Pseudo Three-Dimensional MEMS Imaging using Reflectance Transformation Imaging (RTI) Technique

By

Sahar Ahmadi

Submitted in partial fulfillment of the requirements
For the degree of Master of Applied Science

at

Dalhousie University
Halifax, Nova Scotia
December 2021

Table of Contents

List of Figures

Abstract

The study of MEMS devices is highly dependent on the information that imaging techniques can provide. There are two main approaches to observe MEMS devices: Optical and non-optical imaging techniques. Although each of the techniques has its own advantages and disadvantages, optical microscopy is of great interest due to its low-cost, accessibility, resolution and the colored images it provides. A critical shortcoming of optical microscopy is the strict viewing direction and lighting angle that result in flat images with no visual representation of MEMS 3D structure that may lead to some confusion in terms of interpreting the topography of MEMS surfaces and their out-of-plane displacements.

This thesis aims at enhancing the 2D representation of MEMS devices by capturing a sequence of images at different illumination angles and highlighting the lateral facets, based on the RTI imaging technique, and then creating a sequence of post-processed images. The proposed algorithm offers a rapid, non-invasive, cost-effective and non-contact imaging technique that aims at minimizing the necessity of using other imaging approaches by revealing the 3D aspects of MEMS devices in terms of 2D visualization.

List of Abbreviations of Symbols Used

2D	2-dimensional
3D	3-dimensional
DOF	Depth of Field
GND	Ground
HF	Hydrofluoric acid
HSV	Hue, Saturation, Value
IDE	Integrated Development Environment
L	Incident Light Direction
LED	Light Emitting Diode
MEMS	Microelectromechanical Systems
MP	Megapixel
MUMPs	Multi-User MEMS Processing system
N	Surface Normal
NA	Numerical Aperture
OXIDE	Silicon Dioxide
PGA	Pin Grid Array
PLA	Polylactic Acid
POLY	Polysilicon
PTM	Polynomial Texture Mapping
R	Reflected Light Direction
RGB	Red, Green, Blue
RTI	Reflectance Transformation Imaging
SEM	Scanning Electron Microscopy
USB	Universal Serial Bus
ZIF	Zero Insertion Force

1. Introduction

1.1. Conventional Imaging Approaches

Optical microscopes have been developed for investigating microstructures not visible to the naked eye using a set of lenses and an illumination system. In addition to the binocular tubes to the eyepieces, to photograph specimens and permanently document information microscopes can be equipped with an attached camera monocular tube as depicted in Figure 1.

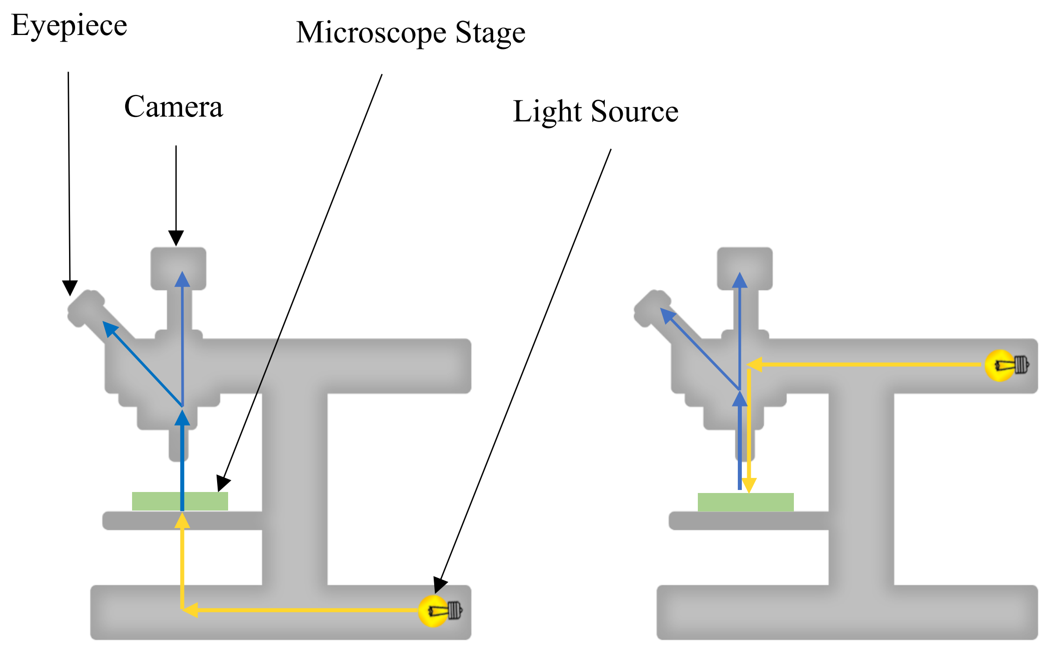


Figure 1: Cutaway views of two types of illumination employed in optical microscope. Left: transmitted illumination. Right: in-line illumination.

Most optical microscopes use one of the two standard lighting arrangements to illuminate the sample: transmitted illumination and in-line reflected illumination. The paths and directions of light through the microscope starting from the light source for both illumination arrangements are depicted in Figure 1. For transmitted illumination (Figure 1 (left)) the light source is below the microscope. The light rays pass though the specimen (yellow arrows) and subsequently enter the objective (blue arrows). The light entering the objective strikes an optical beam splitter which is a mirror-like piece with a coating with partial reflectance transparency. Thereby, the beam splitter splits the incident beam into two separate beams (typically 50/50), one travels to the camera, one to the eyepieces. With transmitted illumination, only transparent or translucent specimens can be observed (for example cells or biological samples).

In-line reflected illumination (also called co-axial or vertical illumination) can be used to observe opaque specimens. Typically, this type of illumination is used for non-biological microscopes and inspection microscopes. With in-line illumination (Figure 1 (right)) the surface of the specimen is illuminated by light that has passed coaxially through the objective (yellow arrows) via a second beam splitter. The incident light is reflected from the specimen and re-enters the objective (blue arrows). The optical pathway reaches completion by being directed towards the camera and/or eyepieces. Note that as the light must pass through both beam splitters twice, image intensity is reduced by a factor of four.

1.1.1. Optical microscope MEMS Imaging Approaches

Micro Electro Mechanical Systems (MEMS) incorporate miniature mechanical and electrical components in the range 1 μm to 100 μm to form three-dimensional configurations that can generate motion in the x-y plane as well as along the z-axis (more detail will be provided in Chapter 5). For instance, Figure 2 shows a MEMS actuation assembly consisting of a suspended central metal platform that can move along the x-axis and y-axis by the force generated by the two central chevron thermal actuators. There is also an out-of-plane motion created by the two symmetrically placed vertical thermal actuators that deflect the suspended rectangular mass upward.

