was 150 nm. Using the collapsed pattern, the measured thickness of collapsed lines was 150 nm, indicating that the optical dose within the dark nulls of the aerial image is below threshold D_1 and providing confirmation that the resist thickness was unchanged within the dark nulls.

Given the measured resist contrast of $\gamma = 3.0$, the model predicts the minimum for each resist to be close to $\Lambda/20$ indicating another factor of two linewidth reduction should be possible. It is likely that other limits imposed by the photoresist and its processing have been encountered. For example, resist adhesion does not appear sufficient for linewidths less than 50 nm: smaller features exhibited delamination or pattern collapse. It is possible that improved resist formulations or process optimization specifically formulated for finer features could reach the lower limit.

These minimum critical dimensions were achieved with a total dose of approximately 1.5 J/cm^2. Exposures were made with a Gaussian intensity distribution with 3.9 mm diameter, and a total incident power of 3.5 mW. Total exposure time for the smallest features was 25 seconds. The laser used for these experiments lacks intracavity optics to maintain single frequency operation. It

was found that for lower laser power, the coherence length was over 1 meter, but for large output powers, the coherence decreased rapidly. For this reason, incident optical power was limited due to the need for large interference contrast. However, commercially available high power, single frequency sources are available, and powers in excess of 100 mW are readily attained. Even with an incident power of 100 mW, the same area could be patterned in under one second.

Exposing Ultra-i with a period of 250 nm, demonstrates that the same technique is possible with higher pattern density (Figure 5.4). As found above, the minimum feature size is again 50 nm. This demonstrates that the absolute lower bound on CD is set by the photoresist, not the interference pattern.

Finally, I analyzed pattern fidelity. Dimensional uniformity and LER were measured over multiple exposures. Details of these measurements are given in Appendix A.2. Typical linewidth variations are on the order 3 nm for all linewidths. This reproducibility demonstrates that linewidth can be deterministically controlled over a wide dynamic rage for a given exposure period. Typical measured LER for Ultra-i was 5 nm on 50 nm features, while typical measured LER for AZ 4210 was 11 nm on 80 nm features. Typically, LER should be no more than 10% of a feature's critical dimension. Given this, Ultra-i is suitable for patterns with critical dimensions 50nm and larger, while AZ 4210 is suitable for features larger than 100 nm.

These results can be replicated in a stepper using chromeless phase masks. Each phase edge results in an approximately parabolic null [28] with NILS = 4. Therefore, this can be treated using the analysis above. Unlike lithography with binary chrome masks, the minimum linewidth and linewidth variations are not set by the phase mask, but rather by the imaging system [28]. Specifically, linewidth variation across the field depends on illumination uniformity while coherence and defocus will determine the null contrast, setting the lower bound for minimum linewidth.

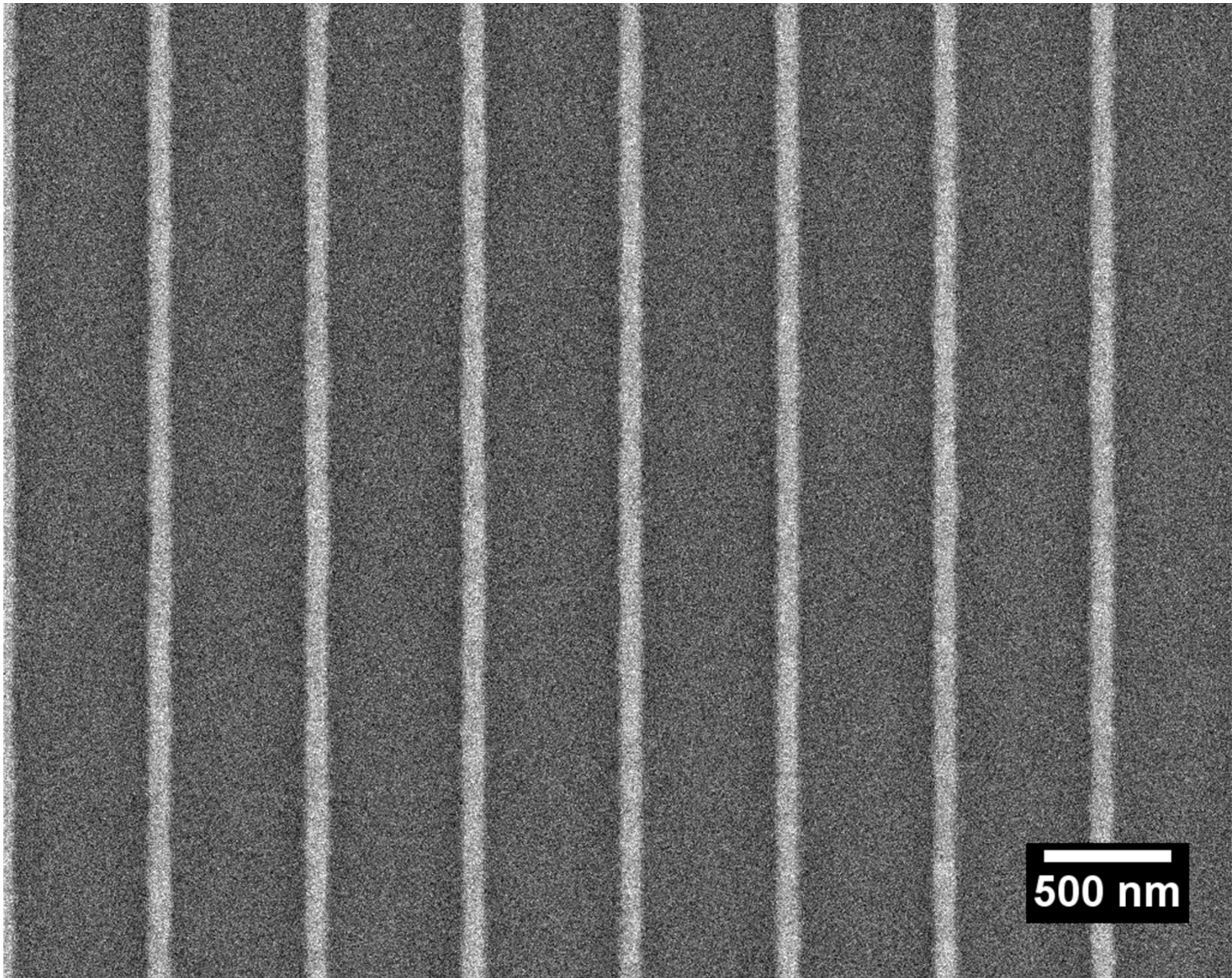

Figure 5.1: In AZ4210, the smallest CD achievable is 80 nm, with a thickness of 150 nm. These lines are on a 620 nm period, with a line edge roughness of 10 nm.

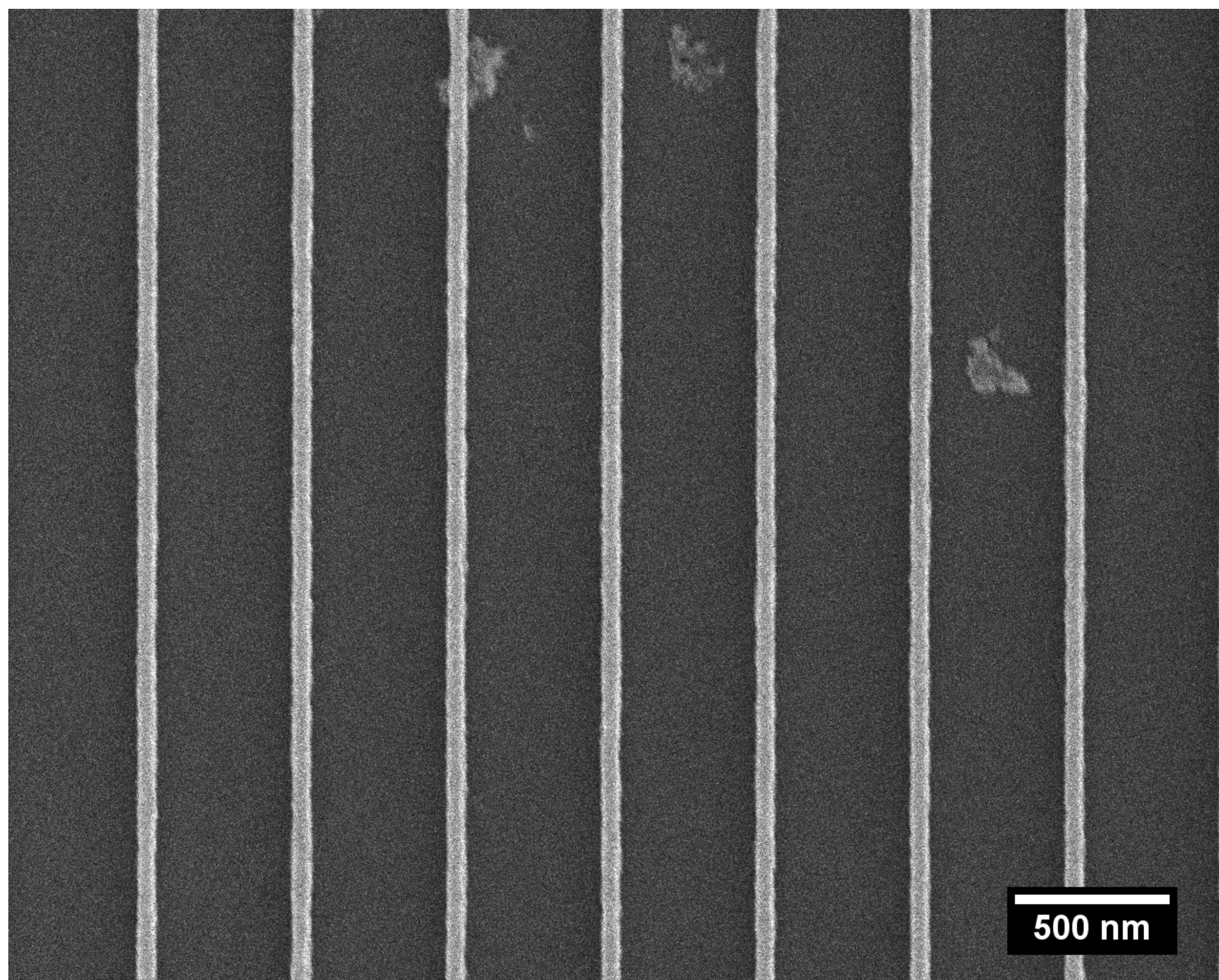

Figure 5.2: In Ultra-I, the smallest CD achievable is 50 nm, with a thickness of 150 nm. These lines are on a 500 nm period, with a line edge roughness of 5.5 nm.

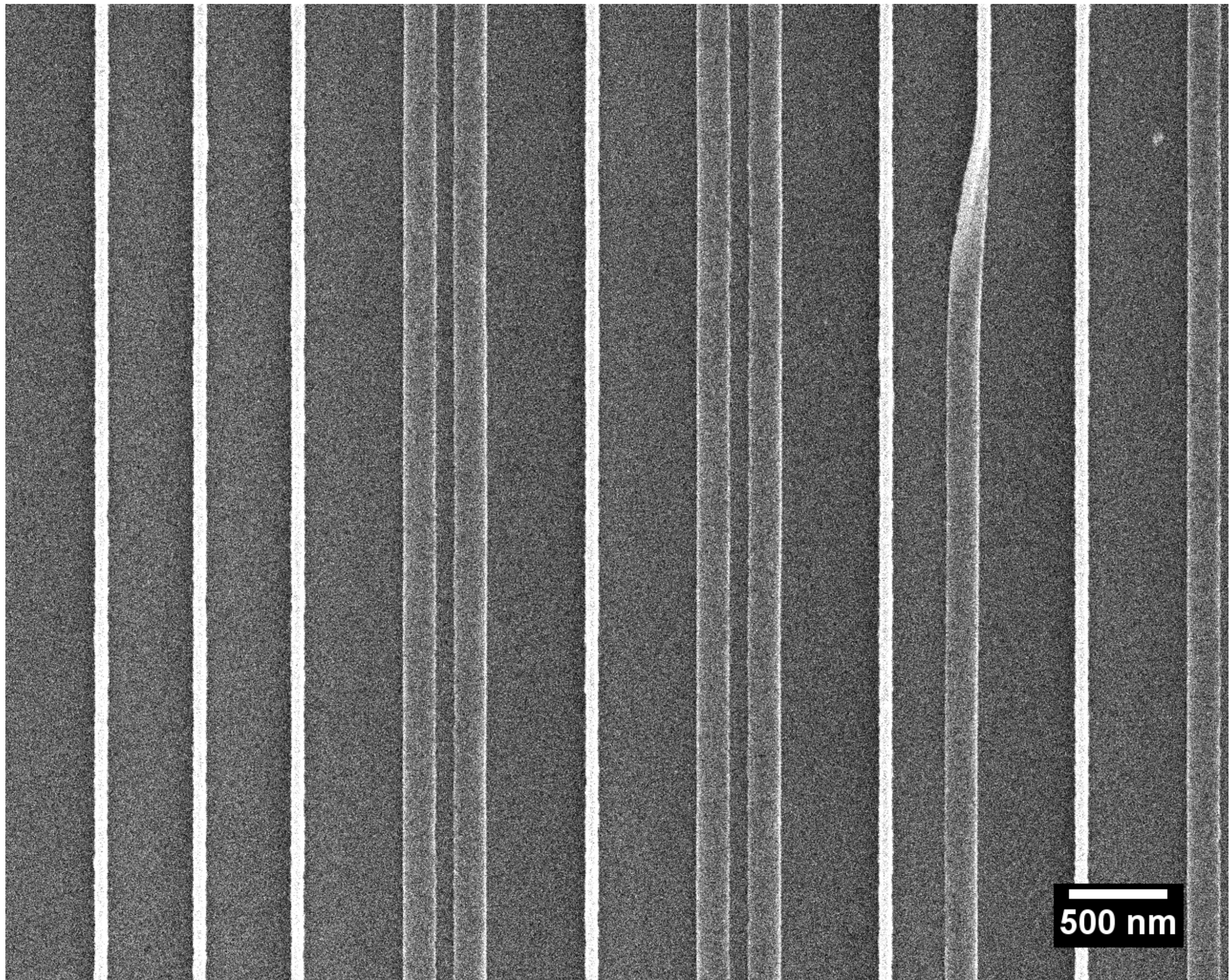

Figure 5.3: An image of pattern collapse observed in Ultra-i resist patterns. The linewidth is 50 nm on a 500 nm period. Several lines have collapsed within the pattern allowing depth and sidewall to be directly observed in the same image. The measured depth is 150 nm, giving an aspect ratio of 3:1.

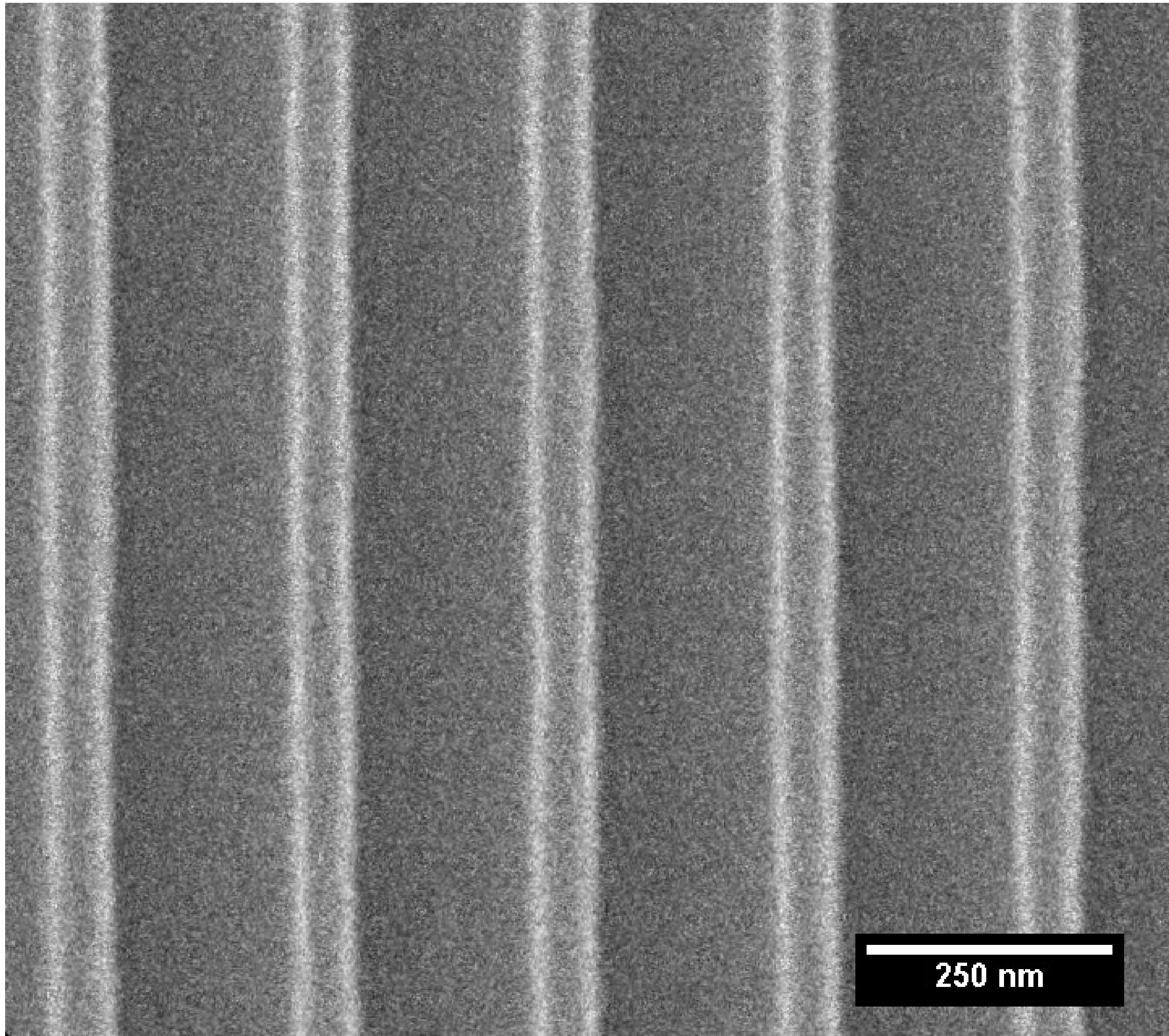

Figure 5.4: Higher pattern densities are possible. In Ultra-I exposed with a 250 nm pitch, the smallest CD achievable is again 50 nm, with a thickness of 150 nm. The lower bound on CD is set by the photoresist.

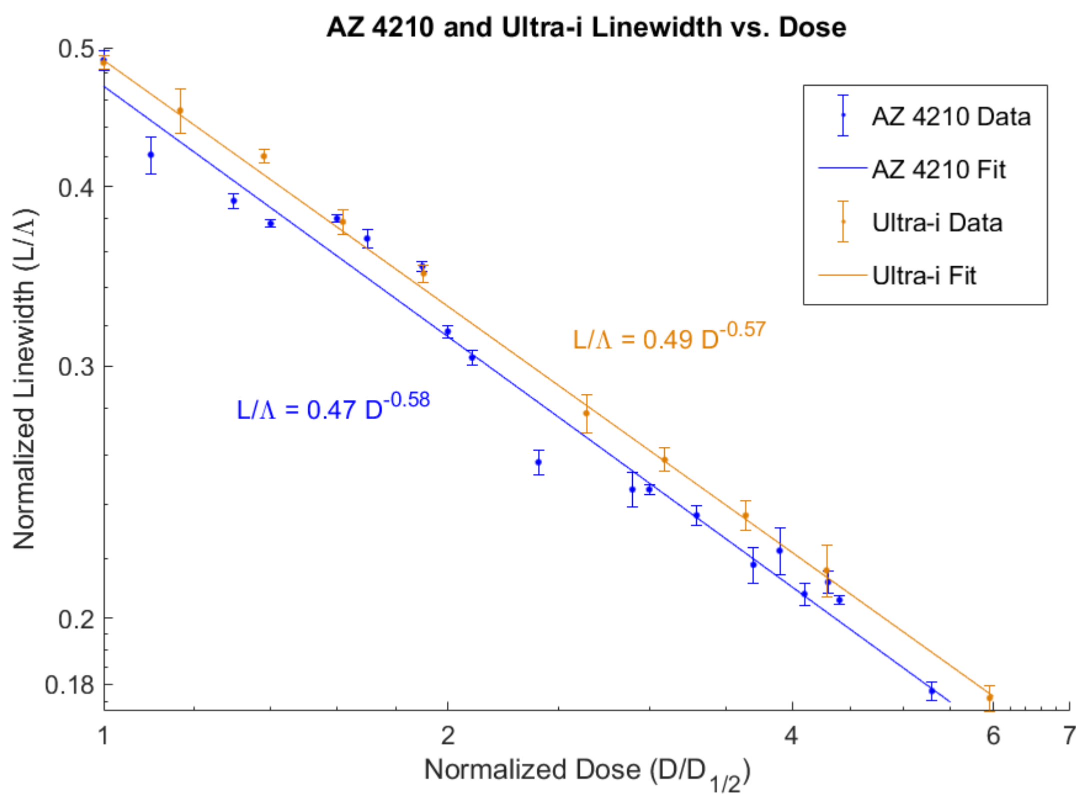

Figure 5.5: Linewidth dose scaling in both AZ 4210 and Ultra-i resist. Linewidth is normalized to period, and dose is normalized to the dose which produces half pitch patterns. Error bars represent 3σ variation in measured width.

5.3 Uniformity Improvement by Beam Shaping

Exposing resist with a Gaussian laser beam produces circular exposure fields with large radial dose variations, discussed in more detail in Section 4.3. Using a commercial refractive beam shaper, the exposure field was reshaped to be a square, flat top intensity distribution with approximately flat phase, as detailed in Section 4.4. Figure 5.6 illustrates the difference in gratings recorded with and without beam shaping. The left image contains gratings recorded with a Gaussian intensity distribution while the right image contains gratings recorded with a square, flat top intensity distribution. Using a beam profiler, the intensity distribution in the plane of interference was measured both with and without the refractive beam shaper in place; these measurements are given in Figure 5.7. With the beam shaper aligned, the intensity distribution was measured to have a standard deviation of 15%, and is expected to yield a linewidth variation within each exposure of $\approx 8\%$.

To validate, dose-exposure grids were exposed and developed. Linewidths were measured across the central 2 mm of each exposure field on a 0.5 mm grid. In addition, several random points within each exposure field were measured. SEM images were processed with SuMMIT using the same process as described above. The measured linewidth mean for these gratings was 121 nm with a standard deviation of 16 nm. This corresponds to a 13% variation (1σ) in linewidth. While larger than expected, it is explained given the many hot spots within the measured intensity distribution.

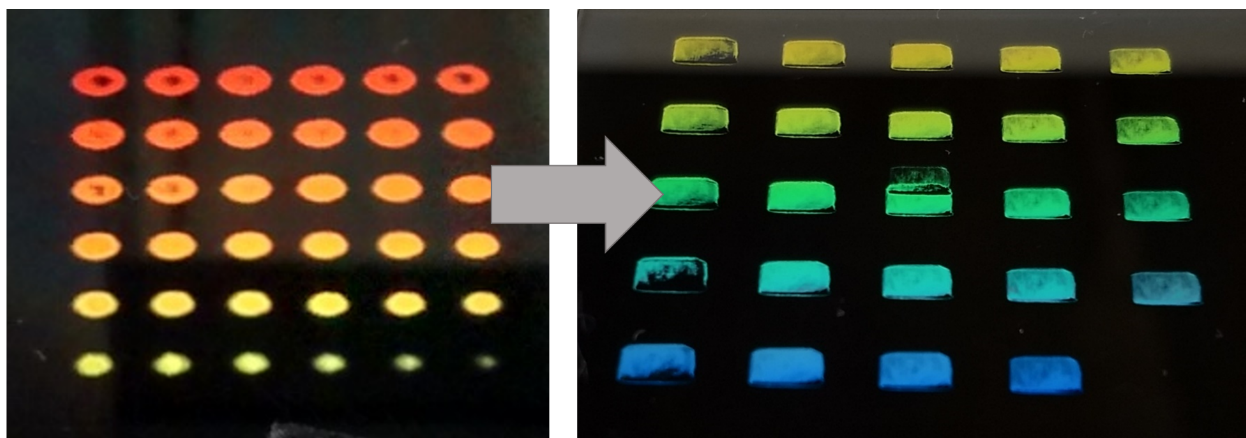

Figure 5.6: Comparison of gratings recorded with and without beam shaping. Both sets of grating were recorded on a 500 nm pitch, with a range of doses increasing from right to left, bottom to top. Left image contains gratings written with a Gaussian intensity distribution. The right image contains gratings written with a square, flat top intensity distribution.

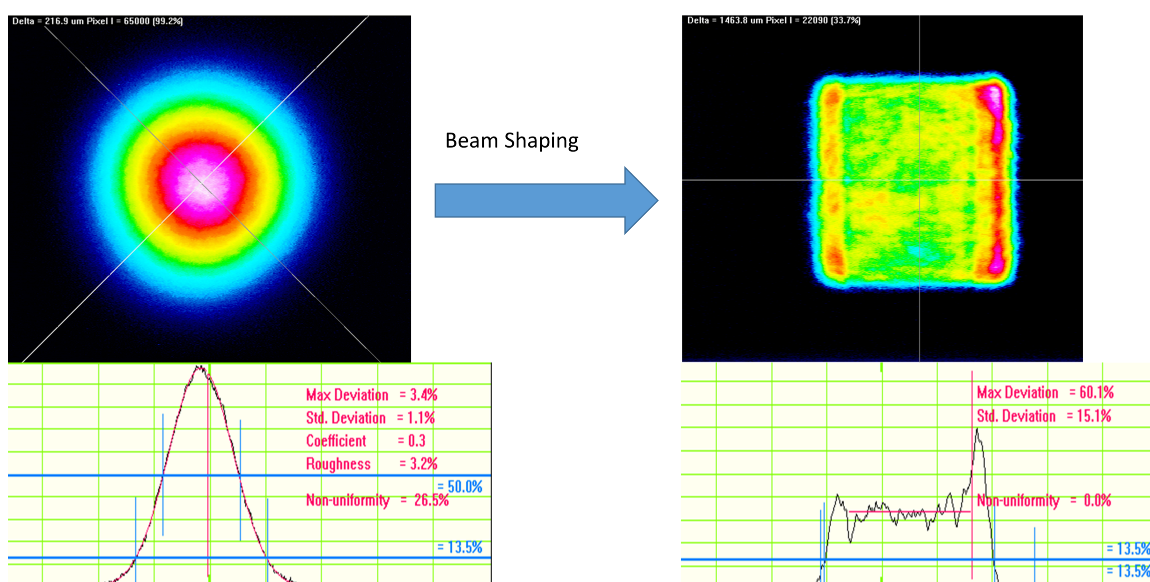

Figure 5.7: Measured intensity distribution in the wafer plane using a beam profiler. Left image shows the intensity of a Gaussian beam and cross section. Right image shows the intensity after a refractive beam shaper, as well as a cross section.

5.4 Beam Scanning

As shown in Section 4.2.3, the interference fringe positions are insensitive to change in beam pointing angle. This can be exploited to further homogenize the intensity by scanning the interfering beams while exposing resist. As the resist integrates the shifting intensity variations, these variations average out in the resist. An illustration of this is given in Figure 5.8. Using imageJ, the measured intensity was shifted one pixel at a time in the horizontal and vertical directions and summed. This was repeated for a total shift of ±20 pixels along each axis. Each pixel corresponds to approximately 5 μm, for a total shift of $\pm 100 \mu m$. The shifted image and its cross section on the right, are measurably smoother than the nominal intensity profile on the left.

This was implemented experimentally by adding piezo adjusters to the last fold mirror prior to the diffraction grating, visible in Figure 4.20. Each piezo actuator (Thorlabs, PE4) has 15 μm travel. Used with a standard 1" mirror mount, this provides an angular range of about 420 μrad. At the wafer, this equates to a maximum scan range of about 400 μm. A high voltage amplifier, modulated by a 1/2 Hz sine wave, drives the piezos. Slow scan speeds are used to minimize the possibility of exciting vibrations on the table. Exposing gratings while scanning did not wash out the grating pattern - sharp gratings were indeed recorded. The pattern uniformity was again measured by SEM and image analysis using the same methodology as above. Across several gratings, linewidth uniformity ranged from 4% to 9%. In all cases, linewidth within a the central regions of a single grating varied by less than 10%. This is in line with the original expectation of $\approx 8\%$ variation across the field. Beam scanning does in fact improve uniformity.

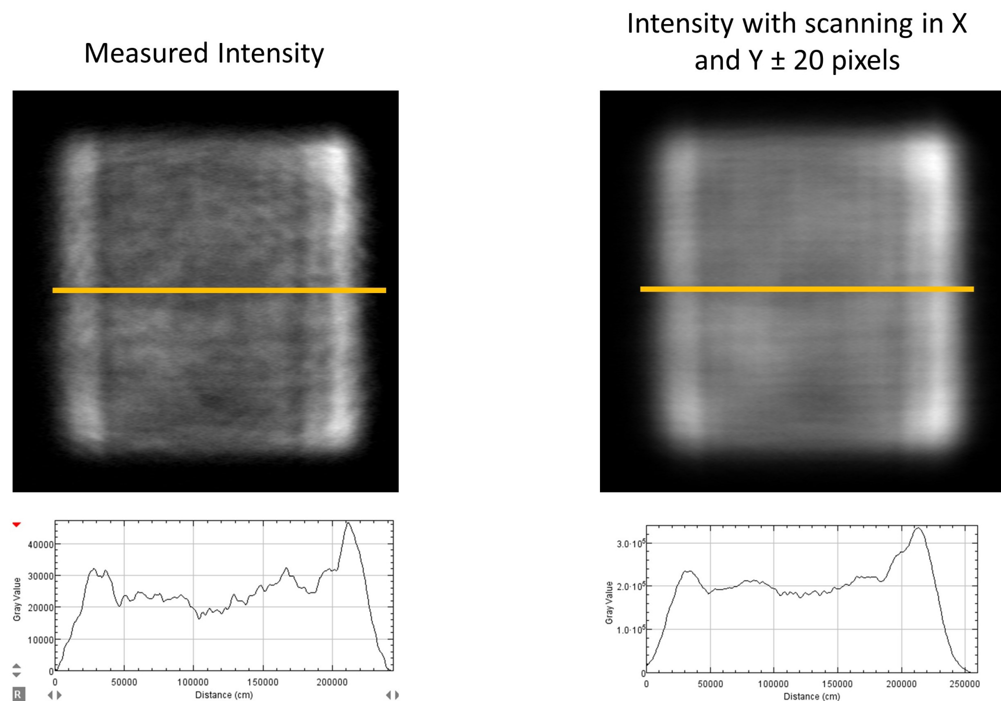

Figure 5.8: Comparison of wafer plane intensity with and without beam scanning. On the left, the stationary intensity distribution measured with a beam profiler, and a cross section (yellow line). On the right, an image of the wafer plane intensity with horizontal and vertical scanning and a cross section (yellow line). Using imageJ, the image was shifted one pixel at a time and summed for a total of ±20 pixels in X and Y. Each pixel corresponds to 5 μm.

Chapter 6

i-Line Stepper System

In order to test phase masks, a small scale stepper was designed and constructed in the lab. For mask development and testing, a projection lithography tool doesn't require the same level of functionality as does a production tool. This minimalist stepper can be broken into four main subsystems. These are mask illumination, projection optics, mask and wafer handling, and control system. The system design, construction, and performance are given in the following sections.

6.1 System Specifications

As this is a tool for University use, it will be used for research, device fabrication, and education. For simplicity, many features of commercial steppers are excluded. Instead, an R&D stepper for University use must be affordable, compact, easily maintained. Omitted capabilities include large field size, high throughput, robotic wafer handling and mask library, as well as automated mask-to-wafer alignment.

The full system fits on a standard optical table and is designed to handle 1" masks and 2" wafers. Alignment and focusing for the mask are performed manually with the aid of a camera. Wafer focus is performed automatically for every die. The illumination is flexible, allowing coherence adjustment from fully coherent, to fully incoherent. Likewise, the projection optics are flexible, allowing magnification and NA to be changed, as well as pupil plane filtering. The wafer handling and exposure are automated and controlled by Matlab script.