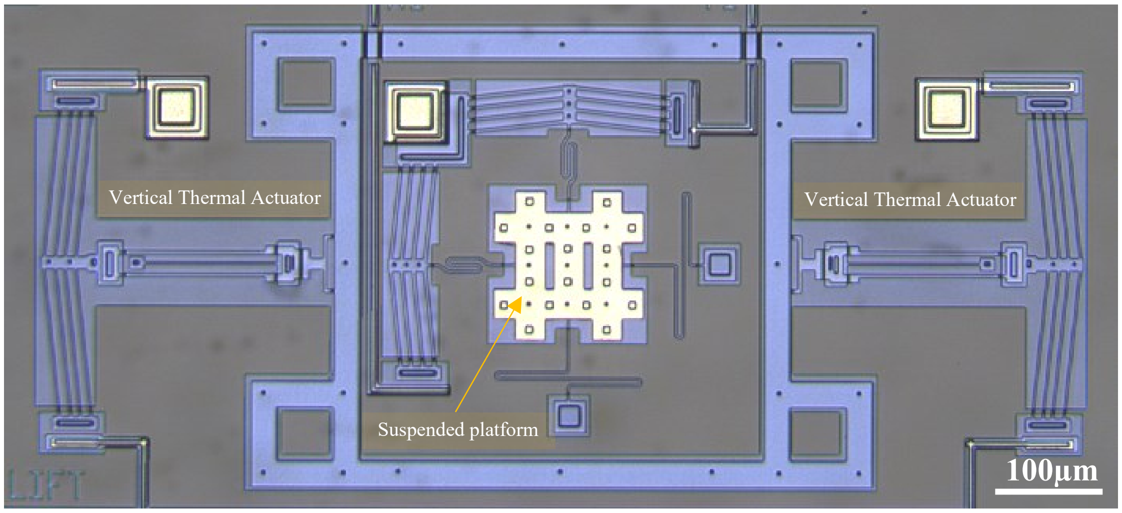


Figure 2: An electrothermal MEMS micro-actuator assembly that can generate in-plane and out-of-plane deflections.

The quality of captured photomicrographs taken from MEMS devices relies on the quality of the microscope and the camera. Optical microscopes with high magnification equipped with high resolution digital cameras deliver a magnified image of MEMS devices. Typical fields of views are several hundred microns, with pixel sizes in the sub µm range. Resolution is limited by the Rayleigh criteria which is generally defined using a standard formula as [1]:

$$R_{Rayleigh} = \frac{1.22\,\lambda}{2NA} \tag{1}$$

Where λ is the wavelength of light in the range of 0.4 µm to 0.7 µm, and NA is the numerical aperture. Numerical aperture indicates the extent of the angle at which light enters the objective, in our case for a 25× objective NA = 0.31, and so in this case the Rayleigh criterion limits the resolution to around 1 µm. However, due to the conservative nature of the Rayleigh criterion, image resolution in modern microscope setups can slightly surpass the Rayleigh resolution [2], [3].

In-line illumination is one of the most widely used methods of observing Micro-Electro-Mechanical Systems (MEMS) in a microscope and this is for two reasons. Firstly, almost all MEMS are opaque. Secondly, most MEMS surfaces are optically smooth and highly reflective. This type of illumination is also called brightfield as flat smooth objects (such as most MEMS backgrounds) appear as bright regions.

Another type of illumination used in microscopy (but rarely used in MEMS) is dark-field illumination. As depicted in Figure 3, dark-field illumination is a reflectance-based method of imaging that illuminates the sample from the side, rather than in-line. For example, by using a ring of shallow light sources. In dark-field illumination the polar angle (the angle from the vertical axis) of the illumination is fixed, and the sample is illuminated from different azimuth angles. The images obtained by dark-field illumination are composed of indirect light rays that have been reflected from surface features towards the objective. Because of the shallow light angle, flat targets do not reflect light back into the objective and hence appear dark, but topographical features such as defects or slopes stand out. Therefore, the image displays surface anomalies against a dark background leading to high contrast images. However, the brightness of the final images obtained through dark field illumination can be low due to the narrow ring of light that strikes the specimen [4].

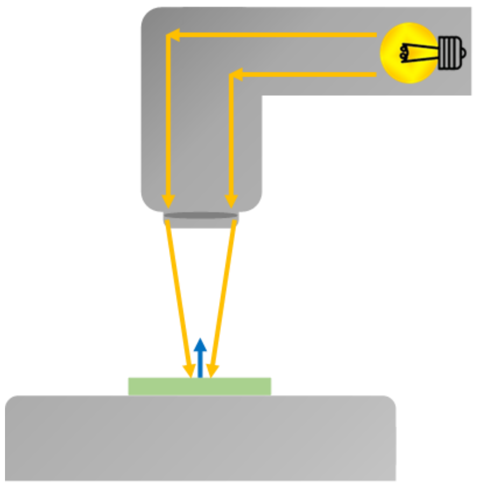


Figure 3: Cutaway view of darkfield illumination. Only oblique light rays travel towards the sample.

Even though some microscopes have binocular eyepieces, both images are in fact identical, since by definition single objective microscopes only show one viewpoint. While there are true binocular inspection microscopes, they operate at far lower magnifications than required for MEMS. A single-angle viewpoint reduces our perception of the real three-dimensional structure. Given this limitation, how else can three-dimensional information be deduced? The variations in reflected light intensity in vertical illumination can reveal approximate topographical features [5], but restricted shadows and highlights make it difficult to differentiate between bumps and depressions, and difficult to discern displacements along the z-axis, inclination variations, and vertical sidewalls.

Observing the MEMS components from another fixed viewpoint is possible by tilting the chip [6].This viewpoint yields oblique images of the MEMS components; however, only a narrow horizontal band of the image is in focus, typically less than 10% of the image. This is due to the restricted Depth of Field (DOF) of optical microscopy, especially at higher magnifications. Depth of Field specifies the axial distance between the nearest and farthest points of the image that are entirely focused. DOF is determined by the following equation:

$$DOF \approx \frac{n\,\lambda}{NA^2} \tag{2}$$

Where *n* is the refractive index of air (n=1), λ is the wavelength of light illuminating the chip surface, and NA is the numerical aperture. The light source wavelength is in the range of 0.4 μm and 0.7 μm, hence the highest depth of field can be estimated (actual DOF is typically smaller):

$$DOF \approx \frac{1 \times 0.7}{0.31^2} \cong 5.8\ \mu\text{m} \tag{3}$$

By comparison, for a 100 x 100 μm MEMS device tilted at 45°, the nearest and farthest points of the image would be ~± 35 μm axially from the center and thus far out of focus. Therefore, while observing the MEMS chip from an angle provides information about the three-dimensional structures and out-of-plane movements, recorded images are mostly out of focus and topographical features of the surface remain obscure. This study investigates the specimen from a fixed viewpoint but using multi-directional illumination sources.

1.1.2. Non-Optical Microscope MEMS Imaging Alternatives

Microscopes are not the only means of representing the topography of three-dimensional geometries in the digital world. There are a variety of technologies generating a detailed representation of three-dimensional surfaces in high resolution. However, as we shall see, these would either have limitations when used with MEMS, or would simply not be possible at the MEMS scale. We will examine 3 technologies: Scanning Electron Microscope, 3D scanners and Photogrammetry.