6.2 System Design

A schematic of the full optical system is given in Figure 6.1 and is pictured in Figure 6.2. The optical system is built on a vertical breadboard to allow the mask and wafer to lay horizontal.

6.2.1 Illumination

Illumination is provided by either an Ar-ion laser or UV LED. The laser provides coherent illumination, while the collimated LED provides either partially coherent or incoherent illumination. The laser is polarized, spatially filtered, and homogenized with a refractive beam shaper. To maintain linear polarization it is important that the polarization be in one of the eigenpolarizations of the fold mirrors. For this reason, polarization control is implemented with a half-wave plate immediately before the mask. Partially coherent and incoherent illumination are provided by a collimated LED assembly. The coherence factor, as explained in Section 2.8, is determined by the ratio of half angle divergence from the LED to the NA of the entrance pupil. The half angle divergence of a collimated extended source is given by

$$\theta_{1/2} = \frac{w}{2f}. \tag{6.1}$$

Here, w is the width of the LED and f is the focal length of the collimating lens. The LED (Thorlabs M365L2) has a 1 mm square emitter.

To control LED polarization, a polarizer is inserted before the mask in place of the half-wave plate.

6.2.2 Projection Optics

The projection optics consist of two microscope objectives. The first objective sets the input NA. The second objective sets the magnification and output NA. Magnification is the ratio of focal lengths, or NAs whichever is lower. Exit pupil NA is limited by

$$NA_{exit} \leq NA_{entrance} * \frac{f_1}{f_2}. \tag{6.2}$$

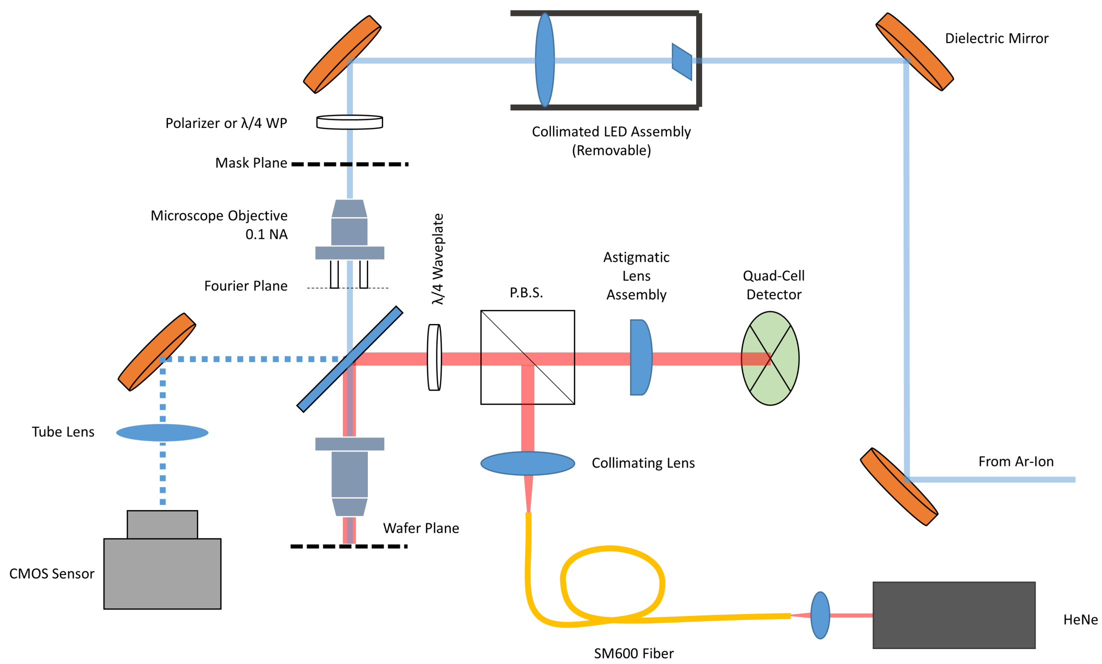

Figure 6.1: Optical schematic for a tabletop i-line stepper. The key systems are illumination, mask projection, and autofocus. Illumination is provided by either laser or collimated LED. Mask projection optics are composed of microscope objectives. The autofocus system is modeled after those found in optical disk drives.

Figure 6.2: Lab-built i-line stepper. The system is constructed on a vertical breadboard to allow the mask and wafer to lay flat and level. Mask alignment, including focus, is performed manually with a camera for feedback. Wafer alignment, including focus, is automated.

To implement pupil filtering, the Fourier plane of the first objective must be accessible. Many microscope objectives are designed as retrofocus lenses, placing the Fourier plane within the mechanical body of the objective, but long focal length, low magnification objectives are not generally of this design. The first objective, therefore, is low magnification, placing the Fourier plane beyond the end of the objective body. A pedestal is installed onto the backside of the entrance objective to hold a pupil filter in the Fourier plane. Between microscope objectives, a dichroic filter allows a second, visible beam to be injected for an autofocus system. A weak reflection from the backside of this dichroic also allows a camera to image the mask. Next to the imaging column, a CMOS camera is set up with a 150 mm efl tube lens for this purpose and is used for mask alignment and focusing.

6.2.3 Autofocus

The autofocus system is modeled after those used in optical disk drives. A HeNe is coupled into a single mode fiber serving as both a spatial filter and beam delivery system. A PCX lens collimates the output of the fiber and a PBS reflects S-polarized light. This reflected light passes through a $\lambda/4$ waveplate before reflecting off the dichroic. The red beam is focused onto the wafer by the second microscope objective. After reflecting off of the wafer, the light has polarization of reversed handedness. Then, after passing back through the waveplate, the polarization is linear and rotated 90 degrees (P-polarization). This allows reflected light to be transmitted instead of reflected by the PBS. Together, the waveplate and the PBS act as a poor-man's circulator, maximizing signal at the detector. This retro-reflected light is focused by a pair of cylindrical lenses onto a quad-cell detector. Error signals for position and focus are calculated as shown in Figure 6.3.

6.2.4 Mask and Wafer Handling

Masks are held by a mechanical chuck attached to a 3 axis manual stage, which in turn is mounted on a 5 axis stage for tip-tilt adjustment. As mentioned, mask positioning and focusing is entirely manual, using a camera for guidance. Wafers are held by a vacuum chuck mounted on top

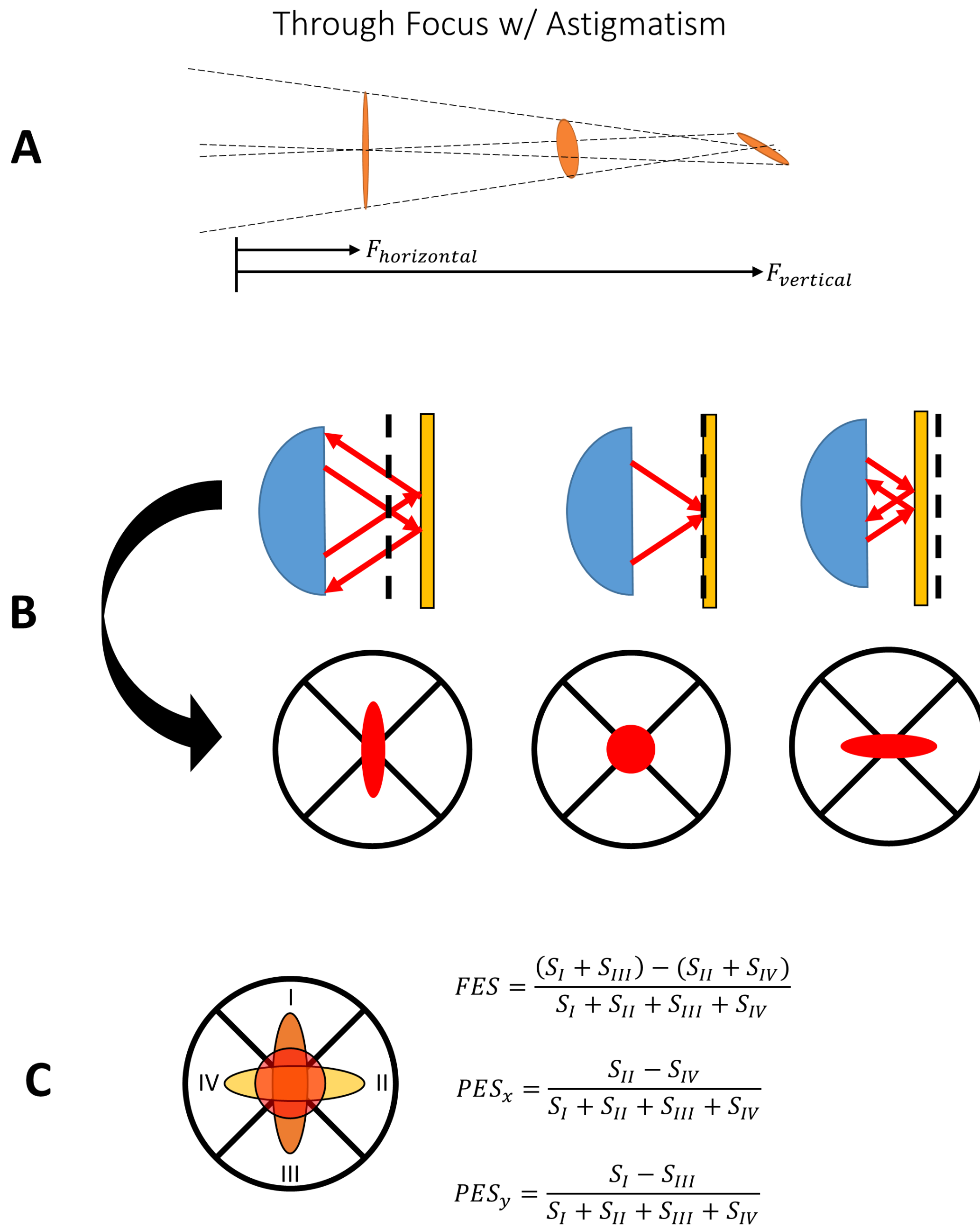

Figure 6.3: Autofocus detection. **A)** Through focus spot size after passing through a pair of cylindrical lenses. At best focus, the spot is circular. Before and after best focus, the spot becomes elliptical with different orientations. **B)** Illustration of backreflection from wafer (gold) and the resulting spots on the quad cell detector. **C)** The labeled quadrants of the detector with equations for focus error signal (FES) and position error signals (PES).

of 3 axis Newport XPS stage.

6.2.5 Electronics and Control

The autofocus detector signal from each quadrant is read via DAQ (LabJack U12) after the signal is amplified by a trans-impedance amplifier. Control signals for the shutter and LED are generated via Arduino, as well as signal timing. Full system control is performed by a Matlab script. The current version of this script exposes a dose-focus matrix across the wafer. For initial system alignment, stand alone codes read out focus error signals and position error signals. The control codes and stand alone autofocus readout codes are given in Appendix C.

6.3 System Performance

The system is currently configured with a 5x, 0.1 NA entrance objective (efl = 25.4 mm) and a 10x, 0.25 NA exit objective (efl = 16.5 mm). This provides a demagnification of about 1.54x, with NA = 0.154. Tests of auto-focus and imaging resolution are given below.

6.3.0.1 Auto-Focus

The autofocus fills the pupil of the exit objective, so the full NA is used regardless of the working NA for imaging. The depth of focus for NA = 0.25 with $\lambda = 633$ nm is

$$DOF = \frac{0.633\mu m}{0.25^2} \approx 10\mu m. \tag{6.3}$$

For a wavelength of $\lambda = 365$ nm, and working NA of 0.154, the depth of focus is

$$DOF = \frac{0.365\mu m}{0.154^2} \approx 15.4\mu m. \tag{6.4}$$

The DOF for red light using the objective's full NA is shorter than the DOF for imaging with UV and smaller working NA. An example of the focus error signal measured as a function of wafer displacement is shown in Figure 6.4. Neither objective is achromatic across UV and visible

wavelengths; both red and UV will focus to different planes. This distance between best focus for red and NUV was found experimentally. A coarse offset was measured using the mask camera and autofocus system. The mask camera was used to focus in UV, then the displacement of the mask was measured while using the autofocus error signal to focus in red. To fine tune focus offset, a dose-exposure matrix to find the offset with best image quality, then this offset was stored in the control code. Interestingly, the particular lens designs used in these microscope objectives places the UV plane is farther than the red focal plane. After finding best focus in red, the system shifts by the stored offset distance between best focus for each wavelength.

6.3.1 Air Force Resolution Test Mask

The Air force resolution test pattern was used to evaluate system resolution. First, each objective was evaluated by imaging the test mask using the mask camera system. Next, test mask was imaged into photoresist. Photoresist is processed in the same manner as in Chapter 5, and detailed in Appendix A.2. Resolution tests involved both coherent and partially coherent illumination of the mask.

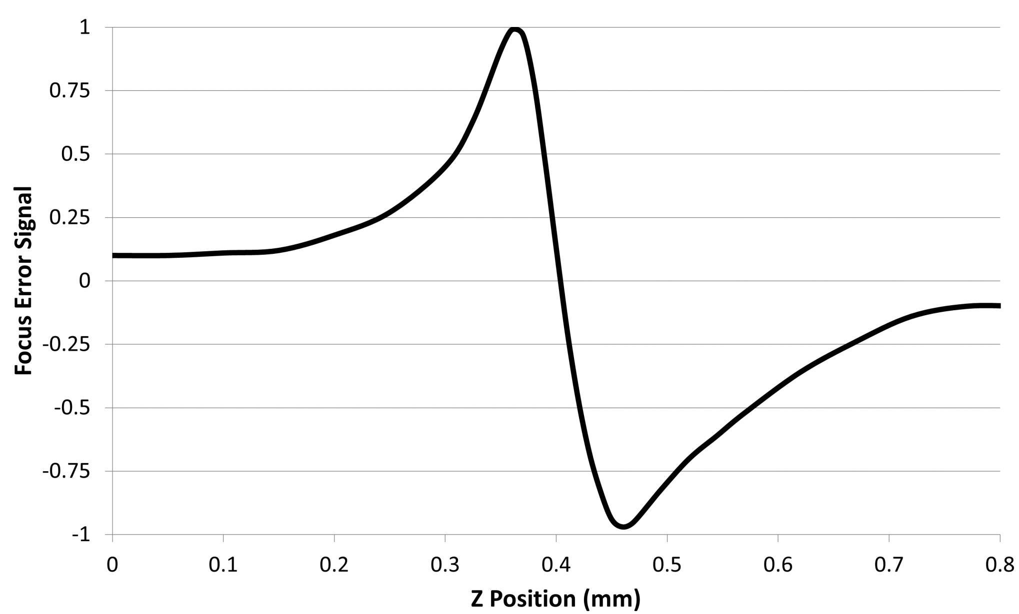

Figure 6.4: The through focus error signal as measured with a 10x 0.25 NA objective.

6.3.2 Contrast measured

The contrast at the highest spatial frequency, 228 line pairs per mm, was measured by image analysis. An image was captured via CMOS camera and processed in ImageJ. This was repeated with coherent and partially coherent illumination. The results for the two objectives are given in the table below. Typical images for the mask with coherent and partially coherent illumination are given in Figures 6.6 and 6.5.

Objective	Magnification	NA	Contrast(coherent)	Contrast(partial coherence)
1	5x	0.1	0.8	0.3 ($\sigma = 0.1$)
2	10x	0.25	0.85	0.6 ($\sigma = 0.04$)

Table 6.1: Summary of measured image contrasts

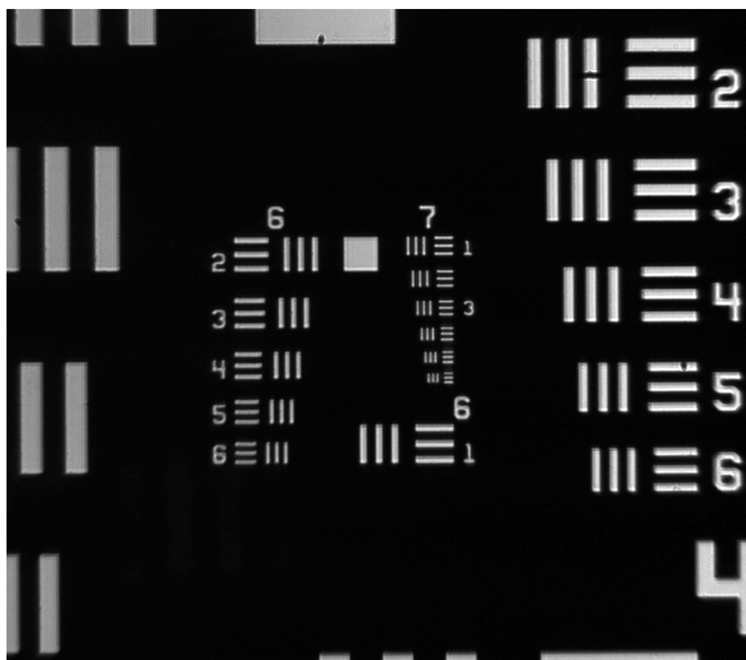

Figure 6.5: AFRT illuminated with LED (coherence $\sigma = 0.1$). Imaged with 5x objective and CMOS camera with a 150mm tube lens

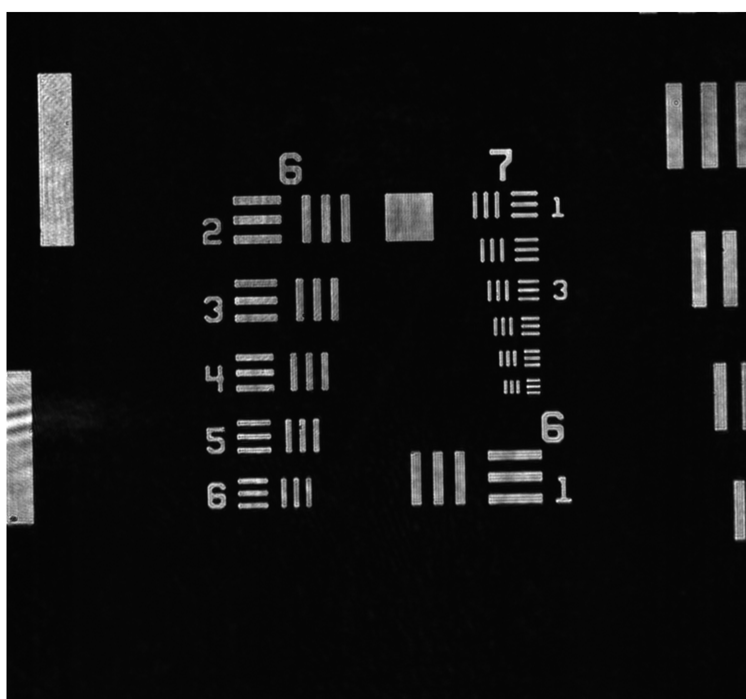

Figure 6.6: AFRT illuminated with laser (spatially coherent). Imaged with 10x objective and CMOS camera with a 150mm tube lens. The largest spatial frequencies (element 6, group 7) are better resolved than with partially coherent illumination, but coherent artifacts are clearly visible.

6.3.3 Developed Resist Patterns

The AFRT was imaged into Ultra-i photoresist with both fully coherent illumination, $\sigma = 0$, and partially coherent illumination, $\sigma = 0.1$. Optical microscopy was used to inspect the developed resist patterns, and typical images are given in Figures 6.7 and 6.8. Coherent illumination yields better resolution: the smallest features resolve fully, whereas with partial coherence they don't, as expected from contrast measurements. However, partially coherent illumination reduces line edge roughness and eliminates coherent artifacts (ringing).

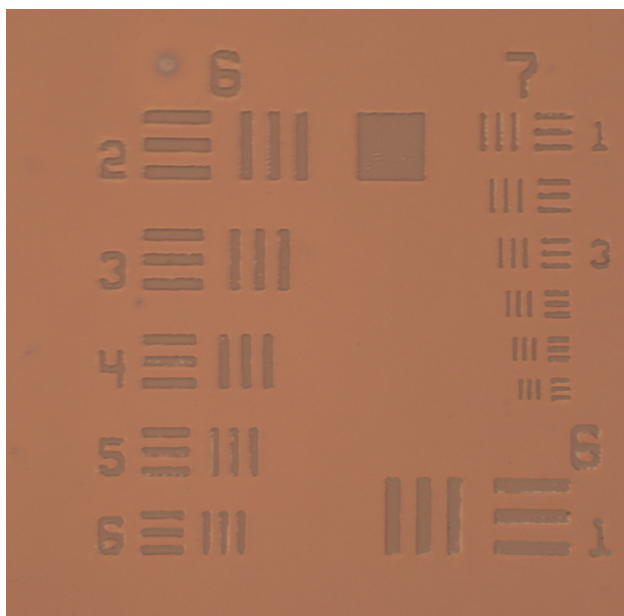

Figure 6.7: Optical micrograph of developed resist patterns exposed with fully coherent (laser) illumination. While the resolution is improved over partially coherent illumination, line edge roughness is poor, and the pattern often suffers from coherent artifacts.

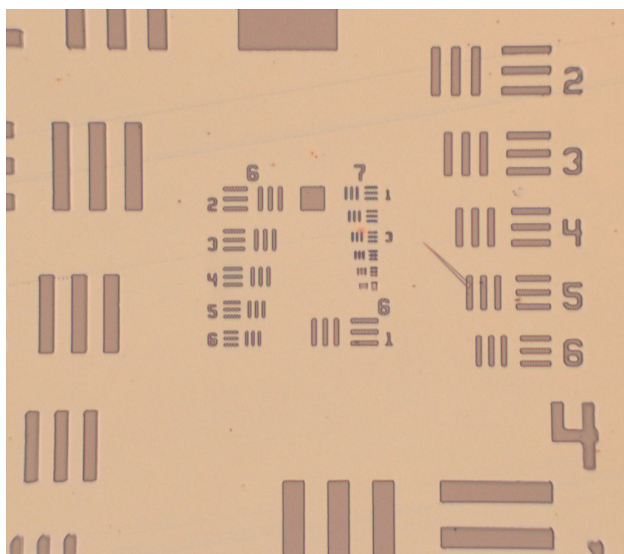

Figure 6.8: Optical micrograph of developed resist patterns exposed with partially coherent (collimated LED) illumination. While the resolution is reduced due to diminished contrast, line edge roughness is greatly improved, and the pattern lacks any coherent artifacts.

Chapter 7

Pixelated Polarization Phase Masks

As discussed in Chapter 2, phase masks offer the most promising resolution enhancements. However, phase masks suffer from several problems. The first problem is that of polarization dependent contrast. To illustrate this, consider imaging a phase grating illuminated with P-polarization as pictured in in Figure 7.1. As discussed in Chapter 3, the image of this can be thought of as plane waves emanating from the exit pupil of the projection lens. The contrast of such an image was analyzed in Chapter 4: for linearly polarized light, this image contrast scales as

$$M \propto DOP * cos(\Theta) \tag{7.1}$$

where Θ is the angle between interfering polarizations. As seen from Figure 7.1, in the case of P-polarization, the interfering polarizations can be orthogonal leading to zero contrast. Optical contrast for two beam imaging with a grating, in air, as a function of polarization orientation and angle of incidence is given by

$$M \propto sin^2(\phi) \left| cos^2(\theta) - sin^2(\theta) \right| + cos^2(\phi). \tag{7.2}$$

Here, ϕ is the polarization angle relative to S-polarization and θ is the angle of incidence. Substituting the definition for pitch resolution, contrast scales as

$$M \propto sin^2(\phi) \left| 1 - \frac{\lambda^2}{2\Lambda^2} \right| + cos^2(\phi). \tag{7.3}$$

Finally, contrast in the resist can be found by substituting the definition of NA

$$M \propto sin^2(\phi) \left| 1 - 2\frac{NA^2}{n^2} \right| + cos^2(\phi). \tag{7.4}$$

Here, NA is that of the projection optics in air and n is the index of the resist. For large pitch, low NA patterns, polarization has minimal impact on contrast. However, for high NA, the contrast for S and P-polarizations diverges, as shown in Figure 7.2A. For NA=0.707, the aerial image contrast for P-polarization drops to zero. However, in resist, refraction at the air-resist interface reduces the angle of incidence, and as a result, the interfering beams can never be orthogonal. For a typical i-line resist with $n = 1.6$, the largest angle for a single beam is 38.7 degrees leading to a contrast of $M \approx 0.22$. For comparison, the NA dependent contrast for S and P-polarizations in resist are given in Figure 7.2B.

From this analysis, it is clear that in order to maximize image contrast, polarization must be aligned along phase steps. As a result of this, contrast cannot be maximized for orthogonal phase steps within the same phase mask. By decomposing the patterns into multiple phase masks, each can be illuminated with the appropriate polarization, but, doing so increases the number of process steps required to write lines in multiple orientations using phase masks. Additionally, to be decomposed into multiple masks, these patterns need to follow a Manhattan geometry. This constraint requires a change in chip layout [33].

If the local polarization can be controlled at the mask plane, polarization dependent contrast can be solved, allowing for arbitrary 2D patterns while retaining high image contrast. Mask level polarization control provides a significant new degree of freedom for lithographic pattern design and mask optimization. Pixelated polarization states along with π phase steps are illustrated in Figure 7.3. Here, the dark lines represent desired phase nulls. The yellow arrows represent the polarization orientation and phase, where arrows 180 degrees apart represent a π phase step.

Traditional phase mask geometry is further limited by both the line end problem and the phase conflict problem. Many layouts cannot be written with a traditional phase mask, such as the

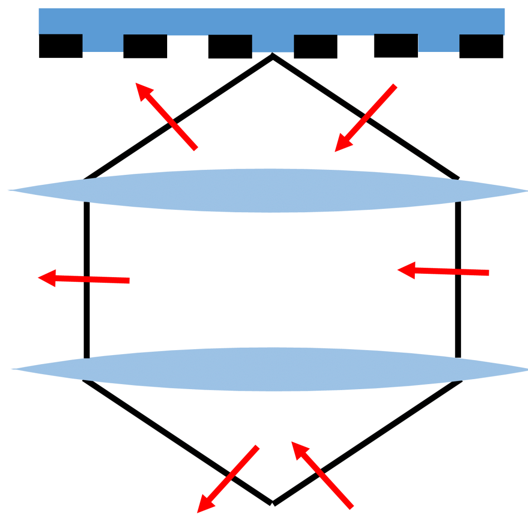

Figure 7.1: Imaging an alternating phase shifting mask illuminated with P-polarization. When interfering polarizations are orthogonal, image contrast is zero. In the case of S-polarization, orthogonal polarizations never occur.

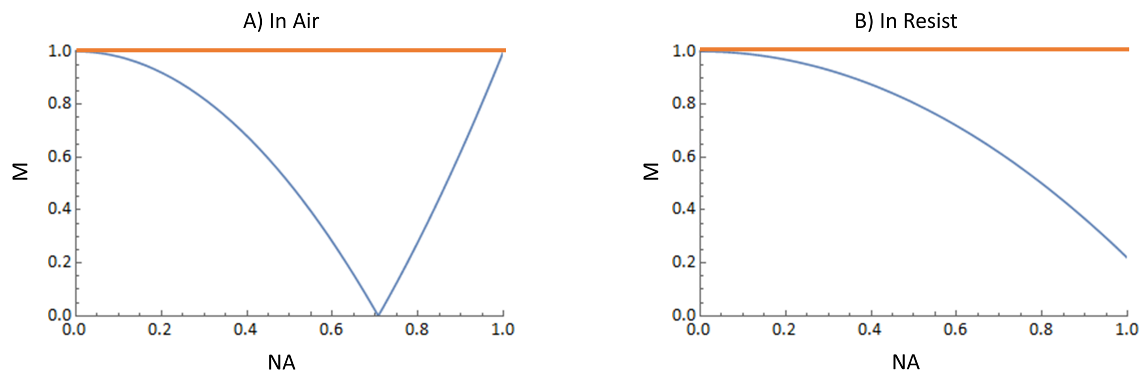

Figure 7.2: Image contrast of phase grating as a function of illuminating polarizations: P-polarization (blue lines) and S-polarization (orange lines). A) Image contrast in air. The P-polarization contrast drops to zero for NA = 0.707 (45 degrees). B) Image contrast in resist (N = 1.6). The P-polarization contrast never drops to zero, but reaches a minimum at M = 0.22.

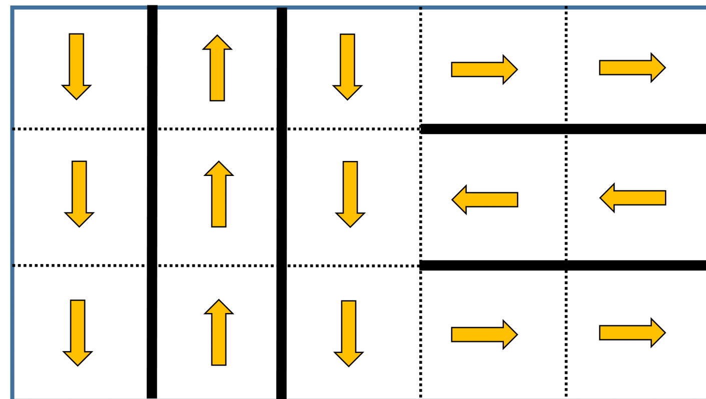

Figure 7.3: Example of multiple orientations of phase steps within single pattern enabled by pixelated polarization control. The dark lines indicate the resulting intensity nulls. Arrows indicate polarization orientation, and direction indicates phase. For polarizations π out of phase, arrows differ by 180 degrees.

3-way intersection or alternating line segments, as discussed in Chapter 2. Local phase control not only solves the the issue of image contrast, but as shown in this chapter, also solves the the other major issues of line ends and phase conflicts. Altogether, resolving these three issues eliminates the need for pattern decomposition, multiple patterning steps, and allows a broader range of pattern geometry to be realized.

This chapter develops pixelated polarization phase shifting masks (P^3SM), a novel lithographic mask design that controls both local polarization and phase for transmitted light. In this thesis, simultaneous polarization and phase control is realized using photopatterned liquid crystal (LC) polymers. Why these polymers are suitable for use in the NUV, they likely are not for DUV. However, as is discussed in Chapter 8 other means of polarization control are possible, including pixelated wire-grid polarizers and sub-wavelength photonic structures, that would be suitable for DUV lithography. Masks designs are simulated with a vector beam propagation code, and the liquid crystal based masks are fabricated and tested with an i-line stepper.

7.1 Projection Lithography Simulations

Projection lithography with a mask is simulated using Fourier optics codes. Electric fields for three principle planes are modeled: the mask plane, the pupil plane, and the image plane. The image plane is the transform of the pupil plane, and the pupil plane is the transform of the mask. This simplified imaging system simulation is depicted in Figure 7.4.

7.1.1 Full Vector Simulations

To model the pixelated polarization phase shifting masks, full vector simulations are required. Polarized projection lithography is simulated by implementing a vector Fourier optics model in a Matlab Script, given in Appendix D. The physical basis for the code comes from [35].