One of the most powerful imaging technologies in MEMS research is Scanning Electron Microscopy (SEM). Rather than photons in optical microscopy capturing features of the specimen surface, focused beams of electrons magnify the object's surface. The interaction between electrons and specimen surface in a vacuum, where electrons directly travel towards the sample without being deflected with air particles, results in detailed grayscale images of the specimen. SEM offer large depth of field and high resolution (nm) and produces three-dimensional images. However, the requirements for this approach are limiting [7]:

1. Cost of booking SEM time
2. Requirement that chips be imaged outside of the MEMS lab
3. Requirement that samples be prepared and mounted for SEM viewing
4. Requirement that samples be in vacuum
5. Difficulty in powering and observing MEMS chips within SEM

Another imaging technology is 3D Laser Scanners. These devices are equipped with a laser probe sweep a 3D structure entirely, while two cameras in two different positions record the point-by-point distance alterations in laser beam at any given moment. This yields the digital 3D shape of the investigated object as a set of points called a “Point Cloud” [8]. However, the accuracy of the laser scanners is in the range of millimeters, and thus are not able to capture components of MEMS devices. In addition, specimens with reflective surfaces including MEMS and many industrial metal objects are not easily 3D scanned. These shiny surfaces reflect the incident laser beam, and the spurious reflected beams interfere with the camera collecting the laser beams from the surface (see [9]).

A third imaging technology is photogrammetry, a simple and low-cost method for measurement of complex surface properties with no physical contact [10]. The principal concept underlying photogrammetry is observing from multiple viewpoints located in various angles. High-quality photographs from different perspectives around the objects are recorded, and then a mesh or point cloud of the 3D object is constructed through commercial photogrammetry software [11]. The software combines all the captured images and aligns them together. The accurate location of each point in the 3D model is found by specifying geometric intersections of images, as well as the camera parameters [10]. Photogrammetry often requires a camera mounted on a rotating stage to take multiple images around the specimen. MEMS components can only be viewed through microscopes, and while it is possible to rotate MEMS sample in plane, this does not alter the viewpoint. Rotating the microscope sample out of plane is more difficult but is possible. However, this is affected by the same limited DOF and out of focus images discussed earlier.

The techniques mentioned above are powerful and established tools for detecting macro surface information; however, they are not ideally suited to study the micro 3D structure of MEMS surface in the MEMS lab; either due to the limitations of the MEMS chip itself (reflectivity, size) or due to the limitations of the method to extract information (accessibility, strict viewing angle, accuracy).

A fourth imaging technology, Reflectance transformation imaging (RTI) could bypass the aforementioned limitations and deliver 2.5D images of MEMS components using the coaxially fixed camera mounted on optical microscopes while using exterior light sources in various directions around the MEMS chip.

1.2. Reflectance Transformation Imaging

Reflectance Transformation Imaging (RTI) is a photographic technique (most commonly used in Archaeology) to extract surface information unattainable in conventional photography. Surface height and inclination variations, curvatures, embossed structures, occluded components, erosion patterns, edge detection are a few examples of the information that an object's surface can provide. In this technique, a camera at a fixed position atop the object takes multiple images while light sources at different angles and elevations illuminate the object's surface in sequence. As a result, surface elements located in shadow or not exposed to direct light through conventional photography become visible. The collected images simulate the object's view from different angles and give an illusion of a three-dimensional view over the surface. In addition, by merging all the captured images, two-dimensional images containing the three-dimensional characteristics of the object are formed. Epstein [12] considered a mechanism in which a camera at a fixed viewpoint and multiple light sources at distributed angles illuminated a human face. Malzbender [13], a Hewlett Packard researcher, introduced a technique named Polynomial Texture Mapping (PTM) for RTI to present an impression of the geometric features of objects with high surface irregularities. In PTM technique, object's surface can be virtually illuminated by assigning color values to each pixel as well as surface normal value which represents the brightness [8].

1.2.1. Multi-directional lighting

A number of factors contribute to capturing an image of an object such as: the object geometric features, the position of the object relative to the surrounding subjects, light reflectance properties and the viewpoint. Depending on the imaging technique and equipment used, as well as the chosen illumination method, the image quality and the information it provides about the subject may change. Looking at a surface from different viewpoints provides more information about the surface details than when the surface is viewed from one angle. However, not all objects can be observed from varied viewpoints, as the surface features or object itself may be extremely small or large, or there might be limited access to the object, or the surface features may be occluded by peripheral components. RTI simulates the varied viewpoints by placing the light source in different locations around the object. Varying shadows and highlights as the lighting angle changes reveal information about the surface characteristics.

As shown in Figure 4, self-shadow and cast shadow are two different types of shadows. A self-shadow occurs when the surface is not directly facing a light source, and a cast shadow is when another subject blocks the light source and therefore, a shadow falls on a surface. Lateral facets (non-flat facets) of bodies are typically in self-shadow in in-line illumination, by changing the lighting orientation, features with vertical lateral sides are illuminated and thus can be recognized. Subjects with a relatively higher elevation than the perimeter cast shadows on the surrounding surfaces. Changing the light position may reveal information about shape, height, or superimposed structures. Figure 4 also shows how different objects dissimilar in height and shape appear similar from the top view and under standard illumination. This is the reason RTI is able to distinguish and display various shapes and elevations, especially when angled views of an object are not feasible due to access limitations.

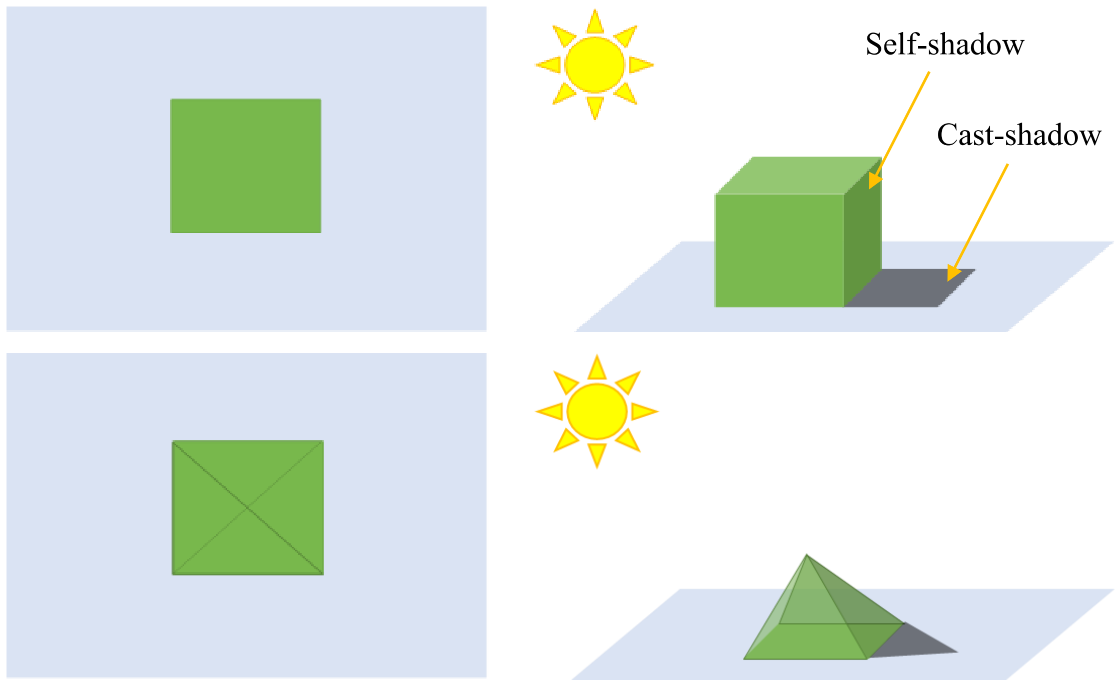


Figure 4: Left: Similar top illumination views of two different shapes. Right: Side illumination views of two shapes with cast shadow and self-shadows.