As described above, three principle planes are modeled. First, the mask E-field is computed and decomposed into orthogonal transverse field components, E_x and E_y. Then, each component is Fourier transformed. These Fourier components for each polarization state are projected onto

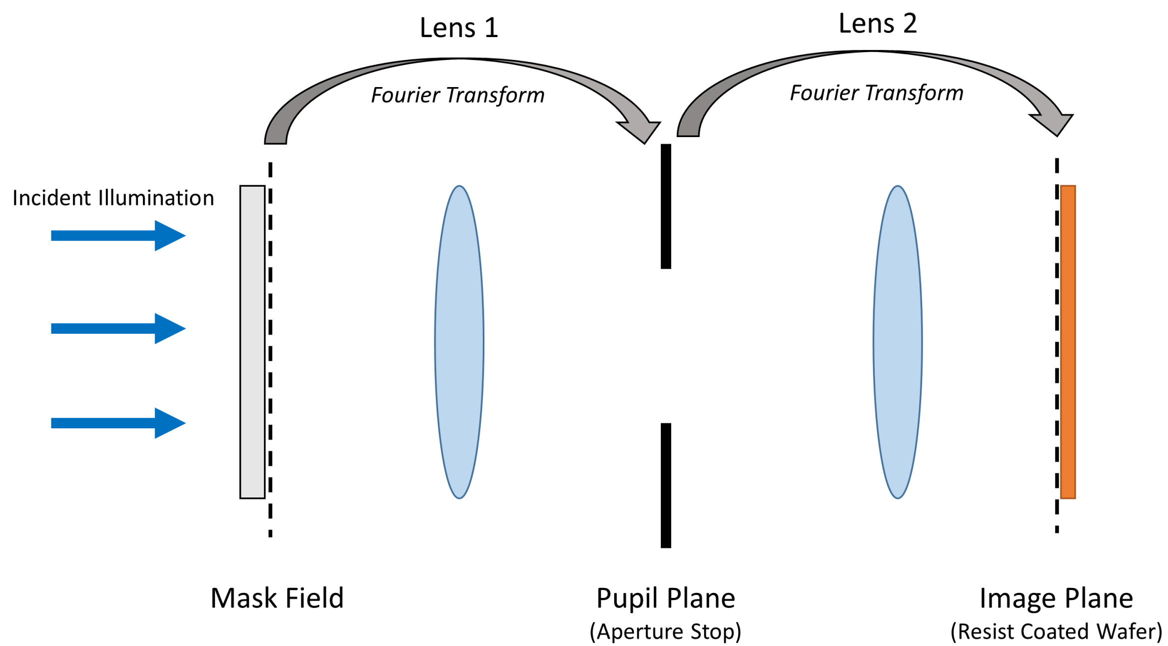

Figure 7.4: Simplified imaging system model used for vector Fourier simulation. The electric fields are computed for the mask plane, the pupil plane, and the image plane. The pupil is the transform of the mask, and the image is the scaled transform of the pupil.

a new basis set consisting of ordinary, $E_{ordinary}$, and extraordinary, $E_{extraordinary}$, polarizations pictured in Figure 7.5. Once in ordinary and extraordinary basis, these are the pupil plane E-fields. To simulate filtering, optical aberrations, defocus, and radiometric corrections, the appropriate functions are applied to the pupil plane e-fields. K-space coordinates are scaled to account for magnification, then each pupil field is transformed and projected back onto a Cartesian basis set E_x, E_y, E_z. The sum is the full vector E-field in the image plane. The image intensity is the squared magnitude of the full E-field.

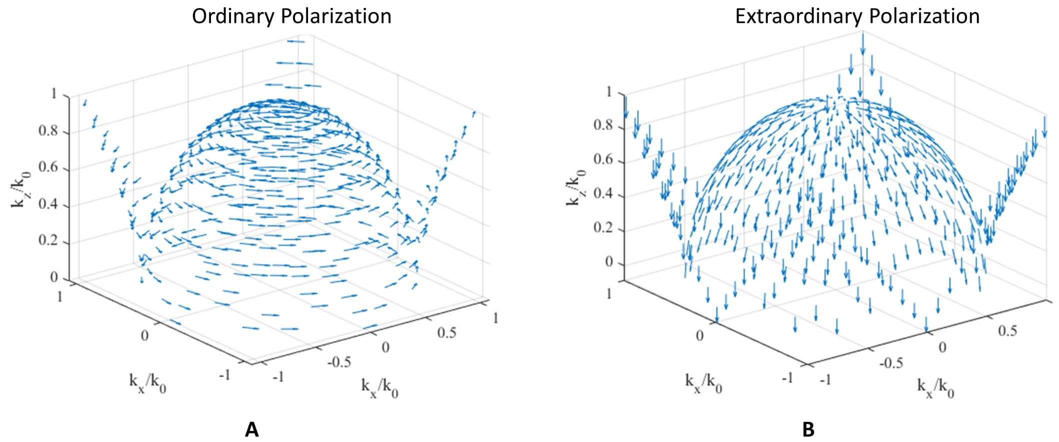

Figure 7.5: A) Ordinary and B) Extraordinary Polarization basis set used in the pupil plane

7.1.2 Polarized Illumination

The illumination is assumed to be coherent and uniform over the whole mask with the incident polarization defined by a Jones vector. To simulate depolarized light, the simulation runs twice, once with S-polarized illumination, and once with P-polarized illumination. The final image is the incoherent sum of image plane intensities from each simulation. In addition to polarization

control, illumination may be tilted to simulate off-axis illumination and to allow simulations of partial spatial coherence. Tilt is implemented as a linear phase across the mask plane.

7.1.3 Coherence

Spatial coherence is simulated using the extended source method [34]. An extended source with angular size determined by σ and $\mathrm{NA}_{entrance}$ is decomposed into discrete point sources in angle space. Each point source is coherent, and each source is uncorrelated with every other source. The simulation iterates over all sources and the final image plane intensity is computed as the incoherent sum of the image from each simulation.

7.1.4 Mask File and Mask Fields

The mask design is saved as an array of waveplate orientations in units of radians. To compute the mask field, Jones matrices are computed for every pixel. Then, the mask field is computed pixel-by-pixel using those Jones matrices, incident Jones vector, and inclined phase due to off-axis illumination. When implementing tilted illumination, the mask is assumed to be thin, such that is tilt does not change the retardance of the mask.

7.1.5 Resist Simulation

Current code does not account for refraction at the resist surface. The current code also assumes the resist is on top of a BARC so that back-reflections and standing waves are negligible. Resist profiles are computed using the model from Chapter 3. Linewidths are then computed as the width at the base of the feature.

7.2 Mask Implementation

LC polymer masks are designed as pixelated $\lambda/2$ wave plates. Fast axis orientation is controlled on a pixel-by-pixel basis by a photoaligned azobenzene alignment layer. LC monomer is deposited to the appropriate thickness for a half-wave of retardance and is polymerized, fixing the

orientations of every pixel. As is true of all half wave plates, linear polarizations are rotated by twice the angle of the waveplate rotation and adjacent pixels with orthogonal axes produce a half-wave phase step across the pixel boundary. In this way, polarization can be optimized locally for any orientation of phase step desired on the mask. These masks are therefore illuminated with linear incident polarization. A unique benefit to LC phase masks is the they have planar topography. The non-planar nature of alternating PSMs can lead to image artifacts. Finally, the LC masks demonstrated here are chrome-less enabling higher throughput as all photons go towards the aerial image.

7.3 Mask Layout

Pixelated polarization phase mask design is more complex than chrome on glass masks due to the additional degree of freedom afforded by local polarization control. Some initial work in mask design strategy and optimization is presented here. There remains ample room for the development of formal mask design rules and optimization strategies.

For some of the designs in this thesis, waveplate orientations are manually laid out. However, to automate the design, it is possible to set the waveplate orientations as a solution to the Laplace equation. Waveplate orientations are fixed to either side of a phase null, and are fixed around the mask boundary, then using an Laplace solver, each remaining pixel orientation is computed as the average of the orientations of its four nearest neighbors. This solver iterates until it converges on a stable solution. Examples in Section 7.4.1 illustrate the difference between a Laplace solution mask, and a naïve manual design. A Laplace solution to pixelated polarization mask design has two advantages. First, energy of the liquid crystals is minimized [20]. Second, polarization changes slowly pixel to pixel, minimizing polarization diffraction. So long as polarization varies slowly pixel-to-pixel, spurious nulls are not introduced. However, some boundary conditions can introduce intensity nulls as a result of geometric phase. An example of this is given in Section 7.4.3.

7.4 Example Lithographic Patterns

Masks were simulated with the following parameters: $NA_{entrance} = 0.1$, $NA_{exit} = 0.9$, Demagnification = 9x, $\sigma = 0$, normal incidence, resist contrast $\gamma = 3.5$. All masks are designed to be illuminated with vertical polarization.

7.4.1 Line Ends

While traditional phase shifting masks lack the ability to terminate a line, P^3SM can terminate lines using the additional degree of freedom provided by polarization. Two example mask polarizations are given in Figure 7.6. Line ends can be terminated using any polarization state that doesn't produce significant interference. So long as any resulting intensity null is of low contrast, it won't print in the resist.

Finite lines were modeled with two different mask designs. The first mask was designed using orthogonal polarizations to terminate a finite line segment, shown in Figure 7.6. The projected image, Figure 7.8 does contain a finite length vertical null, but also two undesired horizontal nulls due to polarization diffraction. These horizontal nulls are shallow enough that even with nominal exposure dose, no line prints in the resist. Figure 7.10 shows the developed resist pattern with nominal exposure. Looking at the intensity cross section of the projected image, Figure 7.9, the null follows the expected Sine Integral function [28].

A second mask was designed using a Laplace solution, shown in Figure 7.11, to terminate a finite line segment. The outer boundary was selected to be uniform, vertical polarization all around. As a result of the slowly varying polarization states, the spurious nulls are eliminated, shown in the projected intensity in Figure 7.12. Interestingly, the line null depth, Figure 7.13 is reduced compared to the prior mask design.

It is interesting to consider the end of each printed line segment. For the first mask, the line ends are bulbous. For the second mask, the line ends are slightly rounded and bent. The natural linewidth of an isolated phase step is given by [28]

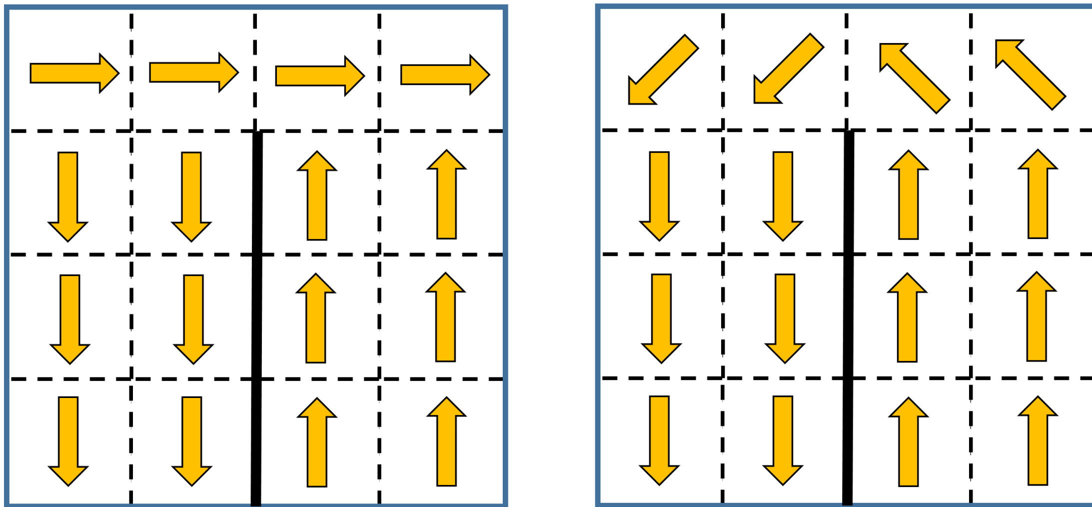

Figure 7.6: Two potential strategies for terminating line ends with polarization.

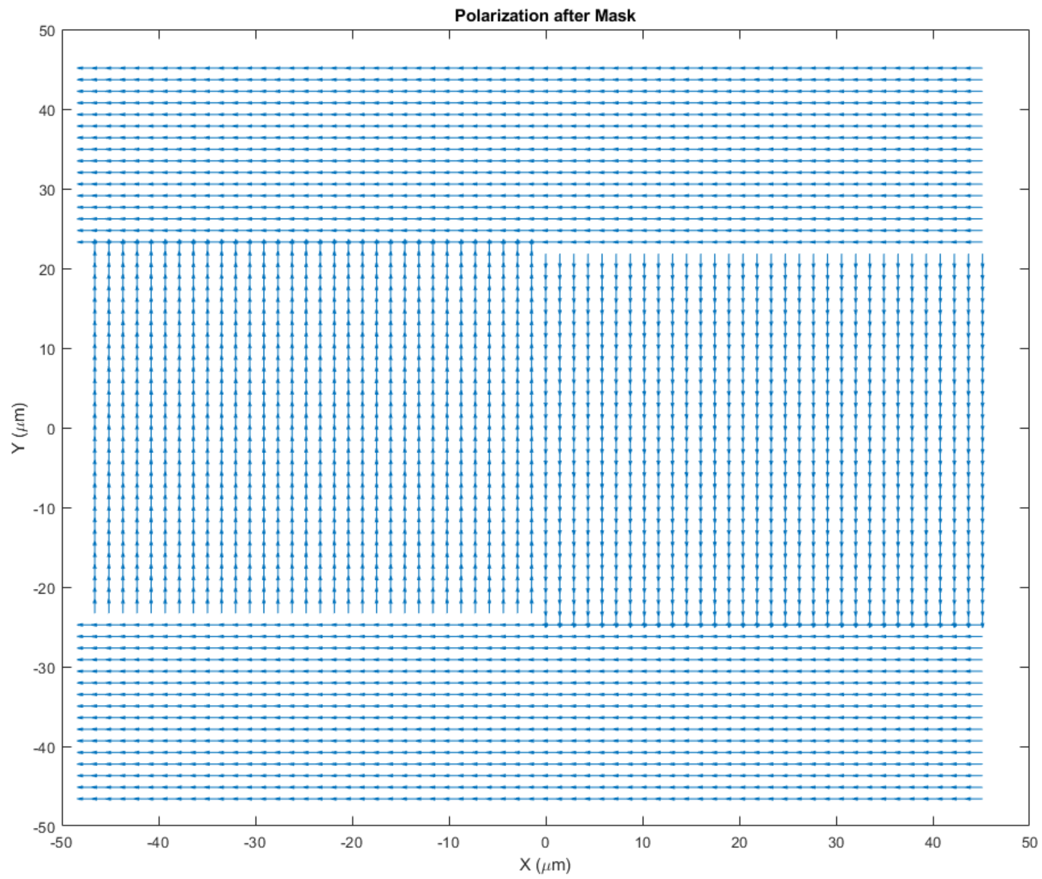

Figure 7.7: Polarization states for a finite line segment. This mask was manually laid out using simple orthogonal polarizations to terminate the line ends.

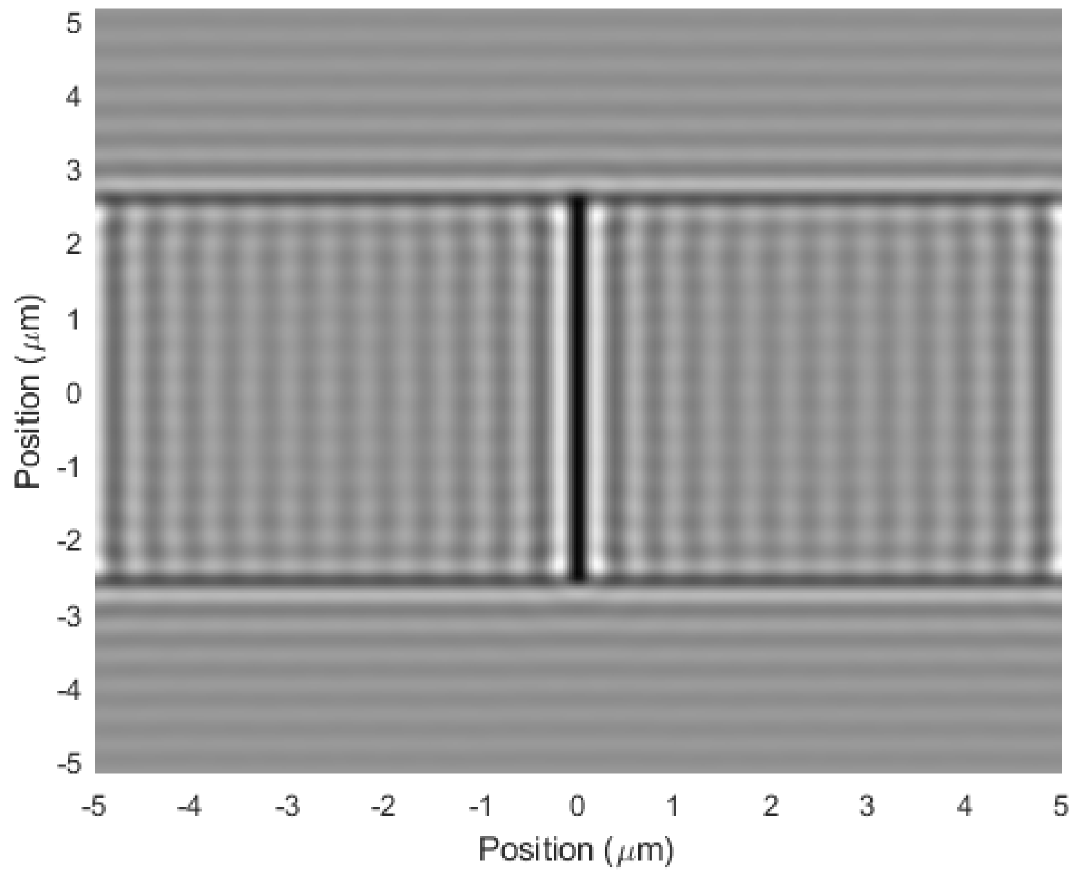

Figure 7.8: Aerial intensity of the projected image of the mask in Figure 7.7. The orthogonal polarizations lead to some polarization diffraction, and shallow horizontal intensity nulls. The spurious nulls are undesired, but have poor contrast, and don't print in resist.

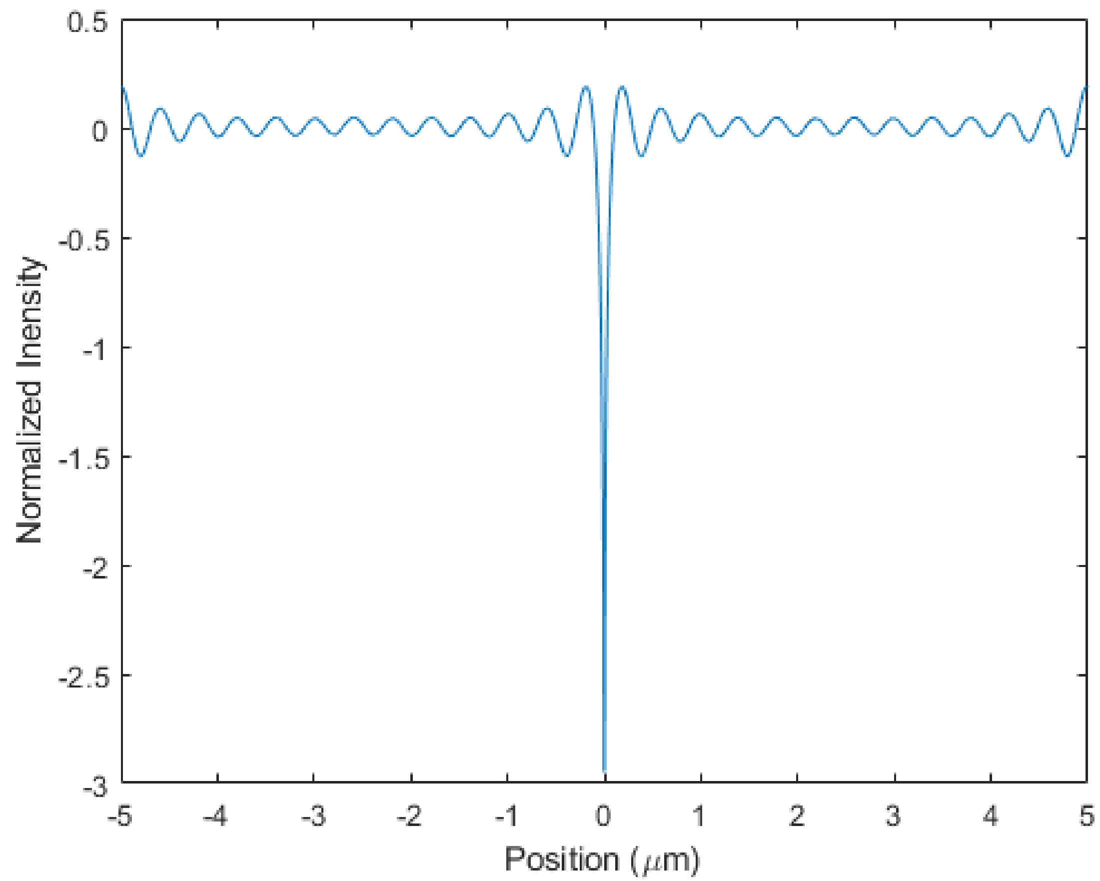

Figure 7.9: Log scale intensity cross section through the center of the image in Figure 7.8. The intensity follows a Sine Integral function as expected.

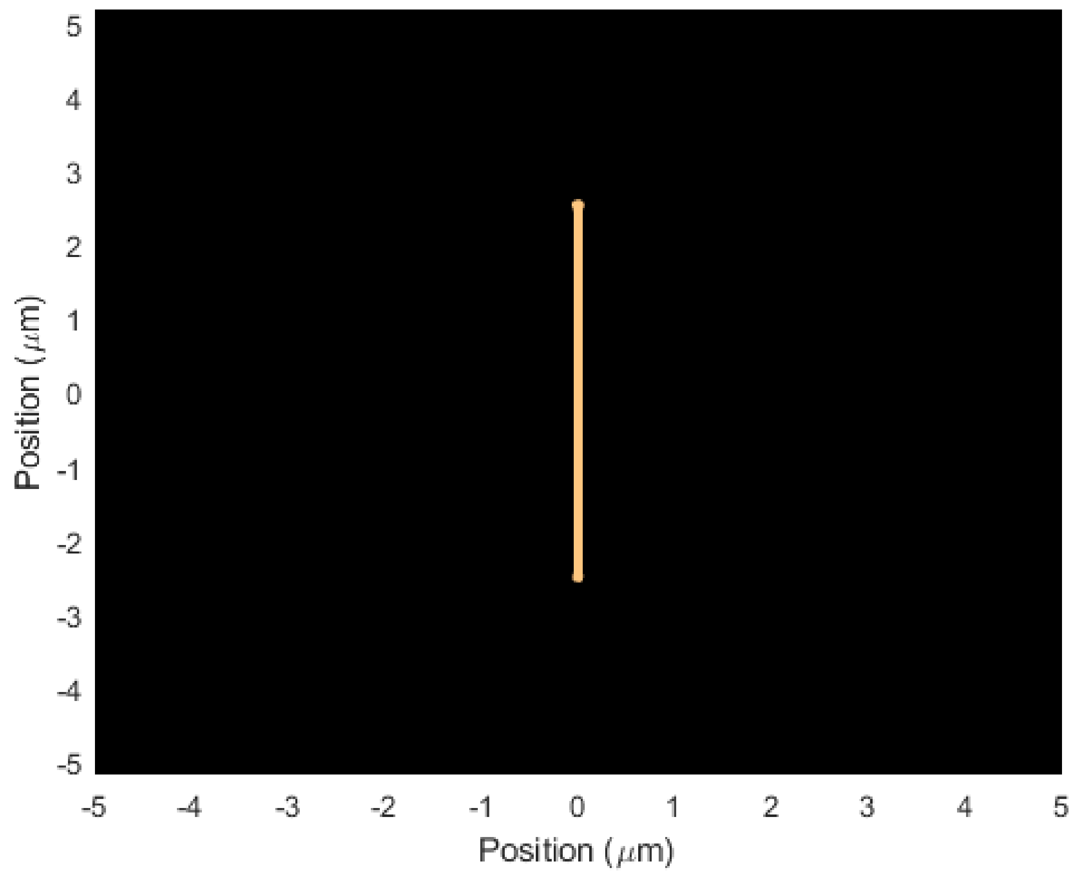

Figure 7.10: Developed resist pattern resulting from a nominal exposure of the intensity distribution shown in Figure 7.8. The spurious nulls are too shallow to print, and the line ends are bulbous.

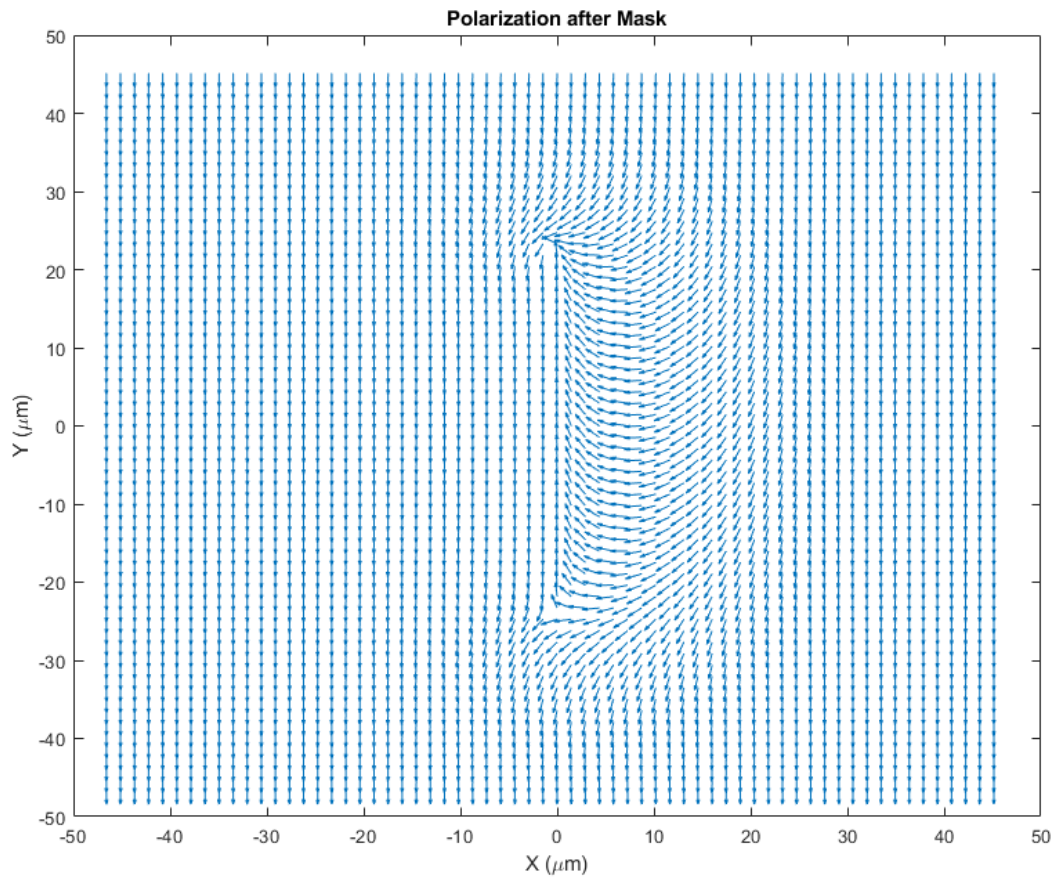

Figure 7.11: Polarization states for a finite line segment. This mask was laid out using polarizations that satisfy the Laplace equation.

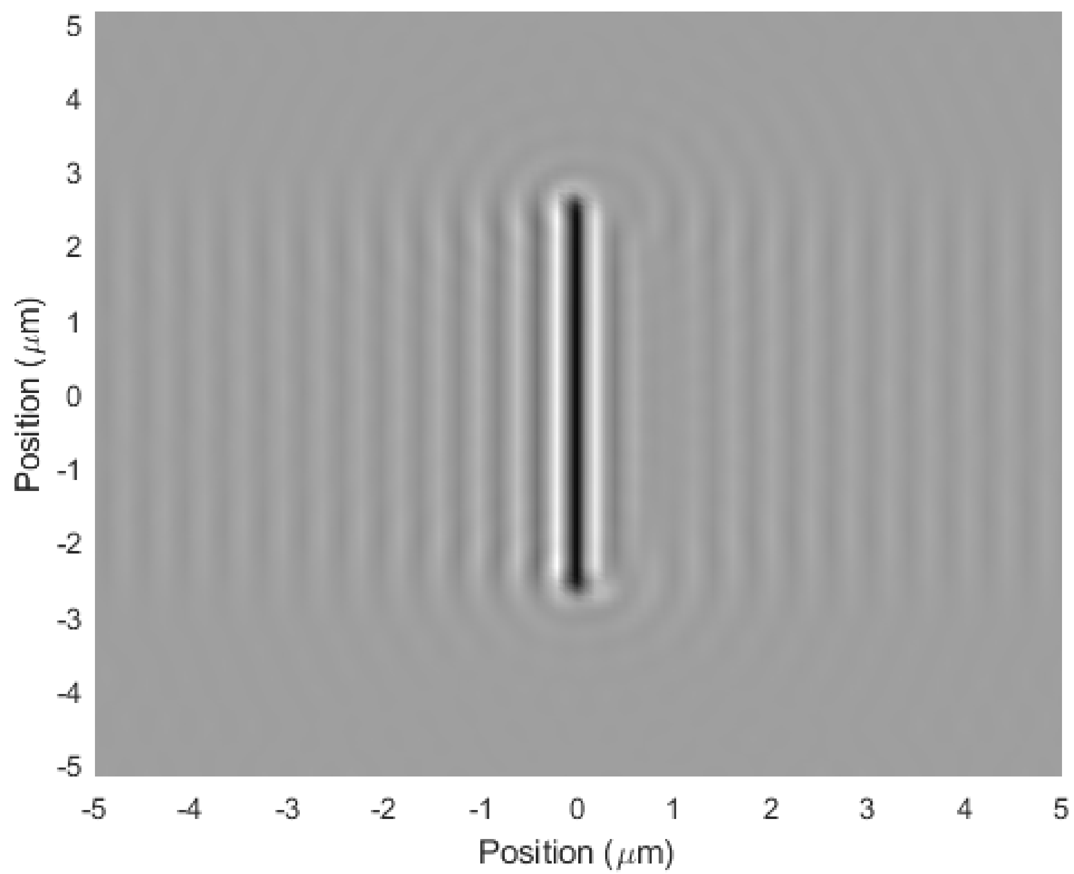

Figure 7.12: Aerial intensity of the projected image of the mask in Figure 7.11. Slowly varying polarization states eliminate spurious intensity nulls.

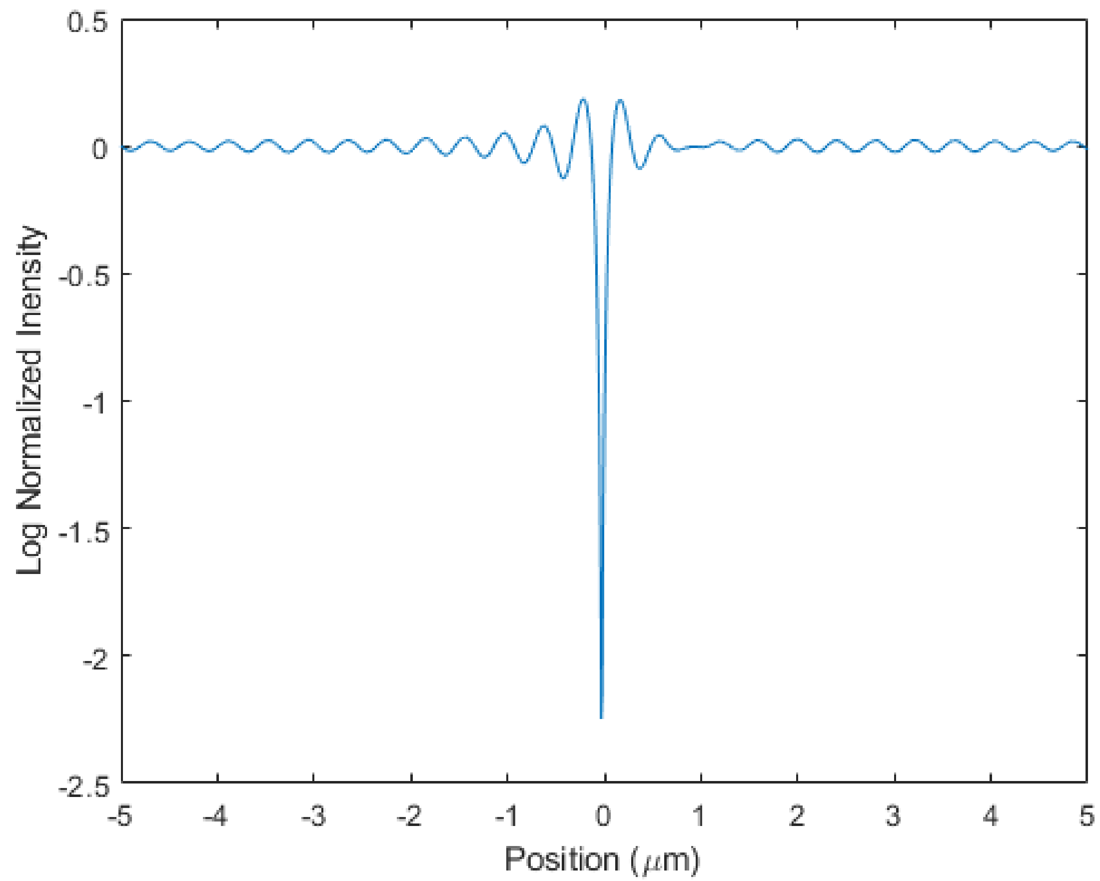

Figure 7.13: Log scale intensity cross section through the center of the image in Figure 7.12. The null depth is reduced compared to the prior mask design.