1.2.2. RTI workflow

The RTI workflow is typically conducted in three stages: light space configuration, capture, post-processing of captured images.

Light space is a three-dimensional construct for placing a certain number of light sources so that the specimen under test is illuminated from different directions. One approach called Highlight-RTI used a single light source sequentially placed in different locations. Mytum [14] conducted experiments with a tripod at three different angles from 15° to 75° above the object surface. A synchronized Highlight-RTI developed by Kitanovski [15] using a light source mounted on a robotic arm that covered 56 different positions and 4 different elevations around the object. Conversely, frame-based RTI lighting assigns fixed known positions to multiple light sources, which is advantageous in the image analyzing stage, reduces the image-acquisition time and can also block ambient light from reaching the object [16]. The stable position of the lights from the object and the uniformity of the illumination are two factors that influence the success of the RTI [17]. As shown in Figure 5, the structure should evenly distribute the light sources around the object. Moreover, light sources are all to be at oblique angles and are recommended to be at the same distance typically 2 to 4 times the diameter of the target object being imaged [16]. The form of lighting rigs can have many shapes from geodesic and hemispherical domes to arcs and arms [18]. A well-established lighting structure is a dome with a fixed suspended camera on top perpendicular to the object's surface equipped with multiple high-intensity controllable light sources [19], e.g., Malzbender [13] used a rigid and opaque dome, approximately 50-centimeter in diameter, with 40 LEDs and Solem [20] took 50 images using a 35-centimeter diameter dome with 50 fixed LEDs. Earl [18] constructed a number of lighting frames, one of them a 1-meter diameter plastic dome with 76 fixed LED lights, and the other an arc-shaped lighting rig, equipped with sixteen LEDs rotating motorized around the object. Pitard [21] used a dome with full sphere lighting, 2×56 LEDs, and 3 different camera positions in order to be able to capture multiple viewpoints of an object without having to move them.

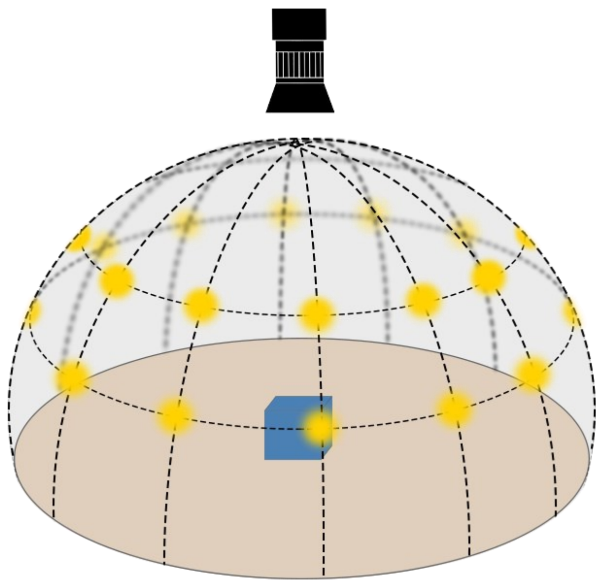


Figure 5: The dome_based light arrangement.

The method used to record images immediately after each light turns on and off can be different. It is important to not control the camera manually as movement and vibration interrupt the quality of the images [17]. Using a camera-control software or remote triggering devices are suggested. Goldman [19] used a portable USB camera to capture images. Coules [22] controlled a digital camera using an infrared remote trigger, sending out infrared light to trigger the shutter. Digital Cameras can also be controlled using Arduino microcontroller boards (see [17]). Moreover, except for the Highlight-RTI with hand-held light sources (see [23], [19]), the light operation must be controlled automatically and remotely synchronized with the camera. Coules [22] and Corregidor [17] developed a lighting array controlled by an Arduino Mega microcontroller that allows determining the lighting sequence as well as the lighting duration for each LED.

Depending on the purpose of conducting RTI, post-processing strategies could vary from reilluminating the specimen virtually through a viewing software in order to observe the features of interest and highlight surface topography, to converting the series of captured images into a composite image that improves the perception of surface data. RTI post-processing stages can artificially manipulates image properties such as color, contrast, and specularity to name but a few. In so doing, minute surface details can be made to appear more distinct, and surface height variations can be studied and measured under various lighting angles.

1.2.3. RTI in Archaeology

RTI offers a non-contact and non-invasive technique to investigate ancient objects and to discern the smallest features of a surface [5], [8]. Heritage artifacts typically contain faint surface details that are not clearly visible through ordinary imaging. RTI is used to determine the best lighting directions that highlight surface features. Additionally, ancient documents can be digitalized through RTI and visualized later by researchers without time and place constraints [17]. Digitalized documents can subsequently be used in post-processing rendering techniques which could generate synthetic images with highlighted properties of the specimen's surface [5]. The application of RTI in archaeology have been published for a wide variety of objects and artifacts, some of which are listed below.

One case study in 2010 by Earl [18] has produced interactive representations of the fragmentary surface of Herculaneum marble head, a 2000-year-old painted Roman statue. Post-processing rendering techniques generate distinct views of scratched markings and allow facets orientation of the object to become distinguishable. Altshuler [5] has extracted valuable data on ancient pottery by studying the captured images of fragments of an ancient vase using the RTI technique. Mytum [14] has investigated multiple objects that provide little information via conventional imaging due to their color or material. For instance, RTI imaging of clay pipes stamped with embossed patterns and texts uncovered invisible details.

Using the RTI technique, the study of art history is also possible. RTI has been proposed an authentication method to detect original works by focusing on the textural details of painting and on the individual characteristics particularly to each artist. Min [24] developed RTI to support the study of paintings and the artist's progress over time to reveal information based on various types of brush strokes.

1.2.4. RTI in Engineering

Surface and material engineers can employ non-invasive techniques for monitoring surfaces, for instance for managing materials degradation over time. The performance of engineered products is highly affected by corrosion, fatigue, creep, and wear that often cannot be easily detected without using technical equipment. Monitoring the surface of any investigated system needs to be quick and without leaving a lasting impact. The non-contact nature of RTI with high resolution provides engineering analysis with a series of images that emphasize surface features from different angles. Documented information through RTI can be used in many engineering disciplines to improve the design process, quality control, and monitor the performance of out-of-reach systems.

Lemesle [25] states how a wide range of physical properties such as erosion, diffusion, decay and wear can be explained by appropriately characterized surface topographies. Lemesle [25] studied metal-ceramic composites that had been worn under industrial conditions. For instance, surface slope is a characteristic that can define the stick-slip phenomenon. As another example, surface glossiness is strongly correlated to radius of surface curvature.

RTI has been proposed as an inspection method to characterize surface defects of shiny surfaces which is particularly important in industrial applications. Pitard [21] detected and analyzed visual anomalies that arise as a consequence of manufacturing process on two shiny surfaces, a gold ring of jewelry and a metal gauge block.

Coules [22] developed RTI to investigate the mechanism of failure development on a surface. A screw under investigation revealed data about failure mode and the direction of shear. A compression spring from the automotive suspension was examined to detect the reason for its failure in service. The surface of the spring was heavily corroded and surface features were not clearly visible. By combining the captured images and creating the composite image, topographical features of the surface could be discerned.

2. Basic Principles of RTI

2.1. Surface Normal

When light falls on an object, the interaction occurs between the surface and the light can be decomposed primarily into scattering, absorption, reflection, and transmission [26]. Reflection is concerned with the portion of the incident light rays that return from the surface and produce an image of the object. Reflectance Transformation Imaging (RTI) is a photographic technique based on the reflected light from an object's surface illuminated from different directions.

The visual appearance of the object is affected by the surface physical and chemical composition, as well as the light direction and wavelength. Of these factors, the reflection intensity is often highly dependent on the angle of the incident beam of light relative to the surface. As illustrated in a profile view of an object in Figure 6, the surface normal (green vector) is a vector perpendicular to the local tangent plane (red line) at any location on the object's surface [8]. Surface normal allows the perception of surface orientation and plays a key role in RTI.

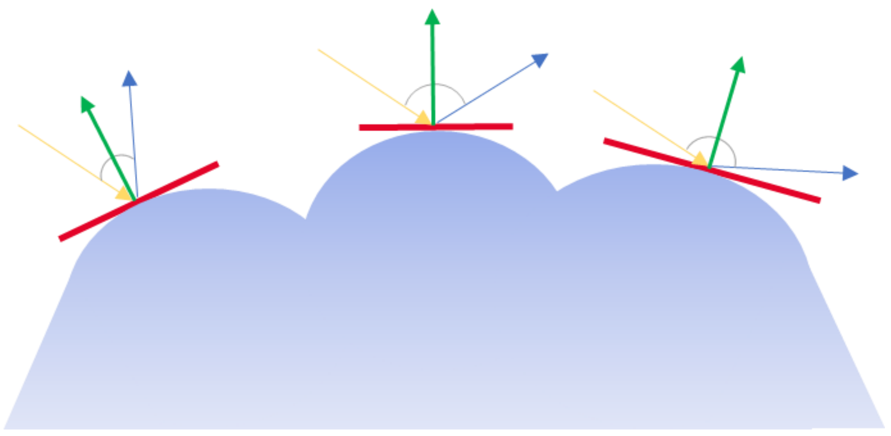


Figure 6: Incident beam (yellow) and reflected beam (blue) in relation to the surface normal (green) at three different points on a surface.

2.2. Types of Reflection

Different materials with different microscopic irregularities, matte features or glossiness, reflect light rays in characteristic ways. Two main categories are generally proposed for reflected light: specular reflection or diffuse reflection. However, the boundaries between these two reflection models and the related surfaces are not entirely separate, and thus there is a third category called spread reflection that is in between specular and diffuse reflection.

2.2.1. Specular Reflection

As shown in Figure 7, specular reflection (or mirror-like reflection) exhibits one concentrated beam of reflected light in response to an incident light. The unit vectors in Figure 7 represent respectively: the incident light direction ($\hat{L}$), local surface normal ($\hat{N}$), viewing angle ($\hat{V}$) and the reflected light direction ($\hat{R}$). According to the law of reflection as shown in Figure 8, at any given location on a surface the incoming angle θ_i and the outgoing angle θ_o with respect to the normal are equal:

$$\angle (\hat{L}) (\hat{N}) = \angle (\hat{R}) (\hat{N}) \tag{4}$$

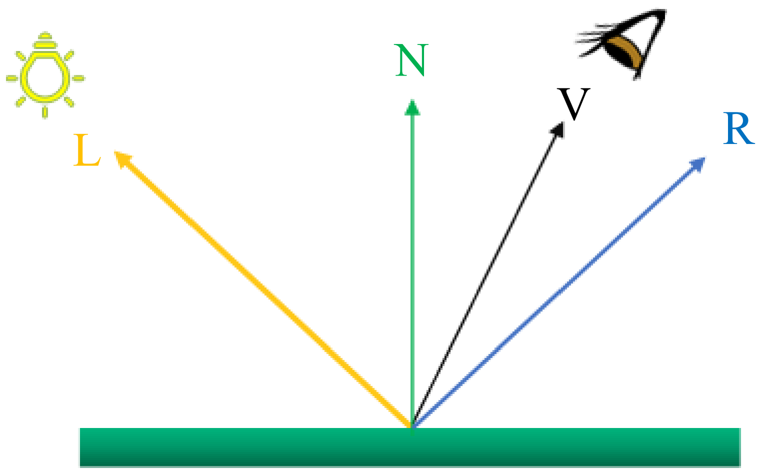


Figure 7: Schematic illustration of specular reflection that reflects light in one concentrated ray.

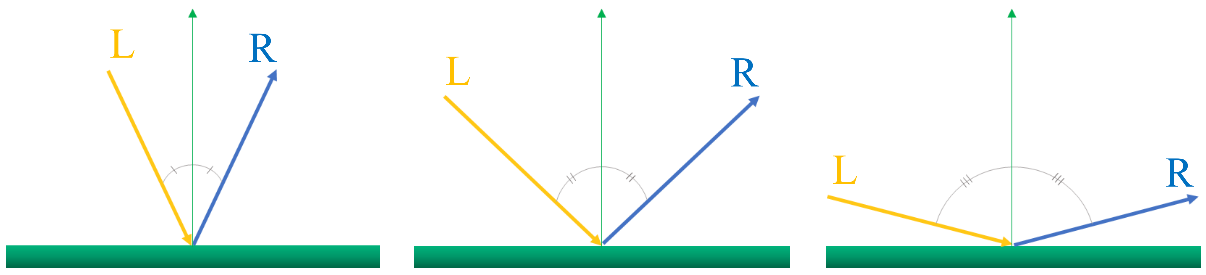


Figure 8: In specular reflection the angle between the incident light ($\hat{L}$) and the reflected light ($\hat{R}$) are equal.

Some of specular reflection properties include the following:

1. The reflected beam ($\hat{R}$) lies in the same plane as the incident beam ($\hat{L}$), and the surface normal.
2. The intensity of a specular reflection ($\hat{R}$) does not depend on the angle of incident light ($\hat{L}$).
3. Unless the reflected beam ($\hat{R}$) and the viewing angle ($\hat{V}$) align perfectly, the viewer cannot see the illuminated point on the surface.