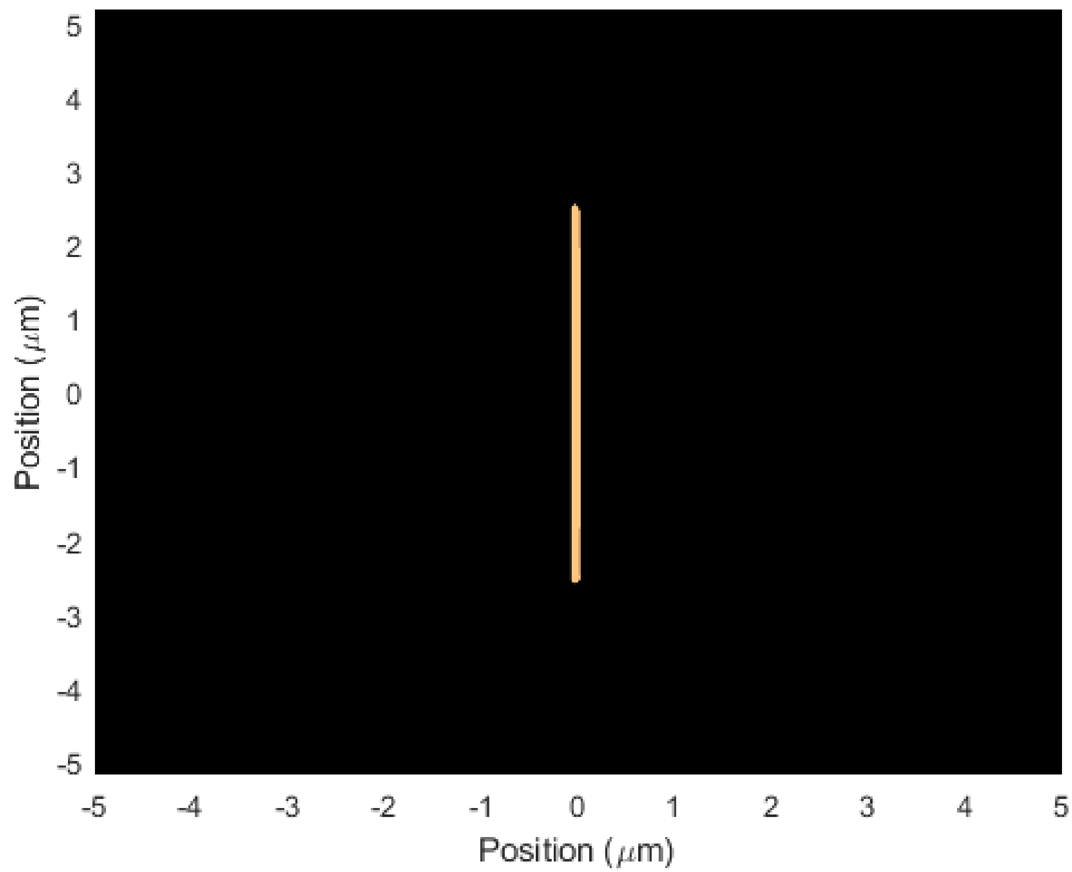

Figure 7.14: Developed resist pattern resulting from a nominal exposure of the intensity distribution shown in Figure 7.12. The line ends are rounded and slightly bent.

$$CD = \frac{\lambda}{4NA}. \tag{7.5}$$

For this simulation, the expected linewidth is 101 nm, and for both masks, the simulated width for a nominal exposure is 111 nm. These simulations have a 10 nm resolution, and this accounts for the difference.

Next, linewidth was simulated as a function of overexposure. Figure 7.15 compares this simulation with the scaling given by the model in Chapter 3. While similar, the scaling with an isolated null likely varies due to iso-dense bias.

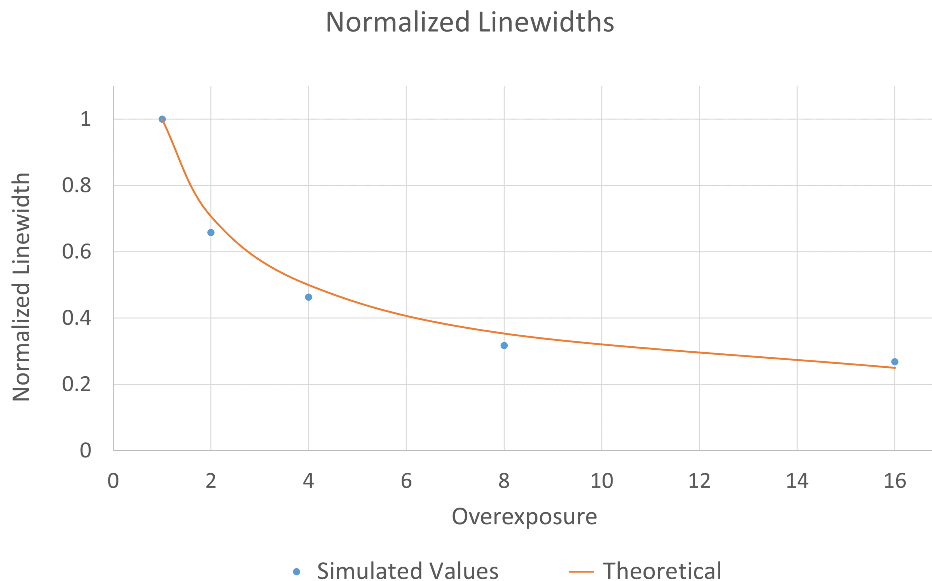

Figure 7.15: Comparison of simulated linewidth scaling for an isolated interference null with the model derived in Chapter 3. The difference is likely due to iso-dense bias.

7.4.2 Circle

To illustrate uniform contrast for all orientations, a mask was designed to produce a circular phase null. For comparison, a standard binary phase mask is also simulated. The binary phase mask is illuminated with vertically polarized light, and is of the same diameter as the P^{3}SM mask. The masks for both binary phase and P^{3}SM are given in figure 7.16. The simulated projected intensities for each mask are given in Figure 7.17, and the developed resist patterns in Figure 7.18. For the binary phase mask, null depth, and therefore contrast, is greatest in regions closely aligned to the incident polarization while contrast is reduced in regions orthogonal to the incident polarization. In contrast, the P^{3}SM mask has uniform null depth and contrast all around. At high NA, binary phase masks can't print in all orientations simultaneously, while P^{3}SM can.

It is interesting to note that in the P^{3}SM image a central null exists due to the geometric phase wrapping around central point in the circular pattern. This could be used to implement vortex vias.

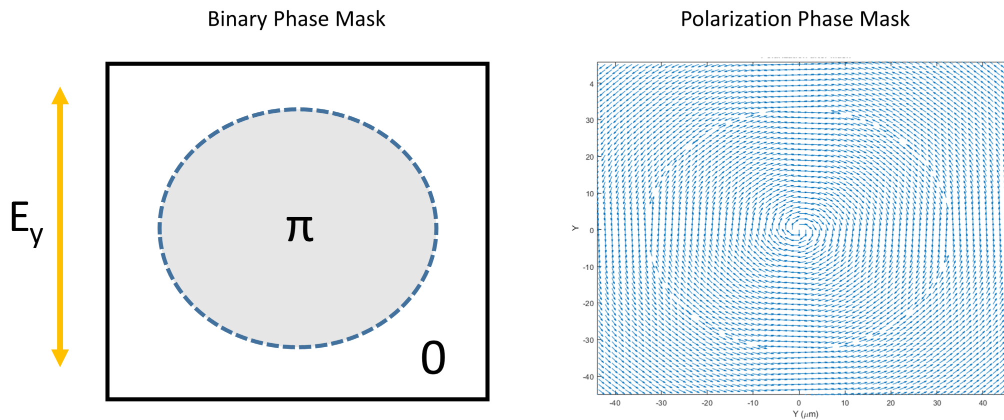

Figure 7.16: Polarization vectors to generate a circular phase null.

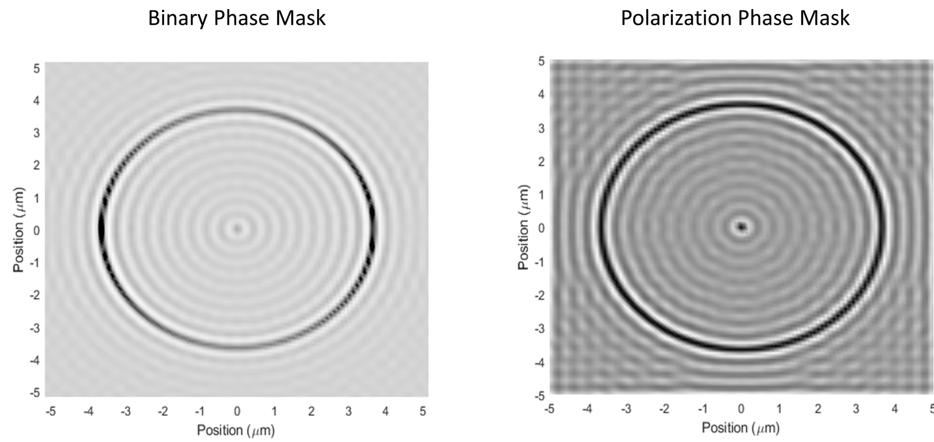

Figure 7.17: Projected intensity distribution of the mask in Figure 7.16.

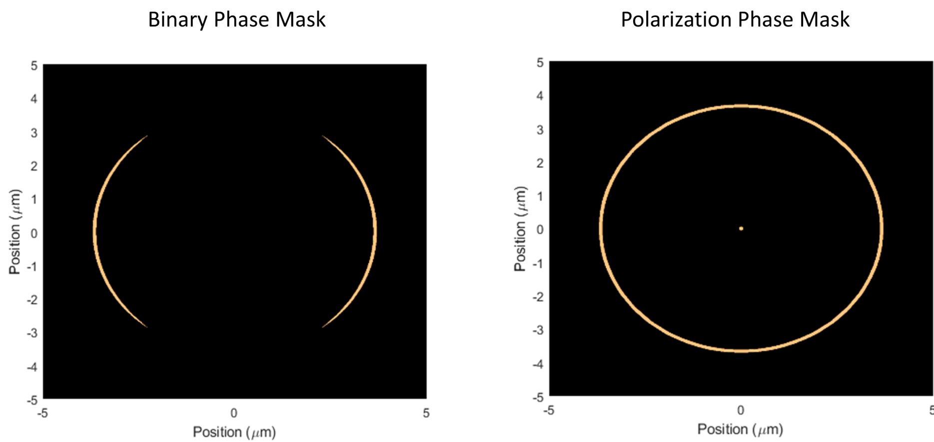

Figure 7.18: Developed photoresist pattern for the projected intensity in Figure 7.17

7.4.3 3 Way intersection

A typical pattern used to demonstrate the limitations of phase masks is a three-way intersection. This problem is solved using the polarization layout given in Figure 7.19. The projected intensity, Figure 7.20 forms the desired intersection. When resist is exposed with this pattern with nominal dose, all three lines meet at the center. With increasing exposure dose however, the lines terminate before intersecting. Some optimization, for example a dark grating, near the intersection may help improve the contrast and print the corner with better fidelity.

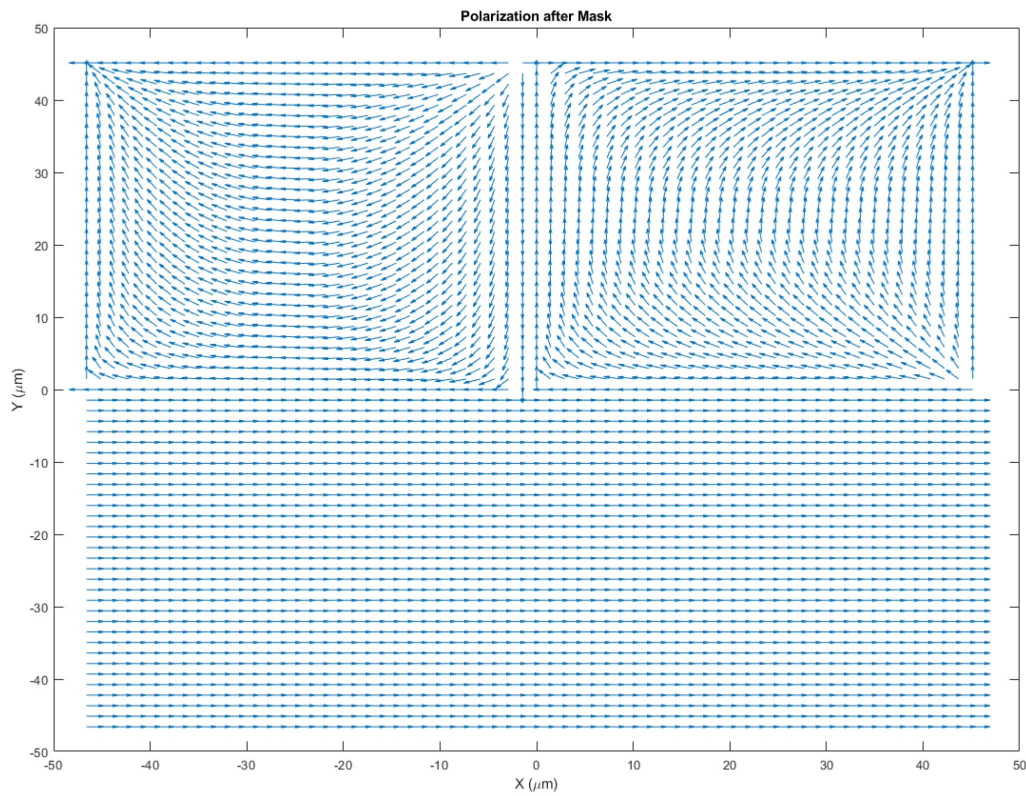

Figure 7.19: Polarization states to generate a three way intersection ("T") pattern.

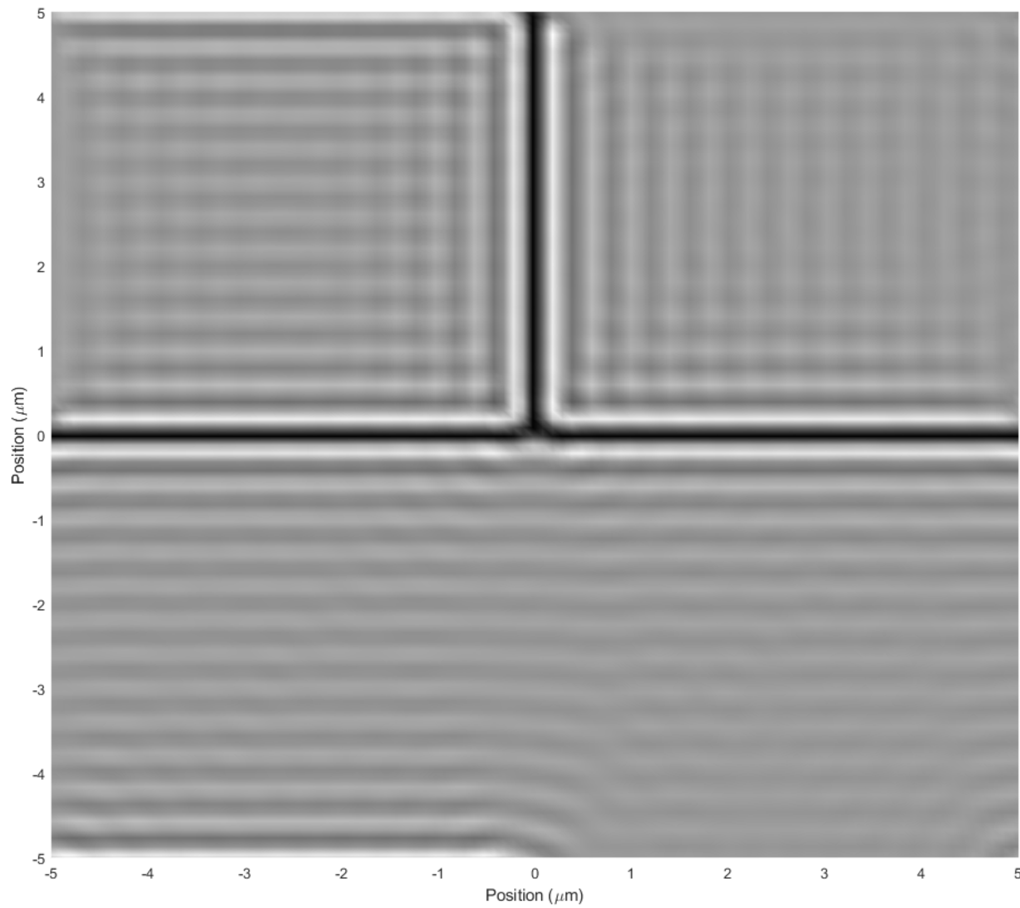

Figure 7.20: Projected intensity of the mask pattern from Figure 7.19

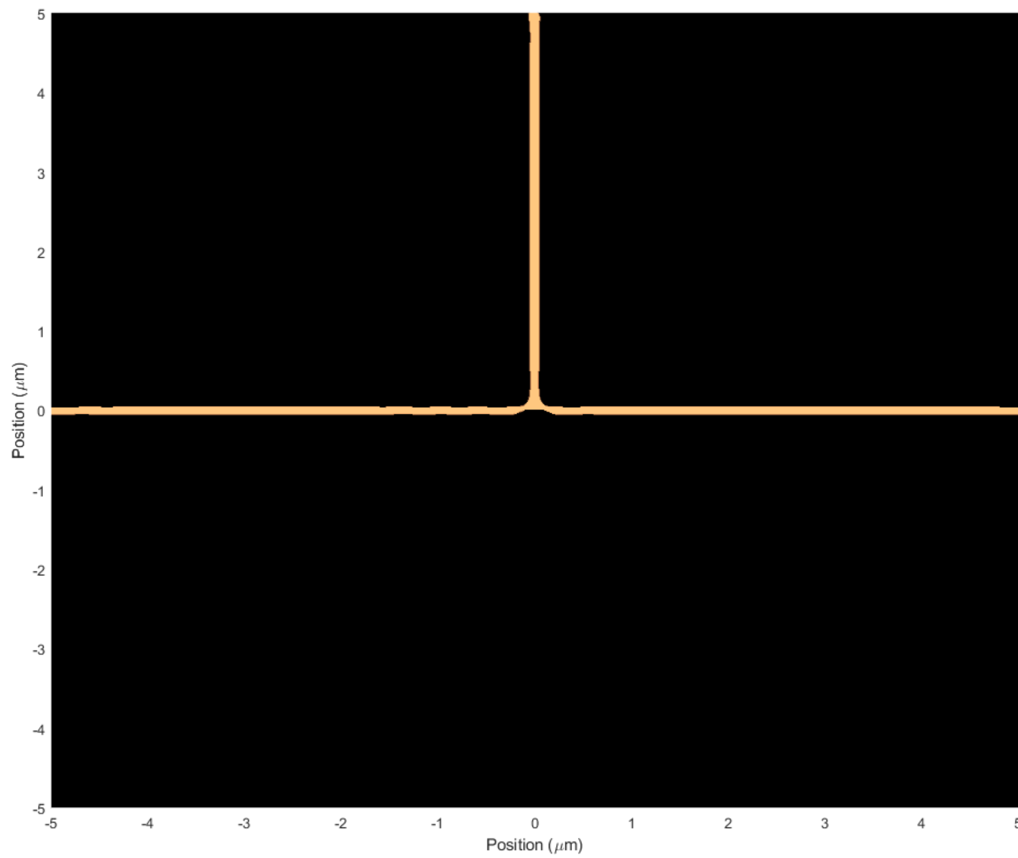

Figure 7.21: Developed resist patterns after exposure with the intensity of Figure 7.20. The lines all connect at the central intersection, although the optical contrast is poor, and with overexposure the lines terminate without connecting.

7.4.4 Alternating Lines

Another pattern that cannot be implemented with traditional phase shifting masks, is that of alternating finite line segments, but is easily implemented using P^{3}SM masks. The waveplate orientations are set around each line segment such that the computed waveplate orientation produces a polarization that "snakes" around each line, shown in Figure 7.22. The projected intensity distribution is given in Figure 7.23, and Figure 7.24 shows that uniform null depth is achieved across all lines. The developed resist pattern is given in Figure 7.25.

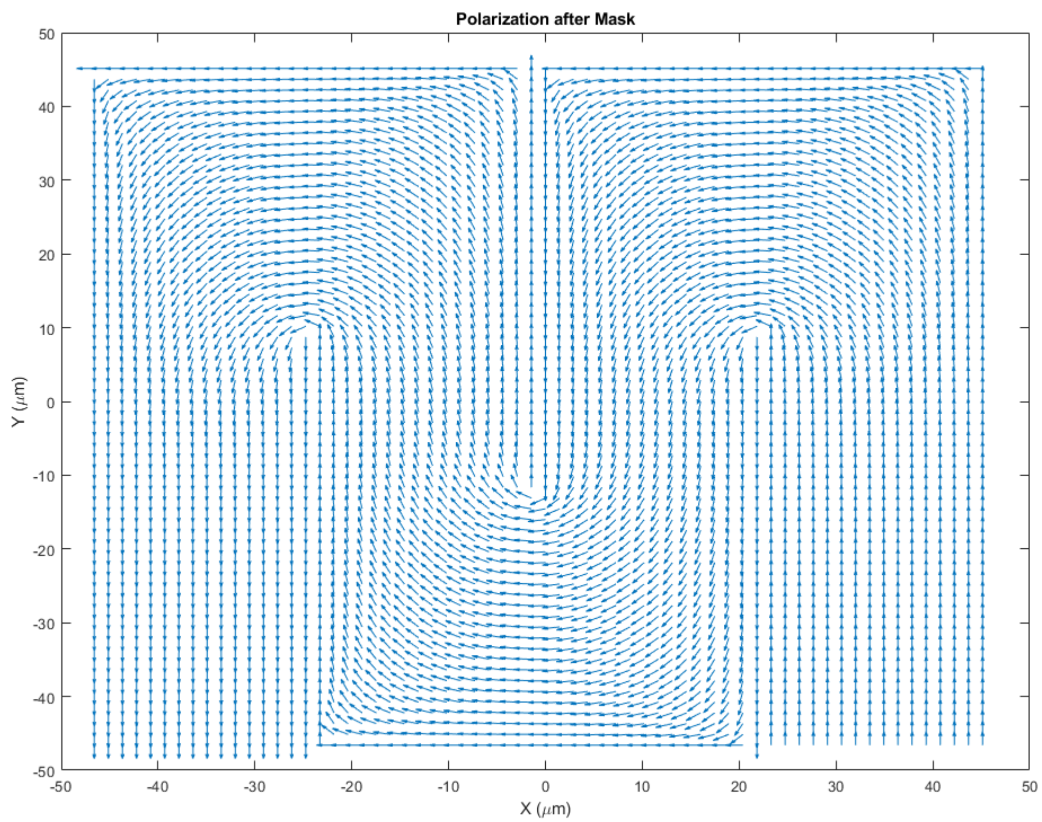

Figure 7.22: Polarization after mask to generate three alternating lines.

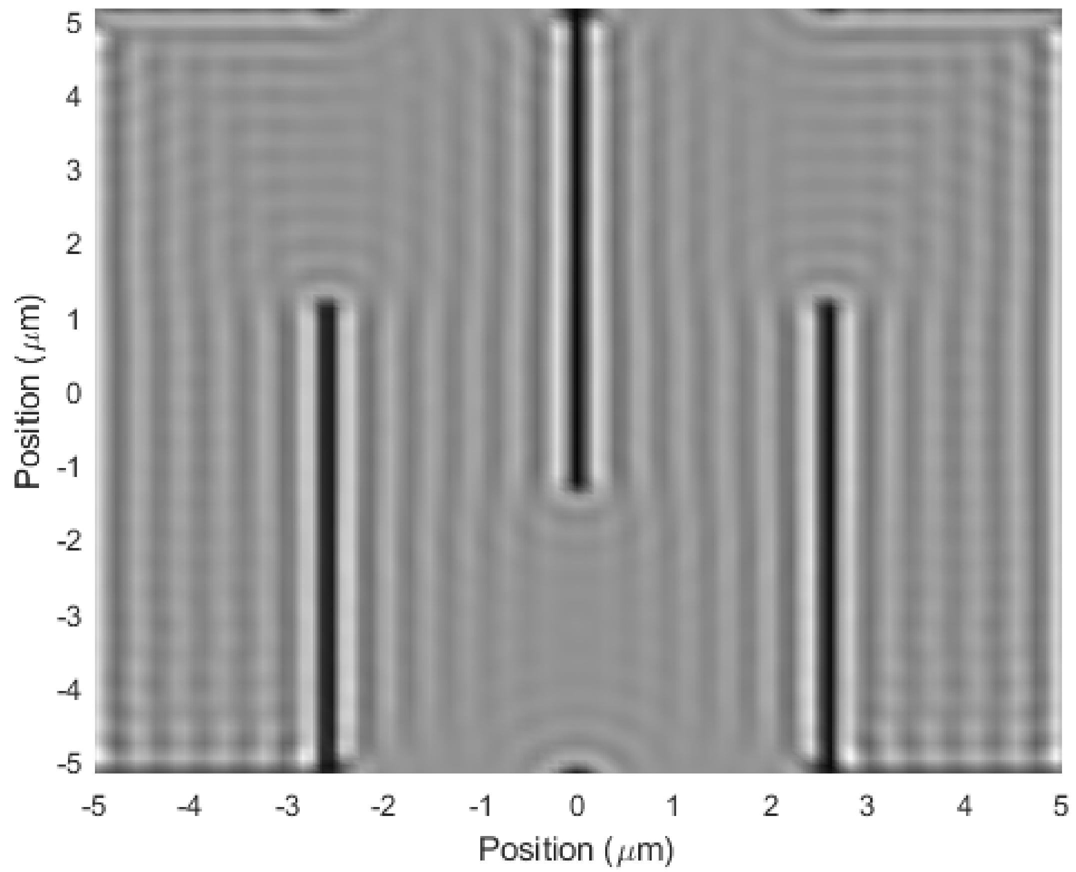

Figure 7.23: Projected intensity distribution for the mask shown in Figure 7.22.

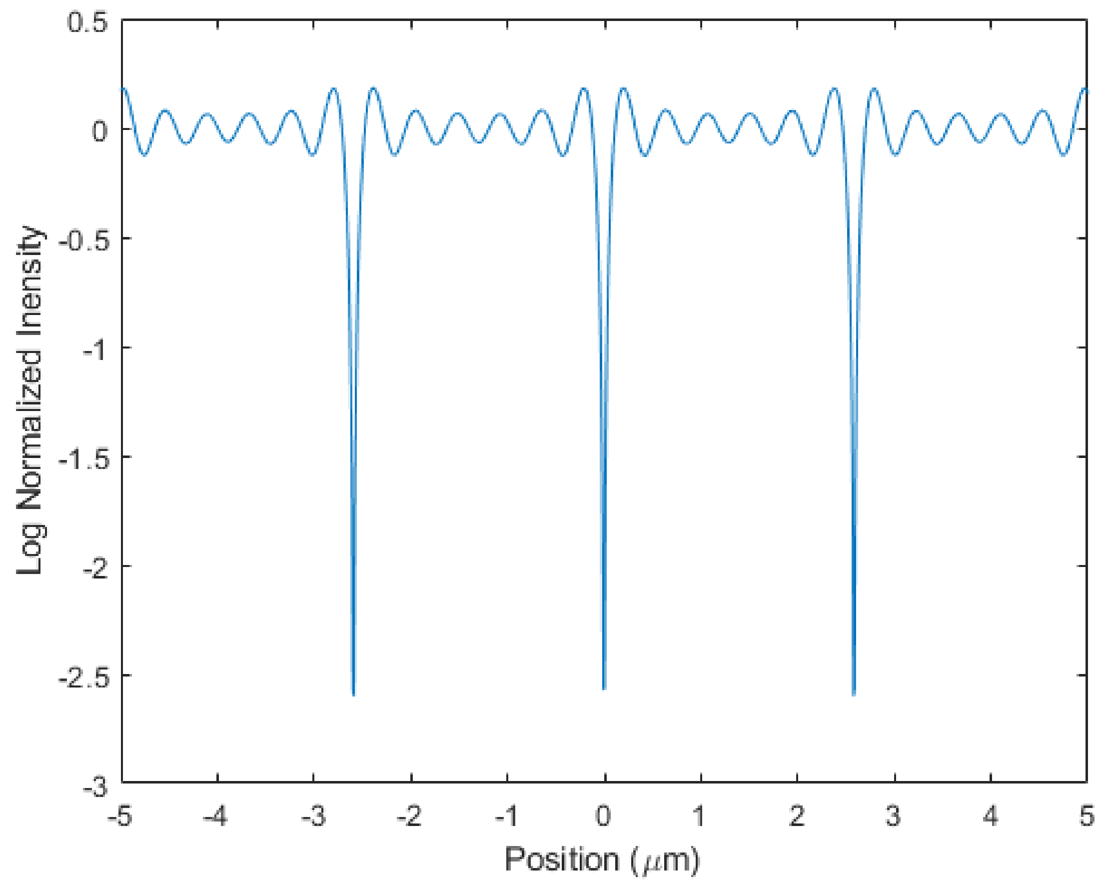

Figure 7.24: Log scale intensity cross section through the center of the aerial intensity shown in Figure 7.23. The nulls each approximately follow the Sine Integral function.

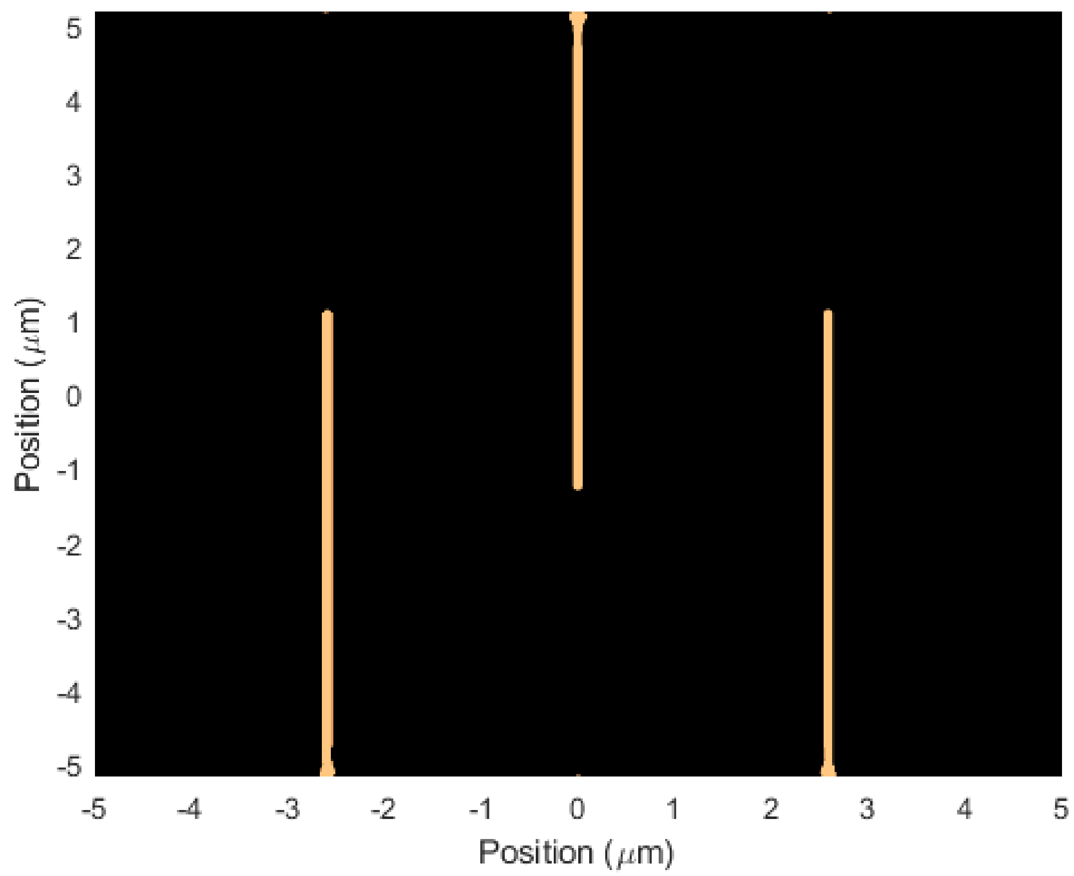

Figure 7.25: Developed resist pattern resulting from a nominal dose delivered by the projected intensity in Figure 7.23

7.4.5 Sub-Contrast Assist Features (SCAFs)

Traditional OPC uses sub-resolution assist features (SRAFs) around isolated features to fix problems associated with iso-dense bias. Similarly, using polarization gratings, low contrast intensity nulls can be formed that don't print in the resist. These are called sub-contrast assist features (SCAFs). By incorporating SCAFs isolated features can be made to behave as periodic structures. An example of a polarization grating SCAF is given in Figure 7.26 along with the projected intensity and an intensity cross section in Figures 7.27 and 7.28. In this example, the central null depth is $10^{-2.5}$ below the average dose, the same as for an isolated line demonstrated above. The periodic nulls in this example only have a contrast of M = 0.25, while the central null has contrast approaching M = 1.

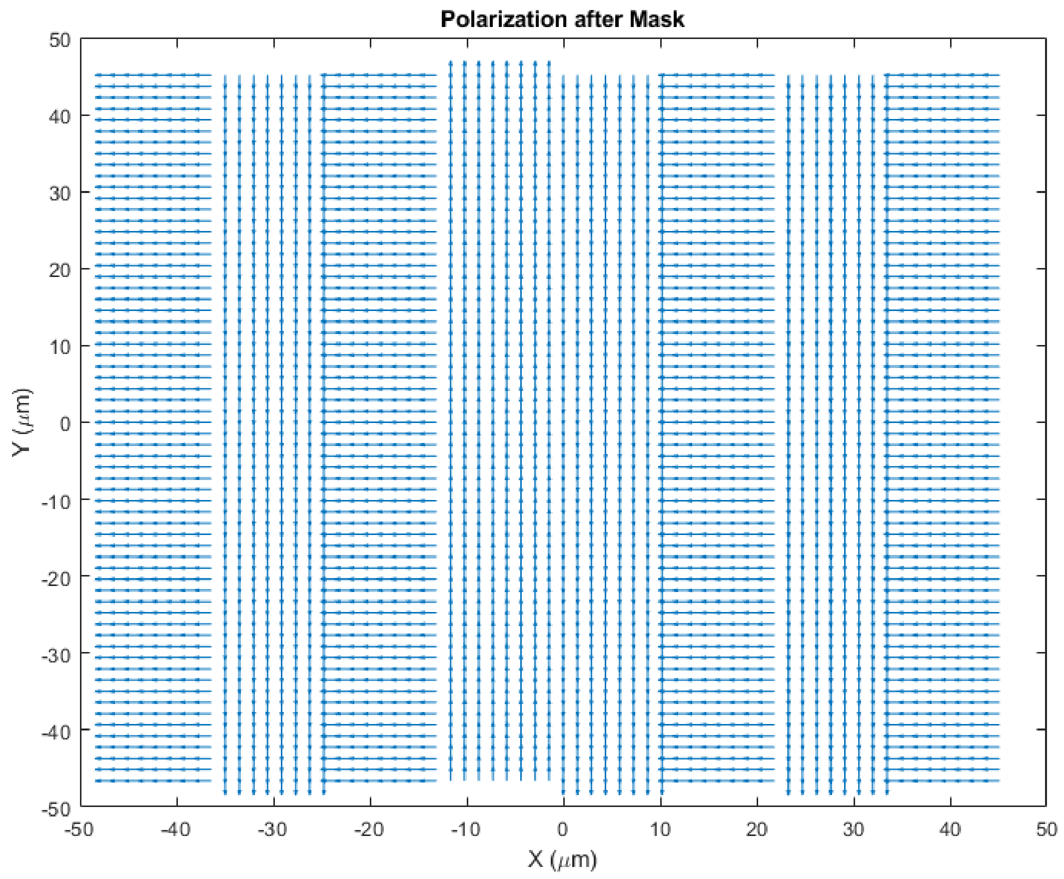

Figure 7.26: Polarization grating used as sub-contrast assist feature (SCAF) around a phase null.

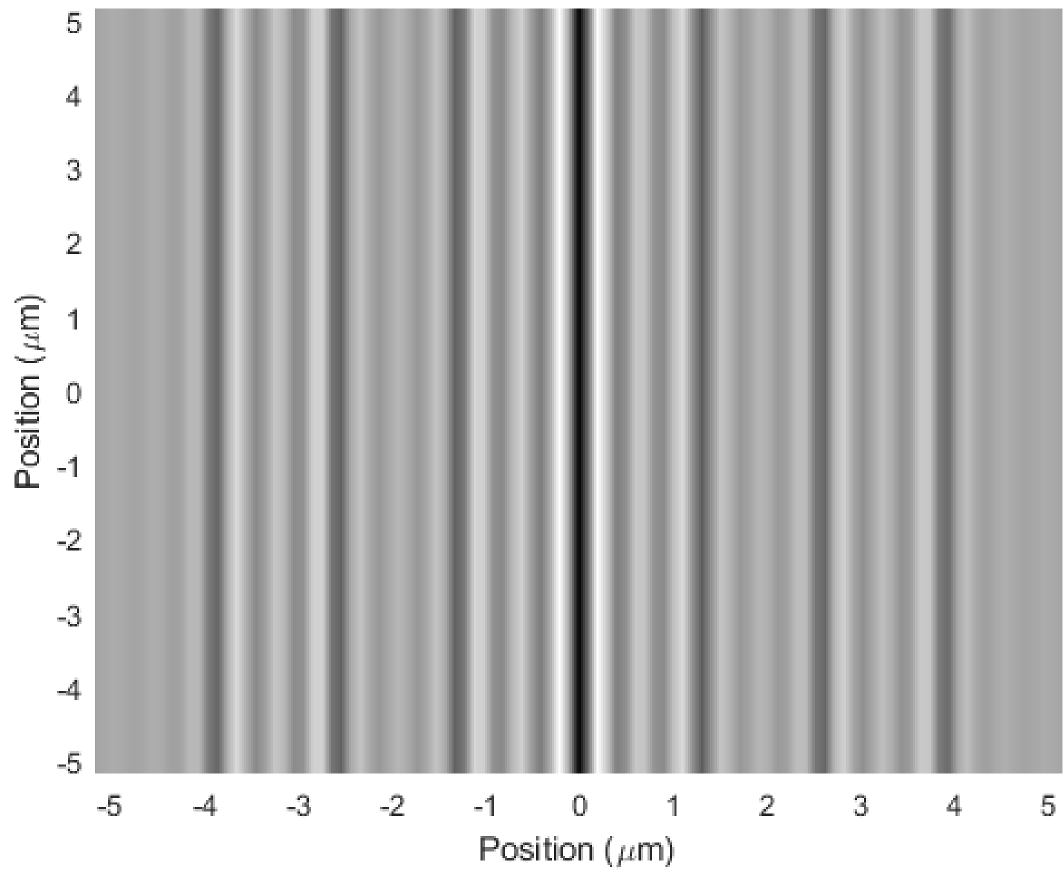

Figure 7.27: Projected intensity of isolated line with SCAF

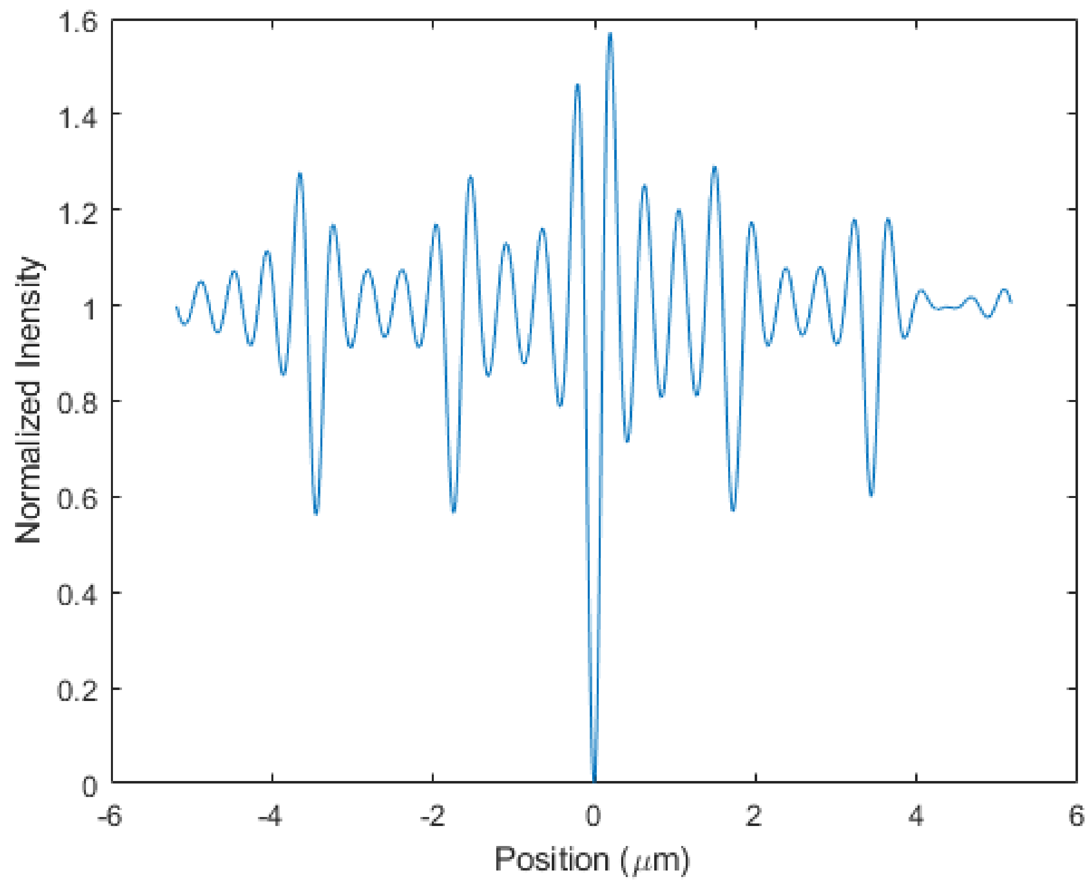

Figure 7.28: Intensity cross-section of Figure 7.27. The SCAF null contrast is only M = 0.25, and does not print in the resist. This allows an isolated line to image as a periodic structure.

7.5 Mask Fab

Pixelated polarization phase shifting masks were fabricated by Beam Engineering Co. They are fabricated such that each pixel has half wave of retardance at a wavelength of 365 nm (i-line). The pixel fast axis are alignments are chosen for illumination with vertical, linear polarization.

7.6 Experimental Results

Mask patterns were projected with the lab-built i-line stepper detailed in Chapter 6. The photoresist used is Ultra-I 123, and the resist processing is the same as in Chapter 5, along with glass ND filter substrates to eliminate the need for a bottom anti reflective coating. After exposing and developing the resist, the wafers were imaged with both an optical microscope and an SEM. A typical grating image is shown in Figure 7.29, with a 10 μm pitch. Line edge variations correspond to inter-pixel artifacts on the mask. These gratings are terminated with an orthogonal polarization, and line ends are demonstrated in Figure 7.30. Again, mask artifacts in the pixel boundaries are apparent in along the resist pattern. Finally, a grating terminated with a dark grating is imaged and demonstrated in Figure 7.31. Grating patterns were overexposed, but doing so caused alternating grating lines to wash out, as shown in Figure 7.32. This appears to be mask related, and further investigation is needed to understand this phenomenon.

Figure 7.29: Optical micrograph of grating lines in resist. Period is approximately 10 μm.

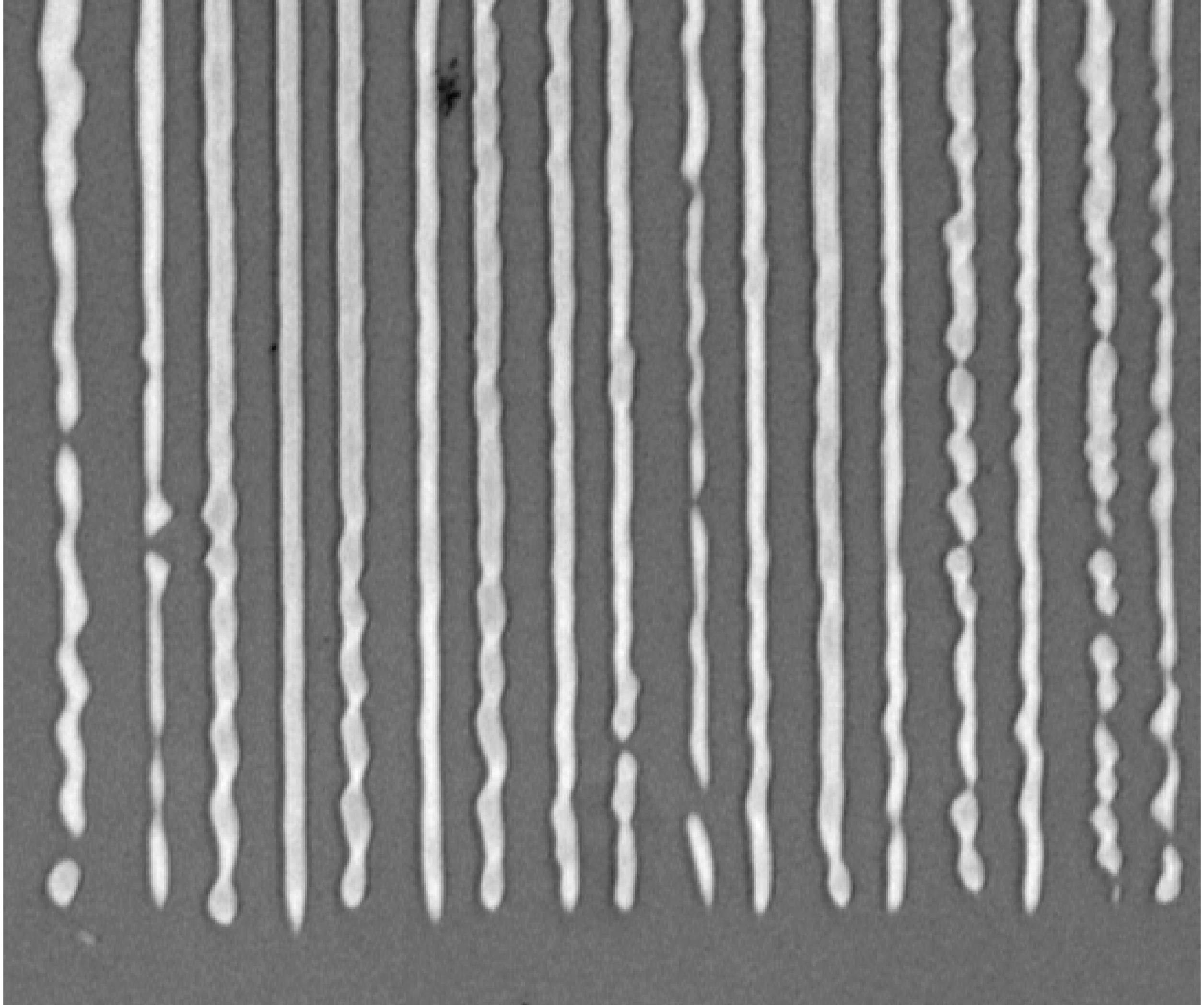

Figure 7.30: Optical micrograph of lines ends in resist. Period is approximately 10 μm.

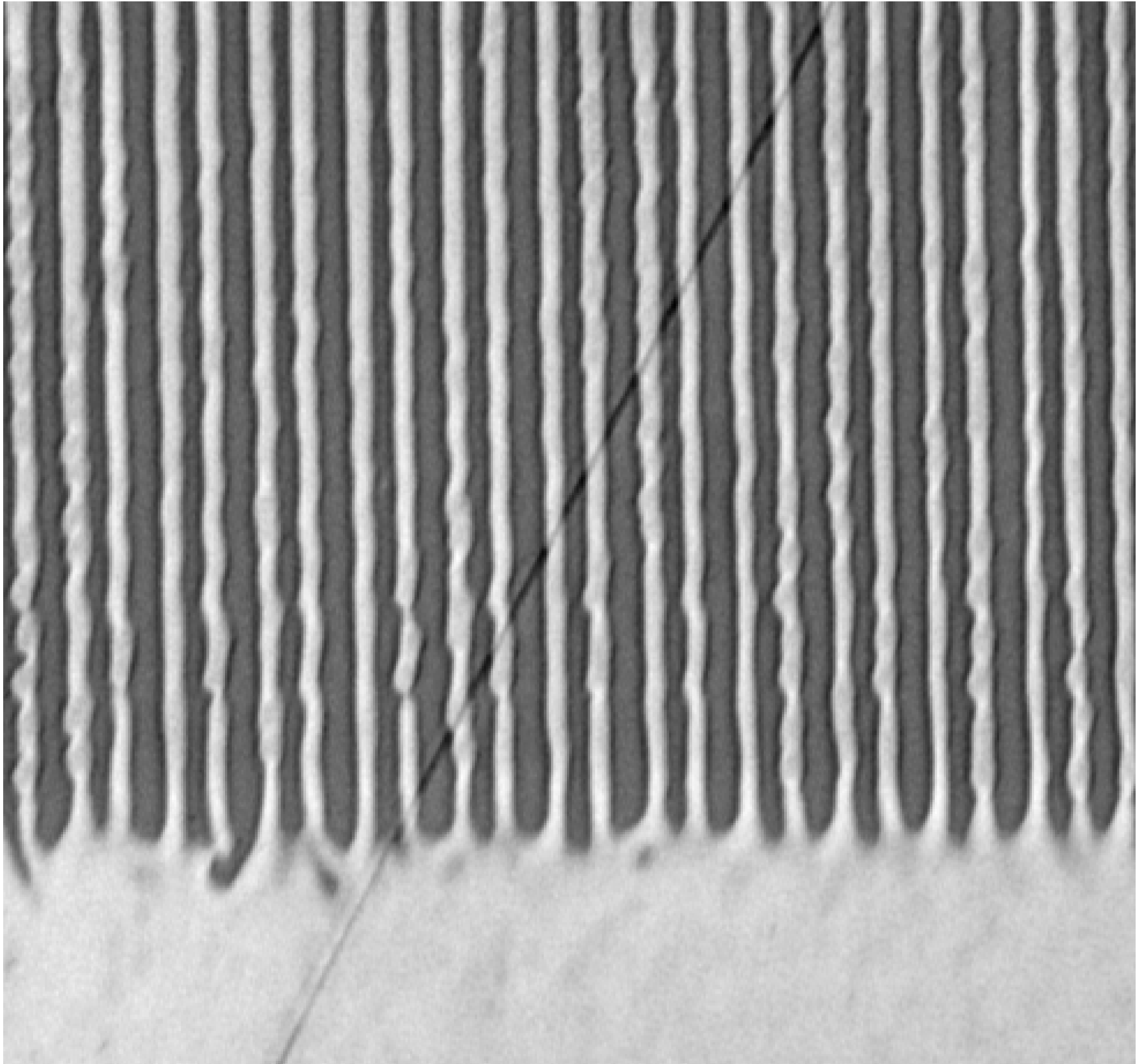

Figure 7.31: Optical micrograph of connected lines (inverted pattern) in resist. Period is approximately 10 μm.

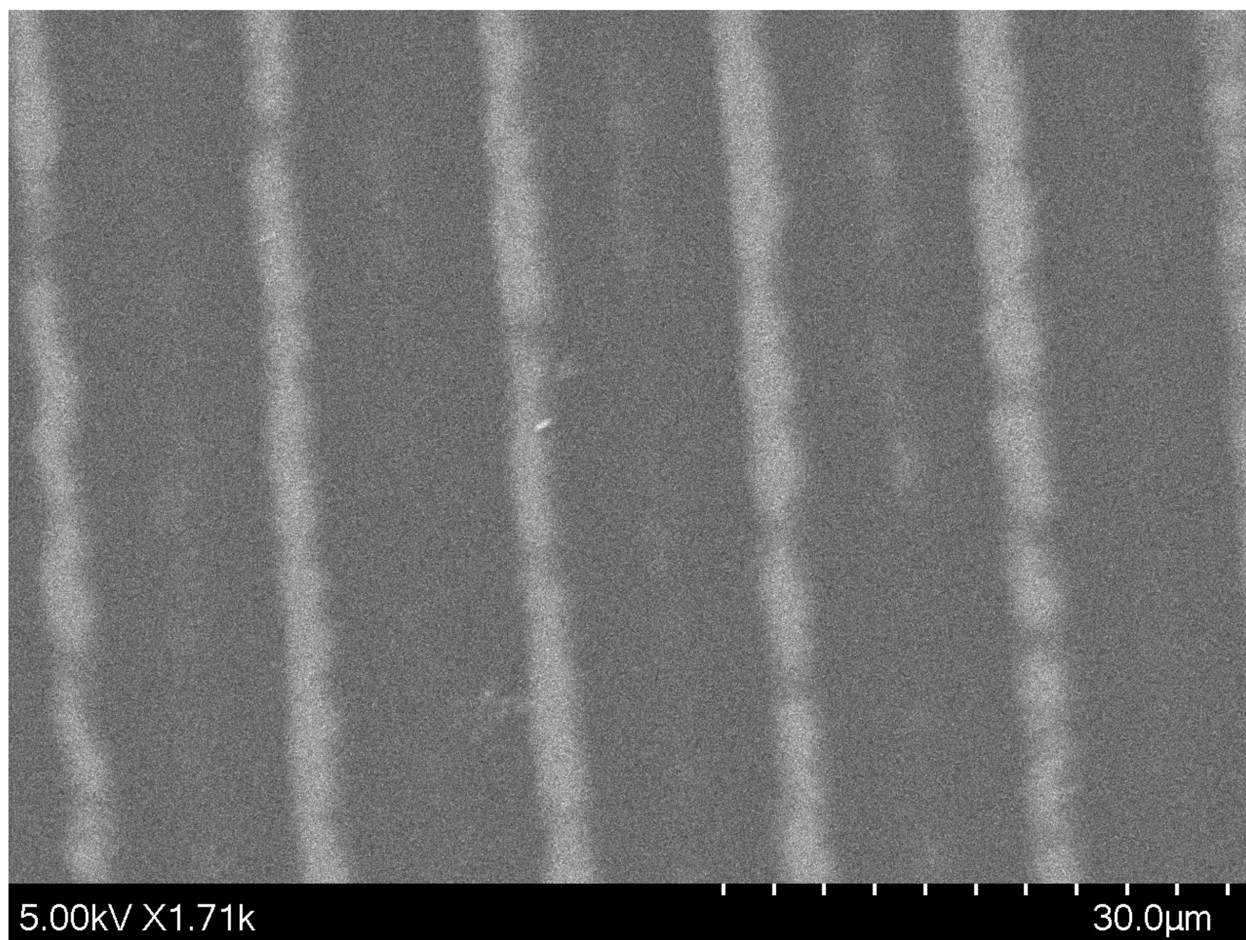

Figure 7.32: SEM image of overexposed grating pattern.

Chapter 8

Future Work and Summary

Pixelated polarization phase masks open many new avenues for exploration. Future work pertaining to liquid crystal P^3SM masks falls into several categories: mask layout and design, mask fabrication, mask limits. Mask layout includes formulating formal design rules for P^3SM masks, exploring mask pattern optimizations, as well as exploring other mask generation algorithms. Mask fabrication includes developing the ability to write LC phase masks in house. With this capability, limitations to mask fabrication can be explored such as minimum LC pixel size, how sharp can inter-pixel boundaries be, and what the LC director does within those boundary regions. Finally, future work involves fabricating new masks to validate designs from Chapter 7 such as alternating lines, three-way intersections, and to demonstrate sub-contrast assist features.

While the masks in Chapter 7 were chromeless, the incorporation of chrome features may enable further pattern optimization. For example, in the three-way intersection, a chrome feature at the intersection of all lines may improve corner fidelity. Also, sub-resolution chrome lines along pixel boundaries may improve polarization purity and pattern fidelity. Finally, sub-resolution assist features may be incorporated in chrome, similar to traditional OPC.

While the LC polymer does not appear to degrade with NUV illumination, it is unlikely to work well with DUV. To extend the work into DUV wavelengths, other mask designs, such as pixelated wiregrid polarizers, can be used. Pixelated wire-grid masks are made from patterned chrome or aluminum layers. By illuminating with unpolarized light, each pixel locally polarizes the illuminating light. Phase steps can be patterned by depositing dielectric on top of the polarizers, or

by etching the spaces between wires. An illustration of such a mask is given in Figure 8.1. Future work includes finalizing polarizer design, fabricating, and demonstrating pixelated polarizer phase masks. In addition to liquid crystal waveplates and wiregrid polarizers, sub-wavelength photonic structures may also be used to implement both phase and polarization control.

Finally, there may be interesting work in exploring orbital angular momentum in lithography. By incorporating spiral polarization patterns, it may be possible to create topological protection for some features.

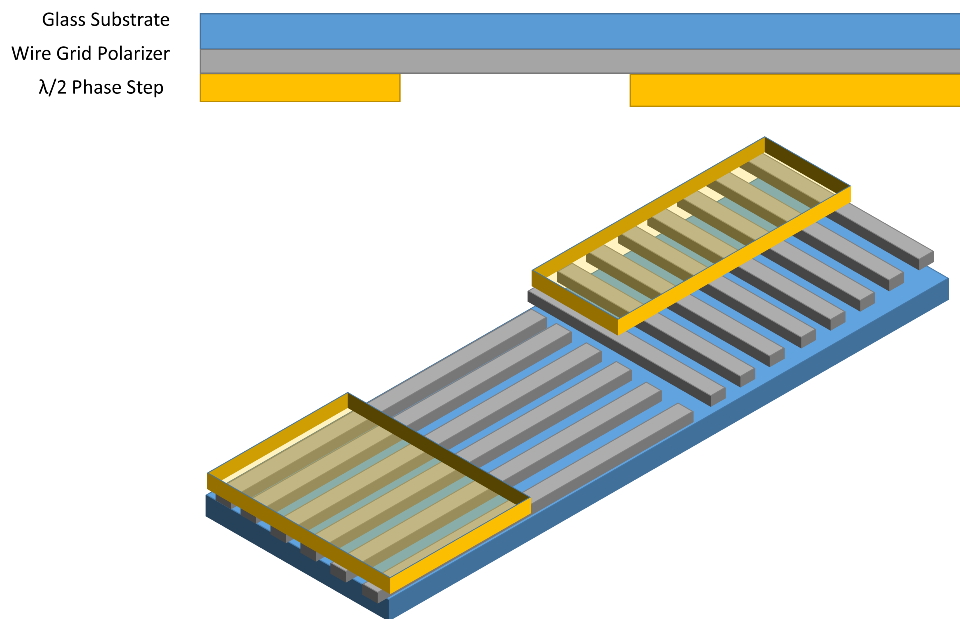

Figure 8.1: Schematic of proposed pixelated wiregrid polarizer phase masks.

Bibliography

[1] Konstantinos Adam and Wilhelm Maurer. Polarization effects in immersion lithography. Journal of Microlithography, Microfabrication and Microsystems, 4(3):1–10, 2005.

[2] Trisha L Andrew, H.-Y. Tsai, and R. Menon. Confining Light to Deep Subwavelength Dimensions to Enable Optical Nanopatterning. Science, 324(5929):917–921, may 2009.

[3] Mark Bohr. A 30 Year Retrospective on Dennard's MOSFET Scaling Paper. IEEE Solid-State Circuits Newsletter, 12(1):11–13, 2009.

[4] Yan Borodovsky, Wen-Hao Cheng, Richard Schenker, and Vivek Singh. Pixelated phase mask as novel lithography RET. Optical Microlithography XXI, 6924(March 2008):69240E, 2008.

[5] Travis Brist, George E. Bailey, Alexander Drozdov, Andres Torres, Andrew Estroff, and Eric Hendrickx. Source polarization and OPC effects on illumination optimization. 25th Annual BACUS Symposium on Photomask Technology, 5992(November 2005):599232, 2005.

[6] S.R.J. Brueck. Optical and Interferometric Lithography - Nanotechnology Enablers. Proceedings of the IEEE, 93(10):1704–1721, oct 2005.

[7] Timothy a. Brunner. Why optical lithography will live forever. Journal of Vacuum Science & Technology B, 21(6):2632–2637, 2003.

[8] Stephen A Campbell. The Science and Engineering of Microelectronic Fabrication . Oxford university press Oxford, second edition, 2001.

[9] Arthur E. Chiou, Wen Wang, Greg J. Sonek, John Hong, and M. W. Berns. Interferometric optical tweezers. Optics Communications, 133(1-6):7–10, 1997.

[10] Robert H. Dennard, Fritz H. Gaensslen, Y. U. Hwa-Nien, V. Leo Rideout, Ernest Bassous, and Andre R. Leblanc. Design of Ion-Implanted MOSFETs with Very Small Physical Dimensions. Proceedings of the IEEE, 87(4):668–678, 1999.

[11] John T. Fourkas, Nikolaos Liaros, Zuleykhan Tomova, Sandra A. Gutierrez Razo, Samuel R. Cohen, Steven M. Wolf, Matthew Thum, Daniel E. Falvey, Hannah M. Ogden, Amy S. Mullin, Adam Pranda, Gottlieb S. Oehrlein, and John S. Petersen. The state of the art in multicolor visible photolithography. Emerging Patterning Technologies 2018, 1058407(March):6, 2018.

[12] M. Fritze, B. M. Tyrrell, D. K. Astolfi, R. D. Lambert, D.-R. W. Yost, a. R. Forte, S. G. Cann, and B. D. Wheeler. Subwavelength optical lithography with phase-shift photomasks. Lincoln Laboratory Journal, 14(2):237–250, 2003.

[13] Joseph W. Goodman. Statistical Optics. Wiley, Hoboken, second edition, 2015.

[14] M. G.L. Gustafsson. Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy. Journal of Microscopy, 198(2):82–87, 2000.

[15] Steven G. Hansen. Source mask polarization optimization. Journal of Micro/Nanolithography, MEMS, and MOEMS, 10(3):033003, 2011.

[16] Benjamin Harke, Jan Keller, Chaitanya K. Ullal, Volker Westphal, Andreas Schönle, and Stefan W. Hell. Resolution scaling in STED microscopy. Optics Express, 16(6):4154, 2008.

[17] Andrew J. Hazelton, Shinji Wakamoto, Shigeru Hirukawa, Martin McCallum, and Nobutaka Magome. Double-patterning requirements for optical lithography and prospects for optical extension without double patterning. Journal of Micro/Nanolithography, MEMS, and MOEMS, 8(1):011003, 2009.

[18] Stefan W. Hell and Jan Wichmann. Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy. Optics Letters, 19(11):780, 1994.

[19] W. Hinsberg. Deep-ultraviolet interferometric lithography as a tool for assessment of chemically amplified photoresist performance. Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures, 16(6):3689, 1998.

[20] Kotaro Kawai, Tomoyuki Sasaki, Kohei Noda, Nobuhiro Kawatsuki, and Hiroshi Ono. Simple fabrication of liquid crystalline grating cells with homogeneous and twisted nematic structures and effects of orientational relaxation on diffraction properties. Applied Optics, 53(17):3679, 2014.

[21] P. T. Konkola, C. G. Chen, R. K. Heilmann, and M. L. Schattenburg. Beam steering system and spatial filtering applied to interference lithography. Journal of Vacuum Science & Technology B, 18(2000):3282–3286, 2000.