2.2.2. Diffuse Reflection

Diffuse reflection occurs for rough / matte surfaces with microscopic irregularities which tend to scatter the incident beam of light diffusely in all directions [26]. As shown in Figure 9, for a perfectly diffuse surface, the output rays are reflected from the incident point in the form of a hemisphere (blue circle) with decreasing luminous intensity as the angle from the surface normal increases. Diffuse surfaces can be seen from a range of angles, due to the reflected beams that scatter all around the point of incident.

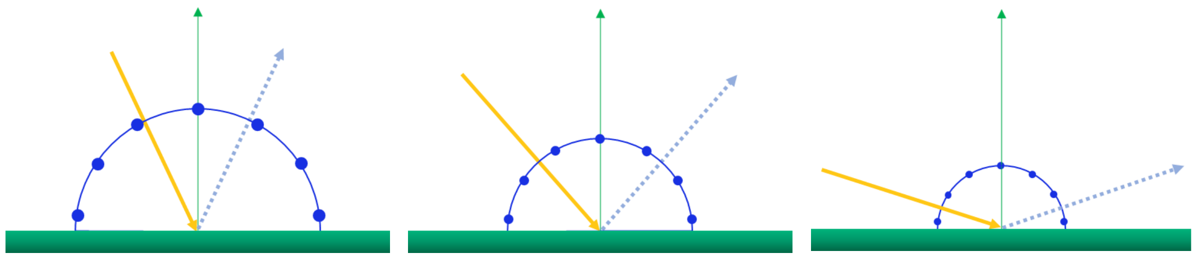


Figure 9: Schematic illustration of an ideal diffuse surface scattering light.

Diffuse reflection can be modeled by the Lambert's law where a rough surface reflects light according to a cosine angular diffusion:

$$I_o = I_i \cos\theta \tag{5}$$

Where I_o and I_i are respectively the luminous intensity of the reflected beam and the incident beam, and θ is the angle between the surface normal and the incident beam. The units of luminous intensity here are optical power per solid angle (steradians).

2.2.3. Spread Reflection

Most objects are visible by visual perception due to diffuse surfaces that contribute to bouncing incident light in all directions, while a specular reflective surface can only be observed at the specific angle according to the angle of the reflected light, like a mirror. There is in fact a continuous spectrum of reflections between the specular and diffuse reflection, that can have more specularity or diffusivity depending on the surface roughness. As depicted in Figure 10, for partially rough surfaces a narrow range of reflections appears around the specular reflection, termed the specular lobe. As the surface shows more roughness and exhibits less glossiness, the width of the specular lobe increases, its length decreases and the surface starts getting matte [27]. This can be modeled empirically as follows.

$$I_o = I_i \cos\theta^n \tag{6}$$

Where I_o and I_i are respectively the intensity of the reflected beam and the incident beam, and θ is the angle between the surface normal and the incident beam. n is the shininess constant which is an empirical value that characterizes how specular or diffuse the reflection is with regards to surface material properties. Higher values of n attenuate the volume of the specular lobe. Note that this empirical equation can be made to match the observed brightness by tuning n, but the parameters in the equation are not derived from any physical roughness parameter of the surface.

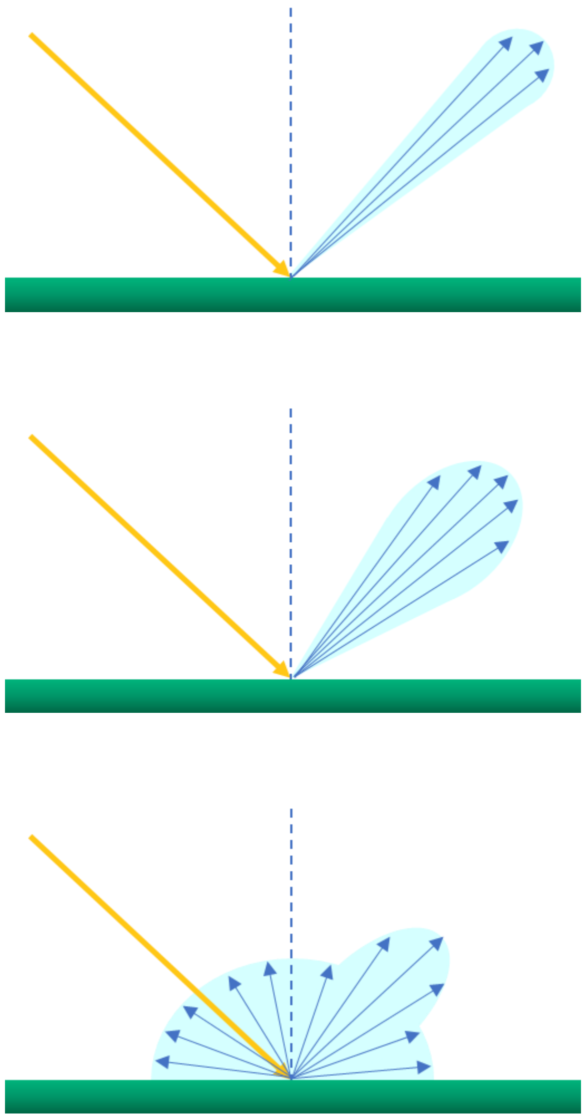


Figure 10: Three reflection states ranging from primarily specular to primarily diffuse reflection. As the roughness increases, the elongated lobe width also expands while the spike size shrinks.

2.3. Reflection Models

2.3.1. Phong Reflection Model

In this thesis we will use empirical or observation-based models describing the reflection of glossy materials. The Phong reflection model as shown in Figure 11 considers the ambient reflection component along with diffuse and specular reflections as the main components to describe the surface reflection [28].

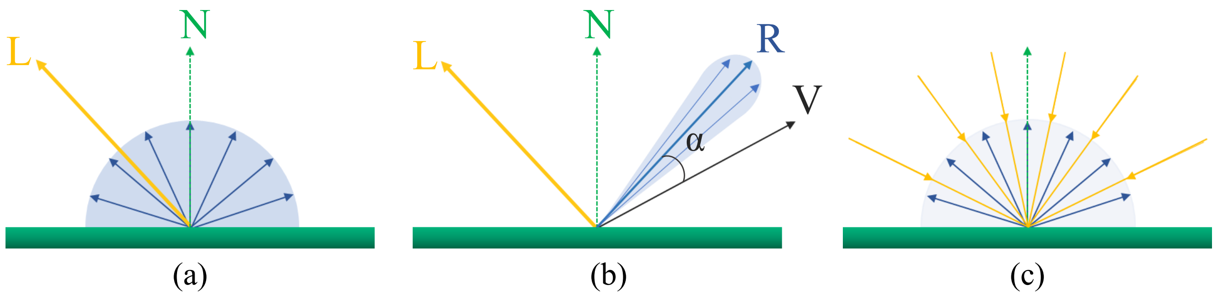


Figure 11: Phong reflection model consists of (a) diffuse reflection, (b) specular reflection and (c) ambient reflection.

The ambient reflection component demonstrates the effect of illumination incident on the point after bouncing off other surfaces in the environment. As part of this study, the experiments are executed with no ambient light and thus the ambient light reflection component of the Phong model is omitted.