[22] David Levenson, John S. Petersen, David J. Gerold, and Chris a. Mack. Phase phirst! An improved strong-PSM paradigm. 20th Annual BACUS Symposium on Photomask Technology, 4186(1):395–404, 2001.

[23] Marc D Levenson, Grace Dai, Takeaki Joe, M D Levenson Consulting, and Saratoga Ca. The Vortex Mask : Making 80nm contacts with a twist ! Exposure, 2017.

[24] Harry J. Levinson. Principles of Lithography. SPIE, 1000 20th Street, Bellingham, WA 98227-0010 USA, nov 2010.

[25] Linjie Li, Rafael R Gattass, Erez Gershgoren, Hana Hwang, and John T Fourkas. Achieving /20 Resolution by One-Color Initiation and Deactivation of Polymerization. Science, 324(5929):910–913, may 2009.

[26] Burn Jeng Lin. Optical lithography: here is why. SPIE, Bellingham, 2010.

[27] Xu Ma, Jie Gao, Chunying Han, Yanqiu Li, Lisong Dong, and Lihui Liu. Efficient source polarization optimization for robust optical lithography. Optical Microlithography XXVII, 9052(March 2014):90520T, 2014.

[28] CA Mack. Fundamental issues in phase-shifting mask technology. In OCG Microelectronics Conference, pages 23–25, San Jose, 1993. KTI Microlithography Seminar.

[29] Chris Mack. The Multiple Lives of Moore's Law, 2015.

[30] Chris a. Mack. Development of Positive Photoresists. Journal of The Electrochemical Society, 134(1):148, 1987.

[31] Chris A Mack. Using the Normalized Image Log-Slope. Microlithography World, 23(Feb), 2001.

[32] Chris A Mack. Using the Normalized Image Log-Slope, Part 2. Microlithography World, 26(Spring), 2001.

[33] Chris A Mack. Field Guide to Optical Lithography. SPIE, 1 edition, jan 2006.

[34] Chris A. Mack. Fundamental Principles of Optical Lithography. John Wiley & Sons, Inc., West Sussex, 2007.

[35] Robert R. McLeod and Kelvin H. Wagner. Vector Fourier optics of anisotropic materials. Advances in Optics and Photonics, 6(4):368, dec 2014.

[36] Prateek Mehrotra, Chris A. Mack, and Richard J. Blaikie. A detailed study of resonance-assisted evanescent interference lithography to create high aspect ratio, super-resolved structures. Optics Express, 21(11):13710, 2013.

[37] David B Miller, Darren L Forman, Adam M Jones, Robert R Mcleod, David B Miller, Darren L Forman, Adam M Jones, and Robert R Mcleod. Super-resolved critical dimensions in far-field I-line photolithography. Journal of Micro/Nanolithography, MEMS, and MOEMS, 18(1):013505–1, 013505–10, 2019.

[38] Jun Hyuk Moon, Jamie Ford, and Shu Yang. Review Fabricating three-dimensional polymeric photonic structures by multi-beam interference lithography. Polymers for Advanced Technologies, 17:83–93, 2006.

[39] G M Moore. Moore's Law ,Electronics. Electronics, 38(8):114, 1965.

[40] Sarohan Park, Inwhan Lee, Sunyoung Koo, Junghyung Lee, and Chang-moon Lim. Extension of practical k1 limit in EUV lithography. In Eric M. Panning and Kenneth A. Goldberg, editors, Extreme Ultraviolet (EUV) Lithography VII, number 9776, page 97761Q, mar 2016.

[41] Rüdiger Paschotta. Noise in Laser Technology - Part 3: Beam Pointing Fluctuations. Optik & Photonik, 5(1):55–57, 2010.

[42] R. Fabian Pease and Stephen Y. Chou. Lithography and other patterning techniques for future electronics. Proceedings of the IEEE, 96(2):248–270, 2008.

[43] Allen Pu and Demetri Psaltis. High-density recording in photopolymer-based holographic three-dimensional disks. Applied Optics, 35(14):2389–2398, 1996.

[44] Eva Rittweger, Kyu Young Han, Scott E. Irvine, Christian Eggeling, and Stefan W. Hell. STED microscopy reveals crystal colour centres with nanometric resolution. Nature Photonics, 3(3):144–147, 2009.

[45] Ainara Rodriguez, Mikel Echeverría, Miguel Ellman, Noemi Perez, Yuri K. Verevkin, Changsi S. Peng, Thierry Berthou, Zuobin Wang, Isabel Ayerdi, Joan Savall, and Santiago M. Olaizola. Laser interference lithography for nanoscale structuring of materials: From laboratory to industry. Microelectronic Engineering, 86(4-6):937–940, 2009.

[46] Bahaa E. A. Saleh and Malvin Carl Teich. Fundamentals of Photonics. Wiley Series in Pure and Applied Optics. John Wiley & Sons, Inc., New York, USA, aug 1991.

[47] Dong-Yu Kim Sangjin Kwon, Pilgyu Kim, Sungho Jeong, Wonseok Chang, Cjaemin Chun. Fabrication of Nano Dot and Line Arrays Using NSOM Lithography, 2005.

[48] T. F. Scott, B. A. Kowalski, A. C. Sullivan, C. N. Bowman, and R. R. McLeod. Two-Color Single-Photon Photoinitiation and Photoinhibition for Subdiffraction Photolithography. Science, 324(5929):913–917, 2009.

[49] Bruce W. Smith, Lena V. Zavyalova, and Andrew Estroff. Benefiting from polarization effects on high-NA imaging. Optical Microlithography XVII, 5377(May 2004):68, 2004.

[50] W.H. Steel. Interferometry. Cambridge University Press, London, first edition, 1967.

[51] Karl A Stetson and Robert L Powell. Hologram Interferometry. Journal of the Optical Society of America, 56(9):1161–1166, 1966.

[52] Donald M. Tennant. Nanotechnology. Springer New York, New York, NY, 1999.

[53] Kenny K. Toh, Giang T. Dao, Rajeev R. Singh, and Henry T. Gaw. Chromeless phase-shifted masks: a new approach to phase-shifting masks. In James N. Wiley, editor, 10th Annual Symp on Microlithography, volume 1496, pages 27–53, mar 1991.

[54] P. Török and P.R.T. Munro. The use of Gauss-Laguerre vector beams in STED microscopy. Optics Express, 12(15):3605, jul 2004.

[55] Andreas J. Wolf, Hubert Hauser, Volker Kübler, Christian Walk, Oliver Höhn, and Benedikt Bläsi. Origination of nano- and microstructures on large areas by interference lithography. Microelectronic Engineering, 98:293–296, 2012.

[56] H Wolferen, Leon Abelmann, and Henk van Wolferen. Laser interference lithography. Lithography: Principles, Processes and Materials, pages 133–148, 2011.

[57] WwwMicroChemicals.com. MicroChemicals ®-Basics of Microstructuring: Resist Rehydration.

[58] Zhihua Xie, Weixing Yu, Taisheng Wang, Hongxin Zhang, Yongqi Fu, Hua Liu, Fengyou Li, Zhenwu Lu, and Qiang Sun. Plasmonic Nanolithography: A Review. Plasmonics, 6(3):565–580, 2011.

[59] Saleem H. Zaidi and Steven R. J. Brueck. Nanoscale Fabrication by Interferometric Lithography. In Ichirou Yamaguchi, editor, Optical Engineering for Sensing and Nanotechnology (ICOSN'99), number May 1999, pages 340–343, may 1999.

Appendix A

Appendix

A.1 Photoresist Contrast Curves

The use of a simplified piecewise photoresist transfer function is justified via a comparison of the simplification against an analytic model. In Figure A.1, both the piecewise model used in this paper as well as an analytic model [30] are overplotted. The photoresist contrast, γ, is defined as the slope of this curve near D_2. As can be seen in the plot, the slope of both curves, and therefore contrasts, are similar near D_2. The two primary differences are that resist thickness begins to decrease slightly for doses less than D_1, and that resist thickness at the knee of the analytic curve is less than the approximation.

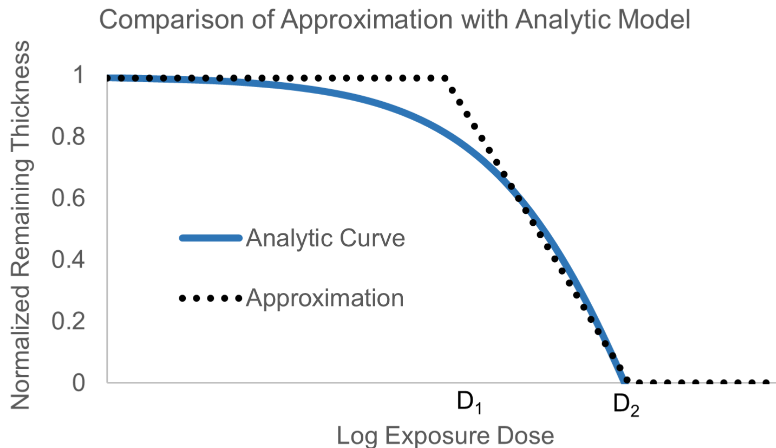

Figure A.1: An analytic model for resist thickness is overplotted on the piecewise approximation used in the main text. As shown, both models yield similar contrast. A key difference is that in the analytic model, the resist thickness begins to decrease at doses below D_1.

A.2 Methods

Two commercial i-line photoresists are used: Ultra-i 123 1.0 (Dow), and AZ 4210 (Merck). Both are diluted with with propylene glycol monomethyl ether acetate (PGMEA) (Sigma-Aldrich). Substrates used are Schott NG4 (1.9 mm thick) absorptive filters, primed with hexamethyldisilazane (HMDS) (Ultra Pure Solutions, Inc). The developer used is Microposit MF-319 (Shipley).

A.2.1 Substrate and Resist Preparation

Schott substrates are prepared by first cleaning the surface with Nanostrip, acetone, methanol, and DI water. Next the substrates are baked on a hotplate at 120°C for 5 minutes. An adhesion promoter, HMDS, is spun on at 6000 RPM after a 20 second dwell with a puddle. Both photoresists (Ultra-i and AZ 4210) are diluted with PGMEA and spin cast onto a prepared substrate. The spin coating parameters are adjusted to give final film thicknesses (after soft bake) of approximately 150 nm. Dilution ratios, spin RPM, spin times, for each resist are given in Table A.1. Spin coating is then followed by a pre-exposure bake (soft bake) on a contact hot plate. Soft bake was optimized for each resist to maximize contrast and minimize dark erosion, using a fixed photoresist development time of 30 seconds. Details of the soft bake optimization are given in Appendix A.3. Table A.2 lists the optimized soft bake time and temperature for each resist on 1.9mm thick NG4 substrates. These parameters will vary with substrate material and thickness.

Table A.1: **Spin Coat Parameters for AZ4210 and Ultra-I**

Resist	Dilution Ratio (PGMEA:Resist)	RPM	Spin Time
AZ4210	3:1	6000	40 s
Ultra-I	2:1	4000	40 s

Table A.2: **Soft Bake Parameters for AZ4210 and Ultra-I on NG4 Substrates**

Resist	Softbake Temp	Softbake Time
AZ4210	95°C	45 s
Ultra-I	90°C	120 s

A.2.2 Aerial Image Generation, Resist Exposure and Development

Each aerial image is generated by a Mach-Zehnder interferometer, detailed in Chapter 4. The interferometer is adjusted for a spatial period of 500 nm. All exposures were performed in an environmentally controlled room, with temperature held between 20-24°C and relative humidity held between 35-40%. No post exposure bake is used. Photoresists are developed via immersion in MF-319. The development time is the same as that used for soft bake optimization: 30 seconds with the developer temperature held at 20°C. After development, the substrates are rinsed with DI water, and dried using dry nitrogen.

A.2.3 Pattern Inspection, Image Processing and Measurement

For pattern inspection, a 2 nm thick layer of platinum is sputtered on top of each sample and the sample is then imaged with a scanning electron microscope (SEM, Hitachi SU3500). For LER measurements, high resolution images are taken with a field-emission scanning electron microscope (FE-SEM, JEOL JSM-7401F). The figures presented in Chapter 5 were taken with the FE-SEM. Image processing is used to determine values for line width, pitch, and line edge roughness for each exposure. All image processing is performed using the SuMMIT software package; details are given below.

A.2.4 SuMMIT Image Processing Details

SEM image processing used SuMMIT Litho Analysis Software Version 11.1.0. Details of the measurement recipe are give below.

Calibration and Filtering:

Default Calibration = *1*

Default Pixel Aspect Ratio = *1*

Include Bias correction ✓

Auto Noise Floor Estimation ✓

Post Filter Cutoff Period = *12.3828*

Prefilter Image Gaussian ✓

Edge Detection:

Relative Threshold = *0.5*

Threshold Reference = *Global Line*

Edge Detection Algorithm = *Linear*

✓Limit search range *20*

LER:

Outlier Correction ✓

Outlier Threshold = *3*

Tilt Removal ✓

PSD Analysis Appodization = *None*

Analysis Length = *Full Line*

Prevent Overlapping Segments ✓

A.3 Photoresist Contrast Optimization

Contrast was measured by exposing a grid of uniform intensity spots with increasing intensity across a wafer. The exposed grid was developed and remaining resist thickness measured. The lowest dose which initiated development was recorded as the lower threshold value D_1. The lowest dose which allowed complete development through the resist was recorded as the upper threshold value D_2. Contrast was estimated as $1/\mathrm{Log}(D_1/D_2)$. Contrast and threshold values both depend on development time, temperature and developer concentration. Wafers coated with photoresist of identical thickness each received a soft bake with a different time and temperature. Each wafer was then exposed and developed.

Appendix B

LIL Control Code

This sections contains the source code to control the laser interference lithography tool. The user inputs parameters in "User Set Up.m" and runs that file. The code is broken into functions, and the "Interference Tool Control.m" calls those functions in order.

B.1 User_Set_Up.m

```
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Interference Litho Tool Setup
%   V0.2
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%   All exposure parameters are set here
%   All units in mm
%
%   Version | Description
%     0.1   | Initial Release, for ESP300_5.m
%     0.2   | Release for PM500 revamp
%-----------------------------------------------------------------

tic

%-----------------------------------------------------------------
%       User inputs
%       Size of exposure grid, source select, exposure times
%-----------------------------------------------------------------


grid.Nx         = 2 ;        % Exposure count in x

grid.Ny         = 2 ;        % Exposure count in y

grid.Lx         = 8 ;        % Width along x in mm, **MAX 50MM**

grid.Ly         = 8 ;        % Width along y in mm, **MAX 50MM**
```

```
light.Source     = 'Laser';   % 'Laser' or 'LED'

light.Power      = 6 ;        % Power in mW

light.Off        = 'No';      % Turns source off at end of run

time.Min         = 20000 ;    % Min time in ms

time.Max         = 30000 ;    % Max time in ms

time.Incr         = 'Log';     % 'Linear' or 'Log' increments

%-----------------------------------------------------------------
%       Coordinate Map Generation
%       Beam center on center of chuck
%       Coordinates shifted accordingly
%-----------------------------------------------------------------

grid.Xcoord = linspace ( 0, grid.Lx ,grid.Nx ) - grid.Lx / 2;

grid.Ycoord = linspace ( 0, grid.Ly ,grid.Ny ) - grid.Ly / 2 ;

%-----------------------------------------------------------------
%       Time Map Generation
%-----------------------------------------------------------------

switch time.Incr
    case 'Linear'
        Time_Map = reshape ( floor( linspace( time.Min, time.Max, grid.Nx*grid.Ny
            )), [ grid.Nx ,grid.Ny ]);
    case 'Log'
        Time_Map = reshape ( floor( logspace( log10( time.Min ),log10( time.Max ),
             grid.Nx * grid.Ny )), [ grid.Nx, grid.Ny ]);
end

%-----------------------------------------------------------------
%       Call control program and pass values
%-----------------------------------------------------------------

Interference_Tool_Control( grid, light, Time_Map );
toc
```

B.2 Interference_Tool_Control.m

```
function [Exit_Flag] = Interference_Tool_Control( grid, light, Time )

% !!DO NOT RUN THIS FILE!! --> Execute from Set_Up.m
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Interference Lithography tool control
%   Controls the stage motion, laser power, shutter timing
%   Uses set up file for input parameters
%   The progress is plotted on an xy grid
%
%   Inputs          |   Type    |   Description
%   ****************************************
%   Nx,Ny           | Scalar    | number of grid points
%   Xabs,Yabs,Zabs| Vector    | X, Y, Z grid points, in order, units of mm
%   Beam_Power      | Scalar    | Laser Power in mW
%   Time_Map        | Array     | Array of exposure times, units of ms
%   Source          | String    | Sets either 'LASER' or 'LED'
%
%   Author:     David Miller
%   Created on: June 16, 2017
%
%   Version |   Description
%   ***********************************************************
%       1   |   Initial Release, code recylced from ESP300_5.m
%           |   uses new drivers called as functions
%       2   |   Revised for PM500 stage control, no z axis
%-----------------------------------------------------------------

addpath( genpath( pwd));

%-----------------------------------------------------------------
%       Test inputs for errors
%-----------------------------------------------------------------

assert(grid.Nx && grid.Ny > 0)

assert( length( grid.Xcoord ) == length( grid.Ycoord ))

assert( 100 > light.Power && light.Power > 0)

assert( size( Time, 1 ) == grid.Nx && size( Time,2 ) == grid.Ny )

%assert( strcmp(light.Source,'Laser') || strcmp(light.Source,'LED'))

%-----------------------------------------------------------------
%       Choose Proper Output
%-----------------------------------------------------------------

switch light.Source
    case 'Laser'
        Port = 1;
    case 'LED'
        Port = 3;
end
```

```
%-----------------------------------------------------------------
%       Establish serial connections PulseGen, Source, PM500
%-----------------------------------------------------------------

PulseGen_init;

if strcmp( light.Source, 'Laser' )
    Laser_init;
    SetLaserPower (light.Power);
end

PM_init;


%-----------------------------------------------------------------
%       Home all Axis
%-----------------------------------------------------------------

HomeAxis('all');

MotionStatus(1);

%-----------------------------------------------------------------
%       Get positions and move
%-----------------------------------------------------------------

X_motion_command = 'XG';                        %abs motion command
Y_motion_command = 'YG';                        %abs motion command

for i = 1 : grid.Nx
    for j = 1 : grid.Ny

        X = num2str( grid.Xcoord( i )*1000);        %x position to move to
        X_command = strcat( X_motion_command, X ); %complete motion command

        if (-1)^i>0                                 %flips index direction each
            time through loop
            k = grid.Ny + 1 - j;                    %snake instead of zigzag
        else
            k = j;
        end

        Y = num2str( grid.Ycoord( k )*1000);        %Y position to move to
        Y_command = strcat( Y_motion_command, Y ); %complete motion command

        fprintf( PM ,X_command )                    %print to port X coord
        fprintf( PM ,Y_command )                    %print to port Y coord

        MotionStatus(1);

        pause( 2 )


        scatter( grid.Xcoord( i ), grid.Ycoord( k ), 'X' )
        text( grid.Xcoord( i )+1 , grid.Ycoord( k ), num2str( Time( i,k )))
        title({'Exposure Map';'Time in ms'; sprintf('Laser Power = %d mW', light.
            Power)})
        xlabel('Position (mm)')
```

```
        ylabel('Position (mm)')
        hold on

        PulseGenOut( Port, Time( i,k ) );

        pause( 1 );

    end
end

%-----------------------------------------------------------------
%       Close serial connections and turn off Laser
%-----------------------------------------------------------------

if strcmp( light.Source, 'Laser' )
    if strcmp(light.Off, 'No')
        Laser_close_on;
    else
        Laser_close_on
    end
end

PulseGen_close

PM_close

Exit_Flag = 1;
```

B.3 PulseGen_init.m

```
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Pulse Generator Object
%   Creates an object that allows control of TTL devices
%   Establishes serial connection and opens port
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Communicates w. Arduino Pulse Generator over serial
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global PulseGen

PulseGen            = serial('COM6');   % serial object "PulseGen"
PulseGen.BaudRate   = 9600;             % required by Arduino
PulseGen.Parity     = 'none';
PulseGen.DataBits   = 8;
PulseGen.StopBits   = 1;
PulseGen.Terminator = 13;

fopen( PulseGen )                       % open port
```

B.4 PulseGen_close.m

```
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Close Pulse Generator
%   Closes Pulse Generator connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global PulseGen

fclose( PulseGen )
delete( PulseGen )
clear PulseGen
```

B.5 PulseGenOut.m

```
function flag = PulseGenOut(port,time)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Writes 3 numbers to serial port, comma separated, \r terminated
%       First number is function type
%           (0=arduino timed, 1=port on, 2=port off)
%       Second number is port number
%       Third number is duration
%           (not active for port on/off commands)
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%       2   | Added fix for times >25 second.
%           | Separate open/close commands issued by computer.
%-----------------------------------------------------------------
global PulseGen

time = round( time );

if time>20000
    Shutter_Open(port);
    pause(time/1000)
    Shutter_Close(port);
else
    fprintf( PulseGen, strcat( num2str(0), ',', num2str( port ), ',', num2str(
        time )))
    pause(time/1000)
end
```

B.6 Shutter_Open.m

```
function flag = Shutter_Open(port)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:     David Miller
%   Created on: August 7, 2018
%
%
%   Version | Description
%       1   | Initial Release
%-----------------------------------------------------------------
global PulseGen

fprintf( PulseGen, strcat( num2str(1) , ',' , num2str( port ), ',' , num2str(0) ))

end
```

B.7 Shutter_Close.m

```
function flag = Shutter_Close(port)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:     David Miller
%   Created on: August 7, 2018
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global PulseGen

fprintf( PulseGen, strcat( num2str(2) , ',' , num2str( port ), ',' , num2str(0) ))

end
```

B.8 PM_init.m

```
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   PM500 Object
%   Creates an object that allows control of PM500 motion controler
%   Establishes serial connection and opens port
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%   All commands from Newport ESP300 Manual
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global PM

PM              = serial('COM4');   % create serial object "ESP"
PM.BaudRate     = 19200;            % required by ESP300
PM.Parity       = 'none';
PM.DataBits     = 8;
PM.StopBits     = 1;
PM.Terminator   = 13;

fopen( PM )                         % open port
```

B.9 PM_close.m

```
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Close PM500
%   Closes PM500 connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global PM

fclose( PM )
delete( PM )
clear PM
```

B.10 HomeAxis.m

```
function flag = HomeAxis(axis)
%------------------------------------------------------------------
%------------------------------------------------------------------
%   Homes PM500 Axis
%   Inputs: X, Y, or 'all'
%   Waits until all motion is complete
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%------------------------------------------------------------------
global PM

if axis == 'all'
    fprintf( PM, 'XF0' ); % home position sequentially on all axis
    fprintf( PM, 'YF0' )
else
    fprintf( PM, strcat( axis , 'F0' ))
end

MotionStatus(1);
```

B.11 MotionStatus.m

```
function flag = MotionStatus(wait)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Checks for motion on all PM500 axes
%   Waits until all motion is complete
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global PM

x_motion = 1;

y_motion = 1;

while x_motion == 1
    fprintf( PM, 'XSTAT' );
    x_status  = fscanf( PM, '%s' );
    while y_motion  == 1
        fprintf( PM, 'YSTAT' );
        y_status  = fscanf( PM, '%s' );
        y_motion  = strcmp( 'YB', y_status );
        pause( wait );
    end
    x_motion = strcmp( 'XB', x_status );
    pause( wait );
end

flag = 0;
```

B.12 Laser_init.m

```
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Ar-Ion Laser Object
%   Creates and object that allows control of the Coherent I-300c
%   Establishes serial connection and opens port
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Laser COM comes from found code, can't find the manual
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global Laser

Laser                   = serial('COM5');   % create serial object "Laser"
Laser.Terminator        = 'CR/LF';
Laser.DataTerminalReady = 'off';
Laser.RequestToSend     = 'off';

fopen( Laser );                             % open port
```

B.13 Laser_close.m

```
%---------------------------------------------------------------
%---------------------------------------------------------------
%   Close Laser
%   Turns off laser (if not already), closes Laser connection and
%   removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%---------------------------------------------------------------
global Laser

fprintf( Laser,'LASER=0' );     % turns off laser if it is still on

fclose( Laser )
delete( Laser )
clear Laser
```

B.14 Laser_close_on.m

```
%---------------------------------------------------------------
%---------------------------------------------------------------
%   Close Laser
%   Closes Laser connection and removes object without
%   shutting off laser
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%---------------------------------------------------------------
global Laser

fclose( Laser )
delete( Laser )
clear Laser
```

B.15 SetLaserPower.m

```
function flag = SetLaserPower(mW)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Laser Power Control
%   Sets output power, takes units of mW as input
%   LaserPower(mW)
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------
global Laser

W = mW/1000;                            % set laser power here in W

fprintf( Laser, 'LASER=1' );            % turns on laser

fprintf( Laser, 'LIGHT=%.3f\n', W );    % sets value for feedback control

flag = 0;
```

Appendix C

Stepper Control Code

This sections contains the source code to control the i-line stepper. The user inputs parameters in "User Set Up.m" and runs that file. The code is broken into functions, and the "Interference Tool Control.m" calls those functions in order.

C.1 User_Set_Up.m

```
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Stepper Litho Tool Setup - DOSE/FOCUS Matrix
%   V0.1
%
%   Author:     David Miller
%   Created on: August 14, 2019
%
%   All exposure parameters are set here
%   All units in mm
%
%   Version | Description
%     0.1   | Initial Release
%-----------------------------------------------------------------

tic

%-----------------------------------------------------------------
%       User inputs
%       Size of exposure grid, source select, exposure times
%-----------------------------------------------------------------


grid.Nx         = 9;        % Exposure count in x

grid.Ny         = 9;        % Exposure count in y

grid.Nz         = 9;        % Focus position count !!!MUST EQUAL Nx!!!

grid.Lx         = 16;       % Width along x in mm, **MAX 50MM**

grid.Ly         = 16;       % Width along y in mm, **MAX 50MM**

grid.Lz         = 0.040;    % Focus range, in mm, for FOCUS/DOSE matrix
```

```
focus.step      = 0.005;    % step size for autofocus in mm

focus.start     = 17.000;   % starting point for autofocus

focus.offset    = -0.075;   % focus offset btwn UV and Red

light.Source    = 'LED';    % 'Laser' or 'LED'

light.Power     = 0;        % Power in mW

light.Off       = 'No';     % Turns source off at end of run

time.Min        = 5000;     % Min time in ms

time.Max        = 30000;    % Max time in ms

time.Incr      = 'Log';     % 'Linear' or 'Log' increments

%-----------------------------------------------------------------
%       Coordinate Map Generation
%-----------------------------------------------------------------

grid_map = Coordinate_Map( grid, focus );

%-----------------------------------------------------------------
%       Time Map Generation
%-----------------------------------------------------------------

switch time.Incr
    case 'Linear'
        Time_Vector =  floor( linspace( time.Min, time.Max, grid.Nx ));
    case 'Log'
        Time_Vector =  floor( logspace( log10( time.Min ),log10( time.Max ), grid.
            Nx ));
end

[ ~, Time_Map ] = meshgrid( Time_Vector, Time_Vector );
%-----------------------------------------------------------------
%       Call control program and pass values
%-----------------------------------------------------------------

Stepper_Tool_Control( grid_map, Time_Map, light, focus, grid.Nx );
toc
```

C.2 Coordinate_Map.m

```
function out = Coordinate_Map( grid, focus )

%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Coordinate Map Generation
%   Coordinate center on center of chuck
%
%
%   Author:     David Miller
%   Created on: August 21, 2019
%
%   Version | Description
%       1   | Initial Release
%--------------------------------------------------------------------

chuck_center    = chuck_data('get');

Xcoords         = linspace ( -grid.Lx/2, grid.Lx/2, grid.Nx ) + chuck_center.
    Xcenter  ;

Ycoords         = linspace ( -grid.Ly/2, grid.Ly/2, grid.Ny ) + chuck_center.
    Ycenter  ;

Zcoords         = linspace ( -grid.Lz/2, grid.Lz/2 , grid.Nz ) + focus.offset;

[grid_map.Xcoord, grid_map.Ycoord]  = meshgrid( Xcoords, Ycoords ) ;

[grid_map.Zcoord, ~]                = meshgrid( Zcoords, Ycoords  );

out = grid_map;

end
```

C.3 Stepper_Tool_Control.m

```
function [Exit_Flag] = Interference_Tool_Control( grid, Time, light, focus, Nx )

% !!DO NOT RUN THIS FILE!! --> Execute from Set_Up.m
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Interference Lithography tool control
%   Controls the stage motion, laser power, shutter timing
%   Uses set up file for input parameters
%   The progress is plotted on an xy grid
%
%   Inputs         |   Type     |   Description
%   ****************************************
%   Nx,Ny          | Scalar     | number of grid points
%   Xabs,Yabs,Zabs| Vector     | X, Y, Z grid points, in order, units of mm
%   Beam_Power     | Scalar     | Laser Power in mW
%   Time_Map       | Array      | Array of exposure times, units of ms
%   Source         | String     | Sets either 'LASER' or 'LED'
%
%   Author:     David Miller
%   Created on: June 16, 2017
%
%   Version |   Description
%   *******************************************************
%       1   |   Initial Release, code recylced from ESP300_5.m
%           |   uses new drivers called as functions
%       2   |   Revised for PM500 stage control, no z axis
%-----------------------------------------------------------------

addpath( genpath( pwd));

%-----------------------------------------------------------------
%       Test inputs for errors
%-----------------------------------------------------------------

%assert(grid.Nx && grid.Ny > 0)

%assert( length( grid.Xcoord ) == length( grid.Zcoord ))

%assert( 100 > light.Power && light.Power > 0)

%assert( size( Time, 1 ) == grid.Nx && size( Time,2 ) == grid.Ny )

%assert( strcmp(light.Source,'Laser') || strcmp(light.Source,'LED'))

%-----------------------------------------------------------------
%       Choose Proper Output
%-----------------------------------------------------------------

switch light.Source
    case 'Laser'
        Port = 1;
    case 'LED'
        Port = 2;
end
```

```
%----------------------------------------------------------------
%       Establish serial connections PulseGen, Source
%----------------------------------------------------------------

PulseGen_init;

if strcmp( light.Source, 'Laser' )
    Laser_init;
    SetLaserPower (light.Power);
end

%----------------------------------------------------------------
%    Establish connection with XPS
%----------------------------------------------------------------

XPS_init

%----------------------------------------------------------------
%    Home all Axis
%----------------------------------------------------------------

%HomeAxis();

%MotionStatus(10);

%HomeAxis();

%MotionStatus(1);

%----------------------------------------------------------------
%    Get positions and move
%----------------------------------------------------------------

XX = reshape( grid.Xcoord, [], 1 ) ;

YY = reshape( grid.Ycoord, [], 1 ) ;

ZZ = reshape( grid.Zcoord, [], 1 ) ;

TT = reshape( Time, [], 1 ) ;

for i = 1 : Nx^2

        XPS_abs_move_X( XX( i ) )
        XPS_abs_move_Y( YY( i ) )

        pause( 0.5 ) ;

        z_zero = auto_focus( focus.start , focus.step);

        XPS_abs_move_Z( z_zero + ZZ(i));

        pause( 0.5 ) ;

        scatter( XX( i ), YY( i ), 'X' )
        text( XX( i )+1 , YY( i ), num2str( Time( i )))
        text( XX( i )+1 , YY( i )+1, num2str( ZZ(i) ))
        title({'Exposure Map';'Time in ms'; sprintf('Laser Power = %d mW', light.
```

```
            Power)})
        xlabel('Position (mm)')
        ylabel('Position (mm)')
        hold on

        PulseGenOut( Port, Time( i ) ) ;

        pause( 0.5 );


end

%-----------------------------------------------------------------
%       Close serial connections and turn off Laser
%-----------------------------------------------------------------

if strcmp( light.Source, 'Laser' )
    if strcmp(light.Off, 'No')
        Laser_close_on;
    else
        Laser_close_on
    end
end

PulseGen_close

XPS_close

Exit_Flag = 1;
```

C.4 chuck_data.m

```
function out = chuck_data(command,varargin);
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Holds info on wafer chuck position
%   Can set the data using chuck_data('set',xposition,yposition)
%   Can read the data using chuck_data('get')
%
%   all units are mm
%
%   Author:     David Miller
%   Created on: August 14, 2019
%
%
%   Version | Description
%       1   | Initial Release
%-----------------------------------------------------------------

system_parameters;

switch command
    case 'get'
        out = chuck;
    case 'set'
        if ~isempty(varargin)
            chuck.Xcenter = varargin{1};
            chuck.Ycenter = varargin{2};
        end

        matlab.io.saveVariablesToScript('system_parameters.m', 'chuck','SaveMode',
            'Update');
        out = 1;
end

end
```

C.5 XPS_init.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   XPS Controller Object
%   Creates and object that allows control of XPS motion controller
%   Establishes .NET assembly and opens connection
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 7, 2019
%
%   All commands from Newport XPS-RL Manual
%
%   Version | Description
%       1   | Initial Release, code partially recycled from PM500.m
%-----------------------------------------------------------------
global XPSc

asmXPS  = NET.addAssembly('Newport.XPS.CommandInterface');      % make assembly
    visible from Matlab
XPSc    = CommandInterfaceXPS.XPS();                            % make the
    instantiation
[~]     = XPSc.OpenInstrument('192.168.254.254',5001,100);      % connect to XPS
    controller

[code]  = XPSc.GroupInitialize('Group1');                       % initialize x
    axis
[code]  = XPSc.GroupInitialize('Group2');                       % initialize y
    axis
[code]  = XPSc.GroupInitialize('Group3');                       % initialize z
    axis
```

C.6 XPS_close.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Close PM500
%   Closes PM500 connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-----------------------------------------------------------------

global XPSc
global asmXPS

[code] = XPSc.CloseInstrument;

delete( XPSc )
delete( asmXPS )

clear XPSc
```

C.7 HomeAxis.m

```
function flag = HomeAxis(~)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Homes ALL Axis
%   Inputs: dummy input
%   Waits until all motion is complete
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-----------------------------------------------------------------
global XPSc

[~] = XPSc.GroupHomeSearch('Group1') ; % home x axis
pause(1)
[~] = XPSc.GroupHomeSearch('Group2') ; % home y axis
pause(1)
[~] = XPSc.GroupHomeSearch('Group3') ; % home z axis

MotionStatus(1);                        % waits for motion to finish
```

C.8 MotionStatus.m

```
function flag = MotionStatus(wait)
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Checks for motion on all PM500 axes
%   Waits until all motion is complete
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-----------------------------------------------------------------
global XPSc
global asmXPS

motion = 1;

while motion ~= 0
    motion = XPSc.GroupMotionStatusGet('Group1',1);
    motion = motion + XPSc.GroupMotionStatusGet('Group2',1);
    motion = motion + XPSc.GroupMotionStatusGet('Group3',1);
    pause( wait );
end

flag = 0;
```

C.9 XPS_abs_move_X.m

```
function flag = XPS_abs_move_X(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-----------------------------------------------------------------

global XPSc

XPSc.GroupMoveAbsolute('Group1', position, 1);

MotionStatus(1);
```

C.10 XPS_abs_move_Y.m

```
function flag = XPS_abs_move_Y(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-------------------------------------------------------------------

global XPSc

XPSc.GroupMoveAbsolute('Group2',position,1);

MotionStatus(1);
```

C.11 XPS_abs_move_Z.m

```
function flag = XPS_abs_move_Z(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-------------------------------------------------------------------

global XPSc

XPSc.GroupMoveAbsolute('Group3',position,1);

MotionStatus(1);
```

C.12 XPS_inc_move_X.m

```
function flag = XPS_inc_move_X(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-------------------------------------------------------------------

global XPSc

XPSc.GroupMoveRelative('Group1',position,1);

MotionStatus(1)
```

C.13 XPS_inc_move_Y.m

```
function flag = XPS_inc_move_Y(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%-------------------------------------------------------------------
global XPSc

XPSc.GroupMoveRelative('Group2',position,1);

MotionStatus(1)
```

C.14 XPS_inc_move_Z.m

```
function flag = XPS_inc_move_Z(position)

% !!DO NOT RUN THIS FILE DIRECTLY!!
%---------------------------------------------------------------
%---------------------------------------------------------------
%   Absolute motion command for XPS
%   **input arguments**
%   position in mm
%
%   Author:     David Miller
%   Created on: November 6, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recycled from ESP300_5.m
%---------------------------------------------------------------
global XPSc

XPSc.GroupMoveRelative('Group3', position, 1);
pause(0.1)
MotionStatus(1)
```

C.15 XPS_position_move_X.m

```
function position = XPS_position_X(~)

global XPSc

[~, position] = XPSc.GroupPositionCurrentGet('Group1',1) ;

position = double(position);
```

C.16 XPS_position_move_Y.m

```
function position = XPS_position_Y(~)

global XPSc

[~, position] = XPSc.GroupPositionCurrentGet('Group2',1) ;

position = double(position);
```

C.17 XPS_position_move_Z.m

```
function position = XPS_position_Z(~)

global XPSc

[~, position] = XPSc.GroupPositionCurrentGet('Group3',1) ;

position = double(position);
```

C.18 PulseGen_init.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Pulse Generator Object
%   Creates and object that allows control of TTL devices
%   Establishes serial connection and opens port
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Communicates w. Arduino Pulse Generator over serial
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global PulseGen

PulseGen            = serial('COM6');   % create serial object "PulseGen"
PulseGen.BaudRate   = 9600;             % required by Arduino
PulseGen.Parity     = 'none';
PulseGen.DataBits   = 8;
PulseGen.StopBits   = 1;
PulseGen.Terminator = 13;

fopen( PulseGen )                       % open port
```

C.19 PulseGen_close.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Close Pulse Generator
%   Closes Pulse Generator connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global PulseGen

fclose( PulseGen )
delete( PulseGen )
clear PulseGen
```

C.20 PulseGenOut.m

```
function flag = PulseGenOut(port,time)
%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Writes 3 numbers to serial port, comma separated, \r terminated
%       First number is function type (0=arduino timed, 1=port on, 2=port off)
%       Second number is port number
%       Third number is duration (not active for port on/off commands)
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%       2   | Added fix for times >25 second.  Separate open/close commands
%             issued by computer.
%--------------------------------------------------------------------
global PulseGen

time = round( time );

if time>20000
    Shutter_Open(port);
    pause(time/1000)
    Shutter_Close(port);
else
    fprintf( PulseGen, strcat( num2str(0), ',', num2str( port ), ',', num2str(
        time )))
    pause(time/1000)
end
```

C.21 Shutter_Open.m

```
function flag = Shutter_Open(port)
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:      David Miller
%   Created on: August 7, 2018
%
%
%   Version | Description
%       1    | Initial Release
%-------------------------------------------------------------------
global PulseGen

fprintf( PulseGen, strcat( num2str(1) , ',' , num2str( port ), ',' , num2str(0) ))

end
```

C.22 Shutter_Close.m

```
function flag = Shutter_Close(port)
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Sends command to Arduino Pulse Gen
%   Turns on Port for Time
%   Time in ms!
%
%   Author:      David Miller
%   Created on: August 7, 2018
%
%
%   Version | Description
%       1    | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global PulseGen

fprintf( PulseGen, strcat( num2str(2) , ',' , num2str( port ), ',' , num2str(0) ))

end
```

C.23 Laser_init.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%------------------------------------------------------------------------
%------------------------------------------------------------------------
%   Ar-Ion Laser Object
%   Creates and object that allows control of the Coherent I-300c
%   Establishes serial connection and opens port
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%   Laser COM comes from found code, can't find the manual
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%------------------------------------------------------------------------
global Laser

Laser                   = serial('COM5');   % create serial object "Laser"
Laser.Terminator        = 'CR/LF';
Laser.DataTerminalReady = 'off';
Laser.RequestToSend     = 'off';

fopen( Laser );                             % open port
```

C.24 Laser_close.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%------------------------------------------------------------------------
%------------------------------------------------------------------------
%   Close Laser
%   Turns off laser (if not already), closes Laser connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%------------------------------------------------------------------------
global Laser

fprintf( Laser,'LASER=0' );     % turns off laser if it is still on

fclose( Laser )
delete( Laser )
clear Laser
```

C.25 Laser_close_on.m

```
% !!DO NOT RUN THIS FILE DIRECTLY!!
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Close Laser
%   Turns off laser (if not already), closes Laser connection and removes object
%   **Takes no input arguments**
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global Laser

fclose( Laser )
delete( Laser )
clear Laser
```

C.26 SetLaserPower.m

```
function flag = SetLaserPower(mW)
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Laser Power Control
%   Sets output power, takes units of mW as input
%   LaserPower(mW)
%
%   Author:     David Miller
%   Created on: October 25, 2017
%
%
%   Version | Description
%       1   | Initial Release, code recylced from ESP300_5.m
%-------------------------------------------------------------------
global Laser

W = mW/1000;                            % set laser power here in W

fprintf( Laser, 'LASER=1' );            % turns on laser

fprintf( Laser, 'LIGHT=%.3f\n', W );    % sets value for feedback control

flag = 0;
```

C.27 auto_focus.m

```
function [focus_ZES_position] = auto_focus( Focus_approx, step_size )
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Steps Through Focus and finds position corresponding to FES=0
%
%   step_size in mm
%
%
%   Author:      David Miller
%   Created on: December 4, 2019
%
%   Version | Description
%       1   | Initial Release
%-------------------------------------------------------------------

global XPSc

XPS_abs_move_Z(Focus_approx) ;  % starting scan position

max_range = 0.12 ;              % maximum scan range

total_steps = max_range / step_size;

XPS_inc_move_Z(-0.06) ;

pause(0.5)

FES = signal_read_in_full();

for i=2:total_steps
    XPS_inc_move_Z(step_size) ;
    FES(i) = signal_read_in_full();
    position(i) = XPS_position_Z() ;
    pause(0.1)
end

%find approx zero crossing

[ dummy1, max_z ] = max(FES);
[ dummy2, min_z ] = min(FES);

FES_trunc = FES(max_z:min_z);                              % truncates to only
    center signal range

FES_shift = circshift(FES_trunc,1);
FES_shift(1) = 1;

z_cross = find( FES_trunc .* FES_shift <=0);    % returns index value

focus_ZES_position = position( z_cross + max_z - 1 );
```

C.28 signal_read_in_manual.m

```
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%    Analog Read-in
%    reads in 4 differential analog channels from labjack
%
%    Pin AI0/AI1 are input #1
%    Pin AI2/AI3 are input #2
%    Pin AI4/AI5 are input #3
%    Pin AI6/AI7 are input #4
%
%    Author:     David Miller
%    Created on: August 15, 2019
%
%    Version | Description
%       1    | Initial Release
%-------------------------------------------------------------------
clc
clear

test=1;
while test==1

%-------------------------------------------------------------------
                        %Configure LabJack
%-------------------------------------------------------------------

ljAsm = NET.addAssembly('C:\Program Files (x86)\LabJackU12Legacy\drivers\LJDotNet.
    dll');

idnum       = -1;                                    % Using first found U12
demo        = 0;                                     % Normal operations
stateIO     = 0;                                     % Output states for IO0-IO3
errorString = NET.createArray('System.Char', 50);   % Empty string for error
    message

%-------------------------------------------------------------------
%                       AISample Parameters
%-------------------------------------------------------------------

updateIO    = 0;                                     % State values are only being
    read
ledOn       = 0;                                     % Turning LED on
numChannels = 4;                                     % Reading 2 channels

% Configure Differential Channels (8-11)

channel     = NET.createArray('System.Int32', 4);   % Array of channel numbers
channel(1)  = 8;
channel(2)  = 9;
channel(3)  = 10;
channel(4)  = 11;

% Differential Channel Gains

diff_gain = 1;
```

```
gain    = NET.createArray('System.Int32', 4);
gain(1) = diff_gain;
gain(2) = diff_gain;
gain(3) = diff_gain;
gain(4) = diff_gain;

disableCal = 0;                                % Will apply calibration constants
overVoltage = 0;                               % Returns if overvoltage was
    detected (>1)

% Returned voltage readings
% Pass an array of all zeros.

voltages = NET.createArray('System.Single', 4);
voltages(1) = 0;
voltages(2) = 0;
voltages(3) = 0;
voltages(4) = 0;

disp(['Driver version: ', num2str(lj.LabJack.GetDriverVersion())]);

%---------------------------------------------------------------------
%                            Call AISample
%---------------------------------------------------------------------

[errorCode, idnum, stateIO, overVoltage] = lj.LabJack.AISample(idnum, demo,
    stateIO, updateIO, ledOn, numChannels, channel, gain, disableCal, overVoltage,
    voltages);
if(errorCode ~= 0)
    lj.LabJack.GetErrorString(errorCode, errorString);
    disp(['AISample error ' num2str(errorCode) ' : ' char(System.String(
        errorString))])
    return
end

%---------------------------------------------------------------------
%                          Display Values
%---------------------------------------------------------------------

disp(['AI' num2str(channel(1)) ' = ' num2str(voltages(1)) ' V'])
disp(['AI' num2str(channel(2)) ' = ' num2str(voltages(2)) ' V'])
disp(['AI' num2str(channel(3)) ' = ' num2str(voltages(3)) ' V'])
disp(['AI' num2str(channel(4)) ' = ' num2str(voltages(4)) ' V'])

%---------------------------------------------------------------------
%                        Sums, Diffs, Normalize
%---------------------------------------------------------------------

total           = voltages(1) + voltages(2) + voltages(3) + voltages(4);

% +45 deg position

sum_12          = voltages(1) + voltages(2);

sum_34          = voltages(3) + voltages(4);

diff_12_24      = sum_12 - sum_34;
```

```
diff_12_24_norm = diff_12_24 / total;

% -45 deg position
sum_14          = voltages(1) + voltages(4);
sum_23          = voltages(2) + voltages(3);
diff_14_23      = sum_14 - sum_23;
diff_14_23_norm = diff_14_23 / total;

% vertical
diff_vert = (voltages(1)-voltages(3))/total;
%horizontal
diff_horz = (voltages(2)-voltages(4))/total;

% focus error signal
sum_13          = voltages(1) + voltages(3);
sum_24          = voltages(2) + voltages(4);
diff_13_24      = sum_13 - sum_24;
diff_13_24_norm = diff_13_24 / total;

p45     = diff_12_24_norm;
p135    = diff_14_23_norm;
FES     = diff_13_24_norm

scatter([diff_horz -1],[diff_vert FES],400,'k','X','LineWidth',3)
axis([-1 1 -1 1])
pause(0.1)

end
```

Appendix D

Vector EM Mask Simulation

This section contains the source code for a full vector simulation of projection lithography, used in Chapter 7. The user sets parameters in "UserSetUp.m" and runs that script. The functions for pupil filtering and for spatial coherence simulation are incomplete. A Laplace mask solver is given at the end.

D.1 UserSetUp.m

```
%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Computes the exposure intensities for single set of input params
%   Computes optical contrast
%   Computes the expected resist profile given the computed intensity
%
%   Author:     D. Miller
%   Written:    March 31, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%--------------------------------------------------------------------
%--------------------------------------------------------------------
tic

addpath('Masks')

%--------Mask Details------------------

Mask_Name    = 'circle' ;    % mask to simulate 'single line' 'isolated line' '
    grating' 'circle' '2 lines'

Retard       = pi ;          % retardance of mask elements

%--------Illumination-----------------

Lambda       = 364*10^-9 ;       % sim wavelength ( for grid size )

Polarized    = 'Yes' ;           % Yes for polarized light, No for unpolarized

E_illum      = [ 0 1 ] ;         % polarization of illuminating field
```

```
Illum_Ang   = [ 0.0 0.0 ] ;     % Incident illumination angle in Radians [
    Horizontal Vertical ]

Sigma       = 0.0 ;             % Spacial Coherence Function (between 0 and 1)


%--------Imaging Parameters------------

NA_ent      = 0.1 ;             % Enterance pupil (object side) NA

Demag       = 9 ;               % demagnification mask to image plane

Delta_Z     = 0*Lambda ;        % Wafer defocus IN DISTANCE

Radiometric = 1 ;               % Radiometric Correction for Magnification 1 or 0

%Aberrations                    IN WAVES!!
Focus       = 0 ;               % Focus term (in pupil)
Astig_H     = 0 ;               % horizontal Astigmatism
Astig_45    = 0 ;               % 45 deg Astigmatism
Coma_X      = 0 ;               % X Coma
Coma_Y      = 0 ;               % Y Coma
Spherical   = 0 ;               % 3rd Order spherical

%--------Resist Parameters-------------

Over_Expose     = 1 ;          % overexposure factor

Resist_Gamma    = 3.5 ;

Dose_High       = 0.25 ;


%--------Sim Parameters----------------

N           = 1026 ;            % simulation grid size

Spacing     = Lambda/4 ;        % Mask plane grid spacing Nyquist ( image plane
    spacing is 1/demag )

Oversample  = 1 ;               % Oversample in image plane (interpolate) via zero
     padding

Render      = 1 ;               % Turns on polarization and mask plots

%**************************************************************************
%                            Store Values
%**************************************************************************

save( 'RunSettings' )

%**************************************************************************
%                            Input Error Check
%**************************************************************************

Error_Flag = ErrorCheck( 'RunSettings' ) ;
```

```
assert( dot( ones( size( Error_Flag )) , Error_Flag ) == 0, 'Input Param Error,
    Check Flag Values' )


%**************************************************************************
%                            Function Calls
%**************************************************************************

if Sigma > 0

    [ coords , I_fp ] = SpatialCoherence( 'RunSettings' ) ;

elseif Sigma == 0

    switch Polarized
        case 'Yes'

            [ coords , I_fp ]  = ProjectionCode( 'RunSettings' ) ;

            I_fp                = 1* I_fp ;

        case 'No'
            % simulate unpolarized light --> runs ProjectionCode for each
                polarization & sums incoherently

            E_illum = [ 0 1 ];
            save( 'RunSettings1' )
            [ coords , I_fp_vert ]  = ProjectionCode( 'RunSettings1' ) ;

            E_illum = [ 1 0 ];
            save( 'RunSettings2' )
            [ coords , I_fp_horz ]  = ProjectionCode( 'RunSettings2' ) ;

            I_fp                    = ( I_fp_vert + I_fp_horz ) / 2 ;
    end
end

m            = OpticalContrast( I_fp, N )

profile      = Resist( I_fp, Dose_High/Over_Expose, Resist_Gamma ) ;

linewidth    = MeasureLinewidth( profile, coords, N )


%**************************************************************************
%                               Plot Output
%**************************************************************************

x = ( -N/2 : 1 : N/2-1 ) * Spacing/Demag ;          % Image space coordinates

%   1D Intensity X-Section
figure;
plot( x/10^-6 , log10(I_fp( Oversample*N/2, 1:Oversample:end )))
xlabel( 'Position (\mum)' )
ylabel( 'Normalized Inensity' )

%   2D Intensity Distribution
figure;
```

```
%pcolor( x( 1:10:end, 1:10:end ) / 10^-6, x( 1:10:end, 1:10:end ) / 10^-6, I_fp(
    1:10:end, 1:10:end ))
pcolor( x(1:10:end)/10^-6, x(1:10:end)/10^-6, I_fp( 1:10:end, 1:10:end ))
shading( 'interp' )
colormap( 'gray' )
xlabel( 'Position (\mum)' )
ylabel( 'Position (\mum)' )

%   Resist X-Section
figure;
plot( x/10^-6 , profile( 1 * N/2, 1:Oversample:end ))
xlabel( 'Position (\mum)' )
ylabel( 'Normalized Height' )

%   Top-Down Resist Plot
figure;
pcolor( x/10^-6, x/10^-6, profile(:,:))
shading( 'interp' )
colormap( 'copper' )
xlabel( 'Position (\mum)' )
ylabel( 'Position (\mum)' )

toc
```

D.2 ErrorCheck.m

```
function Eflag = ErrorCheck( file_name )

%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Checks input values for errors
%   Returns Flag = 0 for no error
%   Flag = 1 for any error in each of the parameter sections
%
%   Author:     D. Miller
%   Written:    April 30, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------

load( file_name );

Eflag = [];

%--------Mask Details------------------

if isempty( Mask_Name )
    Eflag = [ Eflag ; [1] ] ;

elseif isempty( Retard )
    Eflag = [ Eflag ; [1] ] ;

else
    Eflag = [ Eflag ; [0] ] ;

end


%--------Illumination-----------------

if isempty( Polarized )
    Eflag = [ Eflag ; [1] ] ;

elseif Illum_Ang(1) > pi/2
        Eflag = [ Eflag ; [1] ] ;

elseif Illum_Ang(2) > pi/2
        Eflag = [ Eflag ; [1] ] ;

elseif isempty( Sigma )
        Eflag = [ Eflag ; [1] ] ;

elseif ~isequal( size( E_illum ) , [ 1 2 ] )
        Eflag = [ Eflag ; [1] ] ;

else
    Eflag = [ Eflag ; [0] ] ;
```

```
end


%--------Imaging Parameters------------

if NA_ent <= 0 || NA_ent > 1
    Eflag = [ Eflag ; [1] ] ;

elseif Demag < 0 || Demag*NA_ent > 1
    Eflag = [ Eflag ; [1] ] ;

elseif Over_Expose < 0
    Eflag = [ Eflag ; [1] ] ;

else
    Eflag = [ Eflag ; [0] ] ;
end


%--------Resist Parameters-------------

if Resist_Gamma <= 0
    Eflag = [ Eflag ; 1 ] ;

elseif Dose_High <= 0
    Eflag = [ Eflag ; 1 ] ;

else
    Eflag = [ Eflag ; 0 ] ;
end


%--------Sim Parameters----------------


if Render ~= 0 && Render ~= 1
    Eflag = [ Eflag ; 1 ] ;

elseif Oversample < 1
    Eflag = [ Eflag ; 1 ] ;

else
    Eflag = [ Eflag ; 0 ] ;
end
```

D.3 ProjectionCode.m

```
function [ coord, I_focalplane ] = ProjectionCode( file_name )

%------------------------------------------------------------------------
%------------------------------------------------------------------------
%   ProjectionCode
%   Lots of inputs: Lambda, N, dL, Demag, Retard, E_illum, Delta_Z, Mask_Name,
    Render, Oversample
%
%   Simulates mask projection onto resist surface
%   Uses vector Fourier BPM:
%
%   Establishes the coordinate system
%   Establishes the polarization "projections"
%   Calls mask function
%   Calls Fourier Prop
%   Returns coord struct and focal plane intensity
%
%   Original Author:RRM 11/12/15
%   Rewritten:      D. Miller  March 31, 2020
%   Updated:        May 3, 2020
%
%   Rev History
%   v1      -   Initial Release, Re-write by D Miller March 2020
%   v2      -   Updated to include structs
%
%
%------------------------------------------------------------------------
%------------------------------------------------------------------------

load( file_name );

%************************************************************************
%                                UNITS
%************************************************************************

milli = 10^-3;
micro = 10^-6;
nano  = 10^-9;


%************************************************************************
%                   CALCULATED SIMULATION PARAMETERS
%************************************************************************

Aberration  = [  0, 0, 0, Focus, Astig_H, Astig_45, Coma_X, Coma_Y, Spherical ] ;
     % Seidel (zernike) coefficients

NA_ext      = Demag * NA_ent ;          % NA of objective (exit pupil)
DLfoc       = 0.61 * Lambda / NA_ext ;  % Difraction limit at focus plane,
    circular aperture
fCutOff     = 1/( DLfoc * Demag ) ;     % Gaussian frequency in Mask plane (.5 is
    at NA)
Nx          = N ;                       % Number of transverse cells across mask,
    x
Ny          = Nx ;                      % Number of transverse cells across mask,
```

```
    y - mask is square

%***************************************************************************
%                           INITIALIZE
%***************************************************************************

param           = struct ;          % Simulation parameter struct

%   Imaging Paramteres
param.lambda    = Lambda ;          % Optical wavelength
param.demag     = Demag ;           % Magnification (f_tube / f_obj)
param.NAobj     = NA_ext ;
param.DLfoc     = DLfoc ;
param.fCutOff   = fCutOff ;
param.k0        = 2*pi/Lambda ;
param.defocus   = Delta_Z ;         % Wafer displacement in DISTANCE
param.radio     = Radiometric ;     % Turns on/off Radiometric Corrections
param.aberr     = Aberration ;      % First 9 Sidel Abberration in Waves
param.tiltX     = Illum_Ang(1) ;    % Horizontal illumination tilt
param.tiltY     = Illum_Ang(2) ;    % Vertical illumination tilt
%   Mask Parameters
param.retard    = Retard ;          % mask retardance in radians
param.trans     = 1 ;               % mask transmitance
%   Sim Parameters
param.oversample= Oversample ;      % Image plane interpolation
param.render    = Render ;          % Extra plots on/off

simsize         = struct ;          % Simulation size struct

simsize.dx      = Spacing ;         % Pixel size @ mask
simsize.dy      = Spacing ;         % Square cells
simsize.Nx      = N ;
simsize.Ny      = N ;
simsize.X       = N*simsize.dx ;    % Width in X
simsize.Y       = N*simsize.dy ;    % Width in Y


coord           = struct ;          % empty struct for coordinate set up

% Polarizations
P               = struct ;          % empty struct for polarization set up
EPolarization   = struct ;          % empty struct for polarization of field


%***************************************************************************
%               COORDINATE Set up and POLARIZATION Set up
%***************************************************************************

[ coord ] = CoordSetUp( simsize, coord ) ;
    % returns a struct coord contains[x_vec, y_vec, xgrid, ygrid,dKx, Kx_vec,dKy,
       Ky_vec,KXgrid,KYgrid]

[ EPolarization, P, K_z_EP ] = PolSetUp( param, coord, P, EPolarization ) ;
    % returns a struct EPolarization and struct P.
    % EPolarization contains [ Ord, Ext, ExtEP ]. P contains (K Space Ordinary &
       Extraordinary)
```

```
%***********************************************************************
%                Illuminating E-Field and Mask E-Field
%***********************************************************************

E_inc = struct ;

E_inc.x     = E_illum( 1 ) * ones( Nx, Ny ) ;
E_inc.y     = E_illum( 2 ) * ones( Nx, Ny ) ;

% E-field after mask

[ E_mask ]  = MaskField( param, simsize, coord, E_inc, Mask_Name ) ;  % returns a
    struct E_mask containing [X , Y]

%***********************************************************************
%                                PROPAGATE
%***********************************************************************

[ I_focalplane ] = FourierProp( param, simsize, coord, EPolarization, P, E_mask,
    K_z_EP );
```

D.4 CoordSetUp.m

```
function [ coord ] = CoordSetUp( simsize, coord )

%---------------------------------------------------------------------------
%---------------------------------------------------------------------------
%   CoordSetUp
%   Inputs: simsize, coordinates
%               1)simsize is a struct
%               2)coordinates is an empty struct, to be populated
%   Coordinate set up for real space and Fourier space
%   Returns coord struct
%
%   Author:     D. Miller
%   Written:    ?, 2020 ( poor initial documentaion)
%   Updated:
%
%   Rev History
%   v1      -   Initial Release,
%
%
%---------------------------------------------------------------------------
%---------------------------------------------------------------------------

%------------Real Space Coordinates------------

coord.x_vec     = ( ( -simsize.Nx/2 ) * simsize.dx : simsize.dx : ( simsize.Nx/2-1
    ) * simsize.dx )' ;
coord.y_vec     = ( ( -simsize.Ny/2 ) * simsize.dy : simsize.dy : ( simsize.Ny/2-1
    ) * simsize.dy )' ;

[ XX, YY ]      = meshgrid( coord.x_vec, coord.y_vec ) ;

coord.Xgrid     = XX ;
coord.Ygrid     = YY ;


%------------Fourier Space Coordinates------------

coord.dKx       = 2*pi / ( simsize.X ) ;
coord.Kx_vec    = ( ( -simsize.Nx/2 ) * coord.dKx : coord.dKx : ( simsize.Nx/2-1 )
    * coord.dKx ) ;
coord.dKy       = 2*pi / ( simsize.Y );
coord.Ky_vec    = ( ( -simsize.Ny/2 ) * coord.dKy : coord.dKy : ( simsize.Ny/2-1 )
    * coord.dKy ) ;

[K_XX,K_YY]     = meshgrid( coord.Kx_vec, coord.Ky_vec ) ;

coord.KXgrid    = K_XX ;
coord.KYgrid    = K_YY ;
```