In the Phong reflection model the diffuse reflection component is represented by Lambertian surfaces with uniform reflections in every direction. Hence, the diffuse reflection light intensity, I_d, of the Phong model is independent of the viewing direction and is dependent on the angle of illumination, θ, as shown in Figure 11 (a), and it is given by,

$$I_d = K_d I_i \cos\theta \tag{7}$$

Where the Ii represents the illumination light intensity, and K_d the diffuse reflection surface constant.

The second component, the specular reflection light intensity, I_s, as shown in Figure 11 (b), depends strongly on the viewing angle ($\hat{\mathrm{V}}$) with respect to the surface normal ($\hat{\mathrm{N}}$), and the index of surface roughness (n), according to the following equation:

$$I_s = K_s I_i \cos\alpha^n \tag{8}$$

Where n is infinity for a perfect mirror, and it tends to zero as surface roughness increases. The K_s denotes the specular reflection surface constant, and I_l is the intensity of specular component of the light source [29].

Finally, the Phong reflection model is defined as (see Figure 12):

$$I_{Phong} = K_d I_l \cos\theta + K_s I_l \cos\alpha^n \tag{9}$$

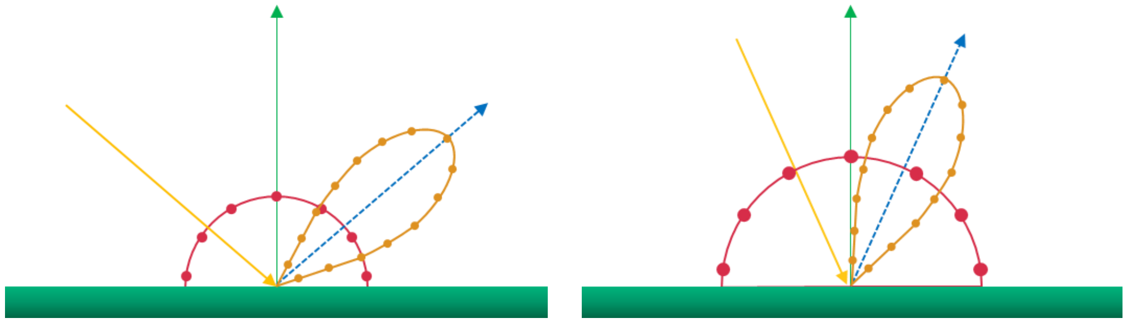


Figure 12: Schematic illustration of Phong reflection model for two different incident light angle and same lobe parameter (n).

2.4. Light Reflectance in RTI

The implementation and configuration of RTI will depend on the reflection characteristics of materials, therefore determining the surface topology and the glossiness/matteness of the surface are important. Many objects reflect light diffusely: reflecting light broadly in many directions, and thus the surface features can be captured and discerned using RTI illumination from a broad range of angles. However, mirror-like surfaces or glossy materials tend to reflect light in one particular direction or a narrow spike about the reflected beam, meaning that the specular reflection is highly viewpoint-dependent. If the viewpoint or illuminating source is not located at the right spot, no image can be viewed. In RTI, the camera is assumed to lie in a fixed location while the light source illuminates the surface from different angles. Success in imaging shiny surfaces through RTI depends to a great extent on empirical examination as to how the lighting structure should target each point on the surface so that an image is captured by a fixed camera. As illustrated in Figure 13, specular reflections with a relatively small lobe can only be captured, if the light source is located close enough to the camera.

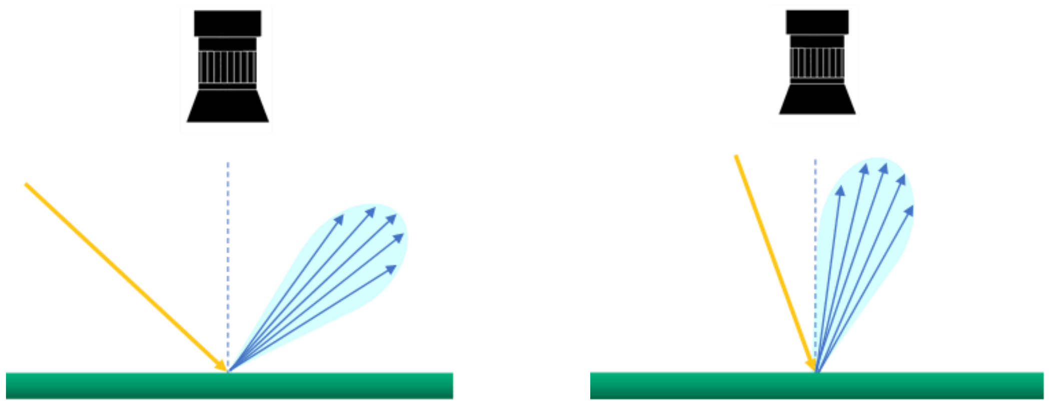


Figure 13: Schematic illustration of capturing glossy surfaces and the importance of illumination direction.

3. RTI Software and Hardware Requirement

This chapter presents the apparatuses required to perform Reflectance Transformation Imaging (RTI). The programming environment used to control the apparatuses is introduced, as is the method for producing multi-directionally illuminated images.

3.1. Light Sources

The specimen under test in Reflectance Transformation Imaging (RTI) is illuminated multi-directionally and photographed from a fixed viewpoint. Hence, the light source used for conducting RTI is required to meet some specifications including:

1. Programmable so that the illumination intervals can be defined
2. Addressable individually so as to be controlled
3. Mountable on different lighting structures
4. Portable in terms of size, weight and shape
5. Consistent output so that the brightness of captured images remains uniform

The light sources used in this thesis are NeoPixels from Adafruit hardware company, which have RGB LED pixels (~5 mm × 5 mm) chained together in different forms such as strips, ring, boards and sticks as shown in Figure 14. Each NeoPixel form factor has 3 input wires to be activated: Ground, Power and Data connection. Each LED on a NeoPixel can be turned on and off individually with control over the lighting intervals, brightness intensity and color via data connection single-wire control. Red, green and blue LEDs can be combined to produce white and intermediate colors. The power required is 5 Volts and 60 milliamps to each pixel. For instance, for a NeoPixel strip with a total of 50 bright high-intensity LEDs, a maximum of 3 Amps is required if all LEDs are to be simultaneously on. Much less power is required if only one or a few LEDs are lit at any one time. Powering NeoPixels does not switch on the LEDs until the data input wire connected to one pin of a microcontroller is activated, which will be discussed in section 3.2.