D.5 PolSetUp.m

```
function [ EPolarization , P, K_ZZ_ep ] = PolSetUp( param , coord , P, EPolarization
    )

%-----------------------------------------------------------------------------
%-----------------------------------------------------------------------------
%   PolSetUp
%   Inputs: param , coord , P, EPolarization
%           1) param is a struct
%           2) coord is a struct
%           3) P is an empty struct , to be populated
%           4) EPolarization is an empty struct , to be populated
%
%   Polarization and vector Fourier optics setup:
%   Vector polarizations at mask plane (see McLeod and Wagner , 2014)
%   Spherical coord notation , theta=angle from z, phi = angle from x in xy plane
%   Note that these will be used to calcualte the two amplitudes launched by
%   the mask.  These two fields will be mapped (by the two-lens projection)
%   to a new set of polarizations due to the magnification of the system.
%
%   Original Author:    RRM
%   Original Written:   11/12/15
%   Current Author:     D. Miller
%   Rewritten:          ?, 2020 - poor inital documentation
%   Updated:
%
%   Rev History
%   v1      -   Initial Release , Re-write by D Miller
%
%
%-----------------------------------------------------------------------------
%-----------------------------------------------------------------------------

K_trans      = sqrt( coord.KXgrid.^2 + coord.KYgrid.^2 ) ;    % mag of transverse k
K_ZZ         = sqrt( param.k0^2 - K_trans.^2 ) ;              % z component of k
K            = sqrt( K_trans.^2 + K_ZZ.^2 ) ;                 % mag of k

%   Ordinary Polarization = [-sin(phi),cos(phi),0]
%       sin(phi) = ky/k_trans
%       cos(phi) = kx/k_trans

E_O_mat = cat( 3, -coord.KYgrid ./ K_trans , coord.KXgrid ./ K_trans , zeros( size
    ( coord.KXgrid ))) ;

E_O_mat( coord.Kx_vec == 0, coord.Ky_vec == 0, : ) = [1,0,0] ;       % Define at
    origin , remove div by 0

EPolarization.Ord = E_O_mat ;

%   Extraordinary Polarization = [cos(phi) cos(theta), sin(phi) cos(theta), -sin(
    theta)]
%       cos(theta) = kz/k
%       sin(theta) = k_trans/k

a = ( coord.KXgrid ./ K_trans ) .* ( K_ZZ ./ K ) ;
b = ( coord.KYgrid ./ K_trans ) .* ( K_ZZ ./ K ) ;
```

```
c = ( -K_trans ./ K ) ;

E_E_mat                                   = cat( 3, a, b, c  );

E_E_mat( coord.Kx_vec == 0, coord.Ky_vec == 0, : )  = [ 0, 1, 0 ] ; % Define at
    origin, remove div by 0

EPolarization.Ext = E_E_mat ;

%------------------------------------------------------------------------
%           Render polarizations on k sphere, mask plane
%------------------------------------------------------------------------

if param.render
    [ Nx, ~ ]    = size( coord.KXgrid ) ;
    s            = round( Nx / 20 ) ;        % sparsity for clarity, about 20
        samples
    is           = 1:s:Nx ;                  % sparse indices

    figure;
    quiver3( coord.KXgrid(is,is)/param.k0, coord.KYgrid( is, is ) / param.k0,...
            abs( K_ZZ( is, is )) / param.k0,real( E_O_mat( is, is, 1 )),...
            real( E_O_mat( is, is, 2 )), real( E_O_mat( is, is, 3 ))) ;
    set( gcf, 'color', 'w' ) ; set( gca, 'FontSize', 12 ) ; set( gca, 'FontName',
        'Times' ) ;
    xlabel( 'k_x/k_0' ) ; ylabel( 'k_x/k_0' ) ; zlabel( 'k_z/k_0' ) ;
    title( 'Ordinary polarization, mask and focal plane' ) ;

    figure;
    quiver3( coord.KXgrid( is,is ) / param.k0, coord.KYgrid( is,is )/param.k0,...
            abs( K_ZZ( is, is )) / param.k0, real( E_E_mat( is, is, 1 )),...
            real( E_E_mat( is, is, 2 )), real( E_E_mat( is, is, 3 )));
    set( gcf, 'color', 'w' ) ; set( gca, 'FontSize', 12 ) ; set( gca, 'FontName',
        'Times' ) ;
    xlabel( 'k_x/k_0' ) ; ylabel( 'k_x/k_0' ) ; zlabel( 'k_z/k_0' ) ;
    title( 'Extraordinary polarization, mask plane' ) ;
end

%------------------------------------------------------------------------
%   Vector polarizations at focal plane  (see McLeod and Wagner, 2014)
%   Spherical coord notation, theta=angle from z, phi = angle from x in xy plane
%   Note that these correspond 1:1 to the mask plane except that transverse
    spatial
%   frequencies are larger by a factor of M.  Ordinary polarizations are not
%   impacted, only extraordinary changes.
%
%   K_*_ep refers to the exit pupil
%
%------------------------------------------------------------------------

K_trans_ep  = param.demag * sqrt( coord.KXgrid.^2 + coord.KYgrid.^2 ) ; %
    magnitude of transverse k

K_ZZ_ep     = sqrt( param.k0^2 - K_trans_ep.^2 ) ;                       % z
    component of k

K_ep        = sqrt( K_trans_ep.^2 + K_ZZ_ep.^2 ) ;                       % mag of k
```

```
%-----------------------------------------------------------------------
%   Extraordinary pol = [cos(phi) cos(theta), sin(phi) cos(theta), -sin(theta)];
%-----------------------------------------------------------------------

a2 = param.demag * coord.KXgrid ./ K_trans_ep .* K_ZZ_ep ./ K_ep ;
b2 = param.demag * coord.KYgrid ./ K_trans_ep .* K_ZZ_ep ./ K_ep ;
c2 = -K_trans_ep ./ K_ep ;

E_E_ep                                          = cat( 3, a2, b2, c2 ) ;

E_E_ep( coord.Kx_vec == 0, coord.Ky_vec == 0, : )   = [ 0, 1, 0 ] ;         %
    Define at origin, remove div by 0

EPolarization.ExtEP                             = E_E_ep ;


%   Calculate polarization projection vectors, p (McLeod and Wagner, eq. 17)
%       p1 = e2 x z / (e1 x e2) . z
%       p2 = z x e1 / (e1 x e2) . z
%-----------------------------------------------------------------------

Denom = ( EPolarization.Ord( :,:,1 ) .* EPolarization.Ext( :,:,2 ) - EPolarization
    .Ord( :,:,2 ) .* EPolarization.Ext( :,:,1 )) ;

P.one = cat( 3, EPolarization.Ext( :,:,2 ) ./ Denom, -EPolarization.Ext( :,:,1 )
    ./ Denom, zeros( size( coord.KXgrid ))) ;

P.one( isnan( P.one )) = 0 ;
P.one( isinf( P.one )) = 0 ;

P.two = cat( 3, -EPolarization.Ord( :,:,2 ) ./ Denom, EPolarization.Ord( :,:,1 )
    ./ Denom, zeros( size( coord.KXgrid ))) ;

P.two( isnan( P.two )) = 0 ;
P.two( isinf( P.one )) = 0 ;

%   Render polarizations on k sphere, focal plane
%-----------------------------------------------------------------------

if param.render
    figure ;
    quiver3( param.demag * coord.KXgrid( is, is ) / param.k0, ...
            param.demag * coord.KYgrid( is, is ) / param.k0, ...
            abs( K_ZZ_ep( is, is ) ) / param.k0, real( E_E_ep( is, is, 1 )),...
            real( E_E_ep( is, is, 2 )), real( E_E_ep( is, is, 3 ))) ;
    set( gcf, 'color', 'w' ) ; set( gca, 'FontSize', 12 ); set( gca, 'FontName', '
        Times' ) ;
    xlabel( 'k_x/k_0' ) ; ylabel( 'k_x/k_0' ) ; zlabel( 'k_z/k_0' ) ;
    title( 'Extraordinary polarization, focal plane' ) ;
    xlim( [-1,1] ) ; ylim( [-1,1] ) ; zlim( [0,1] );
end
```

D.6 MaskField.m

```
function [E_mask] = MaskField( param, simsize, coord, E_inc, Mask_Name )
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask
%   Input are arrays of x & y vector components of the illumination
%   phase_delay is the waveplate retardance
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------
E_mask          = struct ;

%load the jones matrix for the mask

switch Mask_Name
    case 'single line'
        jones_matrix = mask_function( param, simsize ) ;
    case 'isolated line'
        jones_matrix = mask_function2( param, simsize ) ;
    case 'grating'
        jones_matrix = mask_function3( param, simsize ) ;
    case 'circle'
        jones_matrix = mask_function4( param, simsize ) ;
    case '2 lines'
        jones_matrix = mask_function5( param, simsize ) ;
    case 'plain circle'
        jones_matrix = mask_function6( param, simsize ) ;
    otherwise
        jones_matrix = mask_custom( param, simsize, Mask_Name ) ;
end


for i = 1:simsize.Ny
    for j = 1:simsize.Nx

        E                  = jones_matrix{ i, j } * [ E_inc.x( i, j ) ; E_inc.y( i
            , j )] ;
        E_maskX( i, j )    = E( 1, 1 ) ;
        E_maskY( i, j )    = E( 2, 1 ) ;

    end
end

Phase_X_tilt    = param.k0 .* coord.Xgrid .* tan( param.tiltX ) ;
Phase_Y_tilt    = param.k0 .* coord.Ygrid .* tan( param.tiltY ) ;

E_mask.X        = cos(param.tiltX) * E_maskX .* exp( -1i * Phase_X_tilt ).* exp(
    -1i * Phase_Y_tilt ) ;      % thin mask approximation with tilted wavefront
E_mask.Y        = cos(param.tiltY) * E_maskY .* exp( -1i * Phase_X_tilt ).* exp(
```

```
    -1i * Phase_Y_tilt ) ;

if param.render
    figure()
    quiver( coord.x_vec(1:16:end) , coord.y_vec(1:16:end) , real( E_mask.X(1:16:
        end,1:16:end )) , real( E_mask.Y(1:16:end,1:16:end )) )
    xlabel('X');
    ylabel('Y');
    title('Polarization after Mask');
end
```

D.7 FourierProp.m

```
function [ I_focalplane ] = FourierProp( param, simsize, coord, EPolarization, P,
    E_mask, K_z_EP )

%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
%   FourierProp
%   Inputs: param, simsize, coord, EPolarization, P, E_mask, K_z_EP
%
%   Vector Fourier propagation
%   Transforms into pupil plane
%   Filters @ pupil
%   Applies defocus to wavefront
%   Applied aberration to wavefront
%   Transforms with magnification and with oversampling onto image plane
%
%   Returns focal plane intensity array
%
%   Original Author:    RRM
%   Original Written:   11/12/15
%   Current Author:     D. Miller
%   Rewritten:          ?, 2020 - poor inital documentation
%   Updated:            ?, 2020 - ongoing poor documentaion
%
%   Rev History
%   v1      -   Initial Release, Re-write by D Miller March 2020
%   v2      -   Updated to include structs
%
%
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------

%   Take FT of scalar amp

E_x_fft     = ifftshift(fft2(fftshift( E_mask.X )));

E_y_fft     = ifftshift(fft2(fftshift( E_mask.Y )));

%   Use (psuedo)projection vectors to find amplitudes of each polarization mode

E_1_fft     = E_x_fft .* P.one(:,:,1) + E_y_fft .* P.one(:,:,2);

E_1_fft( isnan(E_1_fft)) = 0 ;

E_1_fft( isinf(E_1_fft)) = 0 ;

E_2_fft     = E_x_fft .* P.two(:,:,1) + E_y_fft .* P.two(:,:,2);

E_2_fft(isnan(E_2_fft)) = 0 ;

E_2_fft(isinf(E_2_fft)) = 0 ;

%   Apply pupil plane filter

[ E_1_filter, E_2_filter ]  = PupilFilter( E_1_fft, E_2_fft, param, coord, 'disk'
    ) ;
```

```
%   Apply Wafer Defocus and Aberrations

[ E_1_defoc, E_2_defoc ]    = Defocus( E_1_filter, E_2_filter, param, coord,
    K_z_EP ) ;

[ E_1_aberr, E_2_aberr ]    = Aberrations( E_1_defoc, E_2_defoc, param, coord,
    K_z_EP ) ;


% Apply Radiometric Correction

if param.radio == 1

    [ E_1, E_2 ] = RadiometricCorrection( E_1_aberr, E_2_aberr, param, coord ) ;

else
    E_1 = E_1_aberr ;
    E_2 = E_2_aberr ;

end

%Transform to Image Plane

E_1_focalplane          = cat( 3, ...
            ifftshift( ifft2( fftshift( E_1 .* EPolarization.Ord( :,:,1 )), param.
                oversample * simsize.Nx, param.oversample * simsize.Ny )), ...
            ifftshift( ifft2( fftshift( E_1 .* EPolarization.Ord( :,:,2 )), param.
                oversample * simsize.Nx, param.oversample * simsize.Ny )), ...
            ifftshift( ifft2( fftshift( E_1 .* EPolarization.Ord( :,:,3 )), param.
                oversample * simsize.Nx, param.oversample * simsize.Ny ))) ;

% Polarization 2 **Note use of exit pupil projection**
E_2_focalplane      = cat( 3, ...
            ifftshift( ifft2( fftshift( E_2 .* EPolarization.ExtEP( :,:,1 )),
                param.oversample * simsize.Nx, param.oversample * simsize.Ny )),
                ...
            ifftshift( ifft2( fftshift( E_2 .* EPolarization.ExtEP( :,:,2 )),
                param.oversample * simsize.Nx, param.oversample * simsize.Ny )),
                ...
            ifftshift( ifft2( fftshift( E_2 .* EPolarization.ExtEP( :,:,3 )),
                param.oversample * simsize.Nx, param.oversample * simsize.Ny ))) ;

% Coherently combine E1 and E2 @ focal plane
I_focalplane    = abs( E_1_focalplane( :,:,1 ) + E_2_focalplane( :,:,1 )).^2 + ...
              abs( E_1_focalplane( :,:,2 ) + E_2_focalplane( :,:,2 )).^2 + ...
              abs( E_1_focalplane( :,:,3 ) + E_2_focalplane( :,:,3 )).^2;

%***********************************************************************
%                       Render fields in Fourier plane
%***********************************************************************

if param.render
    figure ;

    % RRM Changes
    % abs( E_1 ) not abs(E_1 / max...  Probably no difference
    % coord.KYgrid instead of coord.Ygrid in following line
```

```
    % Added reporting of max value to titles
    MyPcolor( param.demag * coord.KXgrid / param.k0, param.demag * coord.KYgrid /
        param.k0, log10( abs( E_1 ) / max( max( abs( E_1 ))))...
        ,'hot','flat','Focus k_x/k_0','Focus k_y/k_0',['log_{10}(|E_1|), max = ',
            num2str(max( max( abs( E_1 ))))]) ;
    colorbar ; caxis( [ -2, 0 ]) ;
    xlim( 2 * param.NAobj * [ -1, 1 ]); ylim( 2 * param.NAobj * [ -1, 1 ]) ;
    hold on;
    %drawcircle( [ 0, 0 ], param.demag * 2 * pi * param.fCutOff / param.k0, 100, '
        b.', [ 0, 2 * pi ]);      % Gaussian filter cuttoff
    drawcircle( [ 0, 0 ], param.NAobj,100, 'g-', [ 0, 2 * pi ]) ;
                         % Lens aperture
    legend( 'Cutoff filter', 'Lens aperture' ) ;

    figure ;
    MyPcolor( param.demag * coord.KXgrid / param.k0, param.demag * coord.KYgrid /
        param.k0, log10( abs( E_2 ) / max( max( abs( E_2 ))))...
        ,'hot', 'flat', 'Focus k_x/k_0', 'Focus k_y/k_0', ['log_{10}(|E_2|), max =
             ',num2str(max( max( abs( E_2 ))))]) ;
    colorbar ; caxis( [ -2, 0 ]) ;
    xlim( 2 * param.NAobj * [ -1, 1 ]) ; ylim( 2 * param.NAobj * [ -1, 1 ]) ;
    hold on ;
    %drawcircle([0,0],M*2*pi*fCutOff/k0,100,'b.',[0,2*pi]);      % Gaussian filter
         cuttoff
    drawcircle( [ 0, 0 ], param.NAobj,100, 'g-', [ 0, 2*pi ]);                  %
        Lens aperture
    legend( 'Cutoff filter', 'Lens aperture' ) ;
end
```

D.8 PupilFilter.m

```
function [ E_1_filter , E_2_filter ] = PupilFilter( E_1_fft , E_2_fft , param , coord
    , filter_type )

%---------------------------------------------------------------------
%---------------------------------------------------------------------
%   Pupil Filter
%   Inputs are
%
%   Author:      D. Miller
%   Written:     May 15, 2020
%   Updated:
%
%   Rev History
%   v1     -    Initial Release
%
%---------------------------------------------------------------------
%---------------------------------------------------------------------
switch filter_type
    case 'disk'
        k_trans      = param.demag * sqrt( coord.KXgrid.^2 + coord.KYgrid.^2 ) ; %
            Magnitude of transverse k

        Filter       = ones( size( E_1_fft )) ;
        Filter       = ( k_trans <= param.NAobj*param.k0 ) ;                    %
            Back aperture of objective

        E_1_filter   = E_1_fft .* Filter ;
        E_2_filter   = E_2_fft .* Filter ;

    case 'gaussian'
        % Define Gaussian Fourier filter, specified by 1/e cutoff frequency
        % including NA limit of objective

end
```

D.9 Defocus.m

```
function [ E_1_defoc , E_2_defoc] = Defocus( E_1, E_2, param, coord, K_z_EP )

%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Aberrations
%   Inputs : E_1_filter, E_2_filter, param.aberr, K_z_EP
%
%   Applies defocus due to wafer (image plane) displacement
%
%   Author:     D. Miller
%   Written:    May 14, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------

E_1_defoc = E_1 .* exp( -1 * 2 * pi * param.defocus * K_z_EP ) ;

E_2_defoc = E_2 .* exp( -1 * 2 * pi * param.defocus * K_z_EP ) ;
```

D.10 Aberrations.m

```
function [ E_1_aberr , E_2_aberr] = Aberrations( E_1, E_2, param, coord, K_z_EP )

%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Aberrations
%   Inputs : E_1_filter, E_2_filter, param.aberr, K_z_EP
%
%   Applies wavefront aberrations in pupil plane
%   Includes:   Piston, X-Tilt, Y-Tilt, Defocus, 0 deg Astigmatism, 45 deg
%               Astigmatism, X-Coma, Y-Coma, 3rd order Spherical
%
%   Author:     D. Miller
%   Written:    April 8, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------

% Coefficients are in WAVES

% Piston
a0 = param.aberr( 1 ) ;


% X-Tilt
```

```
a1 = param.aberr( 2 ) ;


% Y-Tilt
a2 = param.aberr( 3 ) ;


% Defocus
a3 = param.aberr( 4 ) ;


% 0 deg Astigmatism
a4 = param.aberr( 5 ) ;


% 45 deg Astigmatism
a5 = param.aberr( 6 ) ;


% X Coma
a6 = param.aberr( 7 ) ;


% Y Coma
a7 = param.aberr( 8 ) ;


% Spherical
a8 = param.aberr( 9 ) ;



%Coefficients = [ a0 a1 a2 a3 a4 a5 a6 a7 a8 ] ;

%Zernike Polynomials = (

Z0 = 1 ;
Z1 = 2 * coord.KYgrid ;
Z2 = 2 * coord.KXgrid ;
Z3 = sqrt( 3 ) * ( 2 * coord.KXgrid.^2 + 2 * coord.KYgrid.^2 - 1 ) ;
Z4 = sqrt( 6 ) * ( coord.KXgrid.^2 + coord.KYgrid.^2 ) ;
Z5 = 2 * sqrt( 6 ) * coord.KXgrid .* coord.KYgrid ;
Z6 = sqrt( 8 ) * ( 3 * coord.KXgrid.^3 + 3 * coord.KXgrid .* coord.KYgrid.^2 - 2 *
     coord.KXgrid ) ;
Z7 = sqrt( 8 ) * ( 3 * coord.KXgrid.^2. .* coord.KYgrid + 3 * coord.KYgrid.^3 - 2
    * coord.KYgrid ) ;
Z8 = sqrt( 5 ) * ( 6 * coord.KXgrid.^4 + 12 * coord.KXgrid.^2 .* coord.KYgrid.^2 +
     6 * coord.KYgrid.^4 - 6 * coord.KYgrid.^2 + 1 ) ;


phase = 2 * pi *( a0*Z0 + a1*Z1 + a2 * Z2 + a3 * Z3 + a4 * Z4 + a5 * Z5 + a6 * Z6
    + a7 * Z7 + a8 * Z8 ) ;

E_1_aberr = E_1 .* exp( -1i * phase ) ;

E_2_aberr = E_2 .* exp( -1i * phase );
```

D.11 RadiometricCorrection.m

```
function [ E_1_corrected , E_2_corrected ] = RadiometricCorrection( E_1, E_2,
    param, coord )

%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Radiometric Correction for Intensity after magnification
%   Inputs are
%
%   Author:     D. Miller
%   Written:    May 15, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%--------------------------------------------------------------------
%--------------------------------------------------------------------

k_trans                         = param.demag * sqrt( coord.KXgrid.^2 + coord.
    KYgrid.^2 ) ;

sin_theta                       = k_trans/param.k0;

sin_theta(sin_theta==1)         = nan ;

sin_theta(sin_theta==-1)        = nan ;

Correction                      = (( 1 - sin_theta.^2 / param.demag^2 ) ./ ( 1 -
    sin_theta.^2 )) .^0.25 ;

Correction(isnan(Correction))   = 1/param.demag^2;

Correction(isinf(Correction))   = 0;

E_1_corrected                   = E_1 .* Correction ;

E_2_corrected                   = E_2 .* Correction ;
```

D.12 MyPcolor.m

```
% ************************************************************************
% MyPcolor:  pcolor plot with common bells and whistles.
% MyPcolor(x,y,z,cmap,shade,xlab,ylab,plotlab)
% Code from R. McLeod
% ************************************************************************
function h = MyPcolor(x,y,z,cmap,shade,xlab,ylab,plotlab,fontsize)

% Default font size = 12
if nargin < 9,
    fontsize=12;
end

h = pcolor(x,y,z);
shading(shade);
colormap(cmap);
set(gcf,'color','w');
set(gca,'FontSize',fontsize);
set(gca,'FontName','Times');
xlabel(xlab);
ylabel(ylab);
title(plotlab);
```

D.13 mask_function.m

```
function jones_array = mask_function( param , simsize )
%---------------------------------------------------------------------
%---------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%---------------------------------------------------------------------
%---------------------------------------------------------------------

orientation_matrix                          = zeros( simsize.Nx, simsize.Ny );
        %waveplate orientations

orientation_matrix ( : , 1:simsize.Nx/2 )   = pi/2 ;

jones_array                                 = waveplate_matrix( param.retard,
   orientation_matrix ) ;
```

D.14 mask_function2.m

```
function jones_array = mask_function2( param, simsize )
%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%--------------------------------------------------------------------
%--------------------------------------------------------------------

orientation_matrix  = zeros( simsize.Nx, simsize.Ny );      %waveplate
    orientations

line_length         = 50;   % percentage of total sim width

frac               = (100-line_length)/2/100;

start               = simsize.Nx * frac ;
stop                = simsize.Nx - simsize.Nx * frac;

orientation_matrix ( start:stop , 1:simsize.Nx/2 )    = pi/2 ;

orientation_matrix ( 1:start , : )    = pi/4 ;
orientation_matrix ( stop:end, : )    = pi/4 ;

jones_array = waveplate_matrix( param.retard , orientation_matrix);
```

D.15 mask_function3.m

```
function jones_array = mask_function3( param , simsize )
%-----------------------------------------------------------------------
%-----------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-----------------------------------------------------------------------
%-----------------------------------------------------------------------

orientation_matrix  = zeros( simsize.Nx, simsize.Ny ) ;     %waveplate
   orientations

grating_halfpitch   = 5*10^-6 ;

pixel_count         = ceil( grating_halfpitch / simsize.dx ) ;

j = 0 ;                                                     % initialize counter

for i=1:simsize.Nx
    j=j+1 ;
    if j <= pixel_count
        orientation_matrix ( : , i ) = pi/2 ;
    elseif j > 2*pixel_count
        j = 0;
    end
end

jones_array = waveplate_matrix( param.retard , orientation_matrix);
```

D.16 mask_function4.m

```
function jones_array = mask_function4( param , simsize )
%--------------------------------------------------------------------
%--------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask (CIRCLE)
%   PATTERN)
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    May 18, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%--------------------------------------------------------------------
%--------------------------------------------------------------------

radius          = 0.50 ;                    %width in percent of field size

x_coord         = ( - simsize.Nx / 2 : 1 : simsize.Nx/2 - 1 ) ;

y_coord         = ( - simsize.Ny / 2 : 1 : simsize.Ny/2 - 1 ) ;

[ XX, YY ]      = meshgrid( x_coord, y_coord ) ;

[ theta, rho ]  = cart2pol( XX, YY ) ;

theta           = - theta / 2 ;

rho_max         = max(max(rho)) ;

for i = 1 : simsize.Nx
    for j = 1: simsize.Ny
        if rho( i, j ) > rho_max * radius
            orientation_matrix( i, j ) = theta( i, j ) - pi / 2 ;
        else
            orientation_matrix( i, j ) = theta( i, j );
        end
    end
end

jones_array         = waveplate_matrix( param.retard, orientation_matrix);
```

D.17 mask_function5.m

```
function jones_array = mask_function5( param , simsize )
%-----------------------------------------------------------------
%-----------------------------------------------------------------
%   2 Line Mask
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-----------------------------------------------------------------
%-----------------------------------------------------------------

line_separation = .08*9 ;     % microns

n                                               = ceil( ( line_separation *10^-6 /
     simsize.dx ) / 2 ) ;

orientation_matrix                              = zeros( simsize.Nx, simsize.Ny );

orientation_matrix ( : , 1:simsize.Nx/2-n )     = pi / 2 ;

orientation_matrix ( : , simsize.Nx/2+n:end )   = pi / 2 ;

jones_array                                     = waveplate_matrix( param.retard,
    orientation_matrix ) ;
```

D.18 mask_function6.m

```
function jones_array = mask_function6( param , simsize )
%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   Circle Phase mask, no polarization control
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of Jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------

radius          = 0.50 ;                            %width in percent of field
    size

x_coord         = ( - simsize.Nx / 2 : 1 : simsize.Nx/2 - 1 ) ;

y_coord         = ( - simsize.Ny / 2 : 1 : simsize.Ny/2 - 1 ) ;

[ XX, YY ]      = meshgrid( x_coord, y_coord ) ;

[ ~, rho ]      = cart2pol( XX, YY ) ;

rho_max         = max(max(rho)) ;

orientation_matrix = zeros(simsize.Nx,simsize.Ny);

for i = 1 : simsize.Nx
    for j = 1: simsize.Ny
        if rho( i, j ) > rho_max * radius
            orientation_matrix( i, j ) = 0 ;
        else
            orientation_matrix( i, j ) = pi/2;
        end
    end
end

jones_array         = waveplate_matrix( param.retard, orientation_matrix);
```

D.19 mask_custom.m

```
function jones_array = mask_custom( param, simsize, mask_name )
%----------------------------------------------------------------------
%----------------------------------------------------------------------
%   Computes the output E-field after pixelated waveplate mask
%   Input is phase_delay, the waveplate retardance
%   Output is a cell array of jones matrices for each pixel
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%----------------------------------------------------------------------
%----------------------------------------------------------------------

load( strcat( mask_name, '.mat' )) ;

orientation_matrix  = kron(mask,ones( 16 )) ;   %UPDATE AS NEEDED

jones_array         = waveplate_matrix( param.retard, orientation_matrix) ;
```

D.20 waveplate_matrix.m

```
function jones_matrix = waveplate_matrix( phase_delay, fast_axis_angle )
%----------------------------------------------------------------------
%----------------------------------------------------------------------
%   Jones Matrix for a waveplate with arb retardance and orientation
%   waveplate_matrix(phase_delay, fast_axis_angle)
%
%   Author:     D. Miller
%   Written:    March 26, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%----------------------------------------------------------------------
%----------------------------------------------------------------------

theta           = fast_axis_angle ;

phi             = phase_delay ;

[ Ny, Nx ]      = size( theta ) ;

rotation_matrix = @(x) [ cos(x) -sin(x) ; sin(x) cos(x) ] ;

retarder_matrix = [ 1 0 ; 0 exp(-1i * phi) ] ;

for i = 1:Ny
    for j = 1:Nx
```

```
        jones_matrix{ i,j } = rotation_matrix(-theta(i,j)) * retarder_matrix *
            rotation_matrix(theta(i,j));
    end
end
```

D.21 OpticalContrast.m

```
function [ m ] = OpticalContrast( Intensity , N )
%---------------------------------------------------------------------
%---------------------------------------------------------------------
%   Computes the contrast of single isolated line in image plane
%   Input 2D intensity distribution, grid size N
%
%   Author:     D. Miller
%   Written:    May 15, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%---------------------------------------------------------------------
%---------------------------------------------------------------------

low     = min( Intensity( N / 2, : )) ;
high    = mean(Intensity( N / 2, : )) ;

m       = ( high - low ) / ( high + low );
```

D.22 Resist.m

```
function [profile] = Resist( intensity, D_high, gamma )
%---------------------------------------------------------------------
%---------------------------------------------------------------------
%   Computes the resist profile after exposure
%   Inputs are intensity array, upper dose threshold, contrast
%
%   Author:     D. Miller
%   Written:    March 31, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%---------------------------------------------------------------------
%---------------------------------------------------------------------

D_low = 10^( log10( D_high ) - 1/gamma ) ;                      % find lower dose
    threshold

profile = -gamma * log10(intensity) + gamma * log10(D_high);    % compute
    intermediate dose regions

profile( intensity <= D_high ) = 1 ;                            % regions below
    threshold - all remain
```

```
profile( intensity >= D_high ) = 0 ;                        % regions above
    threshold - all removed

% clear border artifacts

profile( 1, : )     = 0 ;

profile( end, :)    = 0 ;

profile_no_border   = imclearborder( profile , 8 ) ;
```

D.23 MeasureLinewidth.m

```
function [ line_width ] = MeasureLinewidth( profile, coord, N )
%----------------------------------------------------------------------
%----------------------------------------------------------------------
%   Computes the width of single isolated line in resist profile
%   Input 2D resist profile, grid size N, and coords struct
%
%   Author:     D. Miller
%   Written:    May 15, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%----------------------------------------------------------------------
%----------------------------------------------------------------------

X_section       = profile( N / 2 , : ) ;

line_location   = find( X_section > 0 ) ;           % positions where thickness
    exceeds zero

line_start      = coord.x_vec( line_location( 1 )) ;                % 1st edge
line_end        = coord.x_vec( line_location( length( line_location ))) ; % 2nd
    edge


line_width      = ( line_end - line_start ) ;
```

D.24 SpatialCoherence.m

```
function [ coords , I_fp_out ] = SpatialCoherence( file_name )

%-------------------------------------------------------------------
%-------------------------------------------------------------------
%   SpatialCoherence
%   Inputs : file name for stored input parameters
%
%   Computes image for partial spatial coherence
%   Images computed over a range of incident angles
%   Each adds incoherently
%
%   Author:     D. Miller
%   Written:    May 3, 2020
%   Updated:
%
%   Rev History
%   v1    -   Initial Release
%
%-------------------------------------------------------------------
%-------------------------------------------------------------------

load( file_name ) ;

NA_illum    = NA_ent * Sigma ;

resolution  = 0.01 ;

Theta       = ( -asin( NA_illum ) : resolution : asin( NA_illum )) ;

ThetaX      = Theta ;
%ThetaY      = Theta ;

I_fp = zeros( N, N, length(Theta) );

for i = 1:length(Theta)
        Illum_Ang = [ ThetaX(i), 0 ] ;

        file = strcat( 'RunSettingSigma', num2str( i ) ) ;

        save( file )
end

for i = 1:length(Theta)
    %for j = 1:length(Theta)

    [ coords , I_fp_temp ]   = ProjectionCode( strcat( 'RunSettingSigma', num2str(
        i ) ) ) ;

    I_fp(:,:,i)         = I_fp_temp ;

    %end
end

I_total     = sum( I_fp, 3 ) ;
I_fp_out    = I_total / length(Theta) ;
```

D.25 mask_solver.m

```
%Laplace Equation Based Mask Designer

niter = 50000;                              % number of iterations
%set size

Nx = 64;
Ny = 64;

%***************************for single line********************************

% boundary_mask                  = ones( Nx,Ny ) ;   % multiplied after each
    itteration to reset boundary
%
% boundary_mask( :,255 )         = 0 ;
% boundary_mask( :,256 )         = 0 ;
%
% boundary_condition             = zeros( Nx,Ny ) ;       % added after each
    itteration to reset boundary
% boundary_condition( :,255 )    = 0 ;
% boundary_condition(:,256 )     = pi/2 ;

%--------------------------for line segment--------------------------------

% line_length          = 50 ;   % percentage of total sim width
%
% frac                 = (100-line_length)/2/100;
%
% start                = Nx * frac ;
% stop                 = Nx - Nx * frac;
%
% boundary_mask        = ones( Nx,Ny ) ;
%
% boundary_mask( start:stop, 31 ) = 0 ;
% boundary_mask( start:stop, 32 ) = 0 ;
%
% boundary_condition             = zeros( Nx,Ny ) ;
% boundary_condition( start:stop, 31 )    = 0 ;
% boundary_condition( start:stop, 32 )    = pi/2 ;

%--------------------------for rectangular box-----------------------------

% load('box_mask_data.mat');
%
% mask                 = zeros(Nx,Ny) ;    %rand(Nx,Ny);
%
% mask( 1, 2 : end )   = - pi/4 ;           % bottom edge
%
% mask( : , end )      = pi /2 ;            % right edge
%
% mask( end, 1:end )   = pi / 4 ;           % top edge

% %--------------------------for 3 line mask-----------------------------
%
% load('box_mask_data.mat');
%
```

```
% mask                 = zeros(Nx,Ny) ;    %rand(Nx,Ny);
%
% mask( : , 1 )        = 0 ;               %left edge
%
% mask( 1, : )         = 3*pi/4 ;            % bottom edge
%
% mask( : , end )      = pi /2 ;           % right edge
%
% mask( end, 1:end )   = pi / 4 ;          % top edge
%
%
% i=2:Ny-1;
% j=2:Nx-1;
%
% for k=1:niter
%     mask(i,j) = (( mask( i+1,j ) + mask( i-1,j ) )+( mask( i,j+1 ) + mask( i,j-
    1))) / 4;
%
%     mask = mask .* boundary_mask ;
%     mask = mask + boundary_condition ;
%
% end
%
% x = (1:Nx) ;
% y = (1:Ny) ;
%
% x_orient = cos(mask);
% y_orient = sin(mask);
%
% figure;
% quiver( x, y, x_orient, y_orient)
%
% save('box.mat','mask')

% %-------------------------for 3 line mask----------------------------

load('T-mask_data4.mat');

mask                 = zeros(Nx,Ny) ;    %rand(Nx,Ny);

i=2:Ny-1;
j=2:Nx-1;

for k=1:niter
    mask(i,j) = (( mask( i+1,j ) + mask( i-1,j ) )+( mask( i,j+1 ) + mask( i,j- 1)
        )) / 4;

    mask = mask .* boundary_mask ;
    mask = mask + boundary_condition ;

end

x = (1:Nx) ;
y = (1:Ny) ;

x_orient = cos(mask);
y_orient = sin(mask);
```

```
figure;
quiver( x, y, x_orient , y_orient)

save('t_mask4.mat','mask')
```