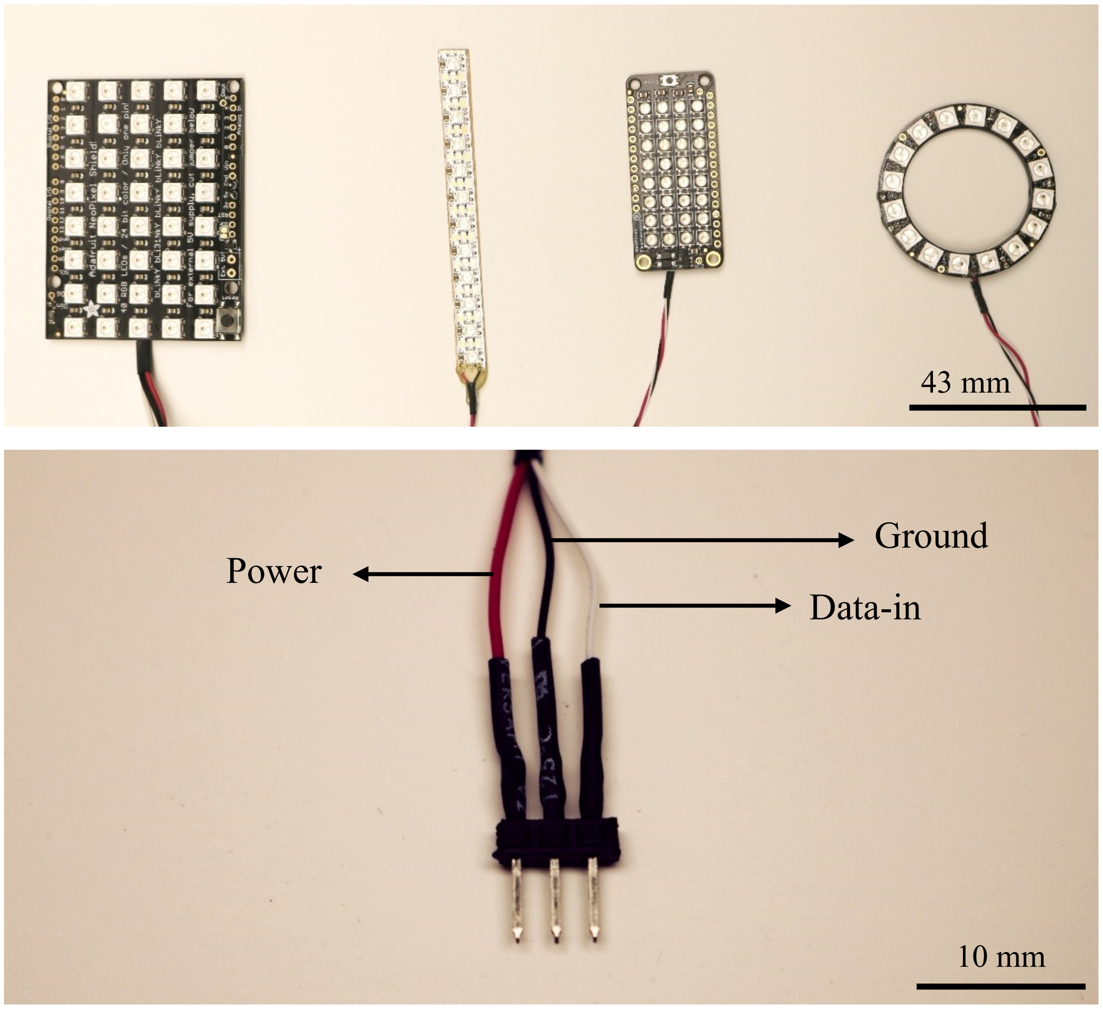


Figure 14: Four different types of NeoPixels (top): 5x8 Shield, 8x1 strip, 4x8 FeatherWing and 16 Ring. Three input wires of each NeoPixel (bottom).

3.1.1. Light Frames

The lighting direction can be altered depending on the dimensions of the test specimen and what features are subject to investigation. A hemisphere lighting frame can be used [13] to illuminate the specimen surface uniformly from different angles; an established method is to place light sources at the same distance from the specimen [16]. Dome-based lighting was first used by Malzbender and following that RTI studies have used dome lighting structures to illuminate subjects (see [30], [18]).

The wide range of NeoPixels form factors allow us to explore different possible lighting frames that can be used to illuminate samples, especially when viewed through an objective having a limited working distance. A NeoPixel based hemisphere was constructed at Dalhousie using 15-centimeter diameter 3D printed PLA dome with a flexible soft strip of 50 NeoPixels hot glued above 50 holes on as depicted in Figure 15.

The LEDs are arranged in 4 latitude bands: 18 LEDs near the equator, next a band of 15, then a band of 11, and a top 'polar' band of 6 LEDs. The number decreases towards the poles as the circumference decreases and fewer LEDs are required to maintain the inter LED spacing.

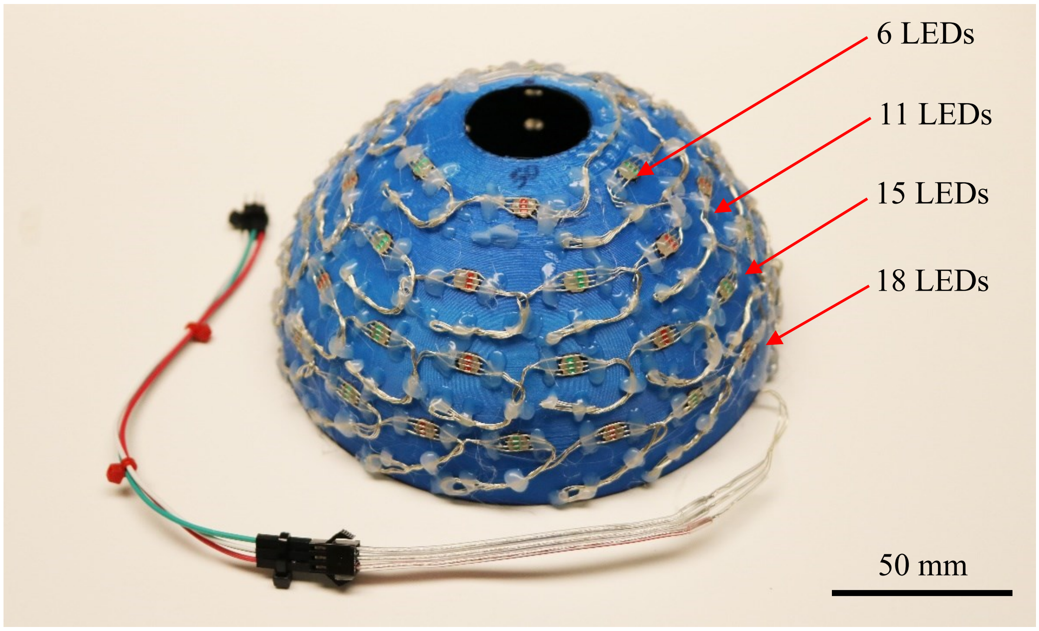


Figure 15: 3D printed hemisphere and the NeoPixel flexible strip.

3.2. Arduino

In addition to a power supply, a microcontroller is required to address each LED on a NeoPixel and specify its color and brightness. NeoPixels are sold by Adafruit which also provides libraries for Arduino so that the NeoPixels can be controlled though simple codes written in open-source Arduino IDE software. For example, an Arduino Uno as shown Figure 16 is an inexpensive circuit board that contains a microcontroller that can be programmed to control NeoPixels.

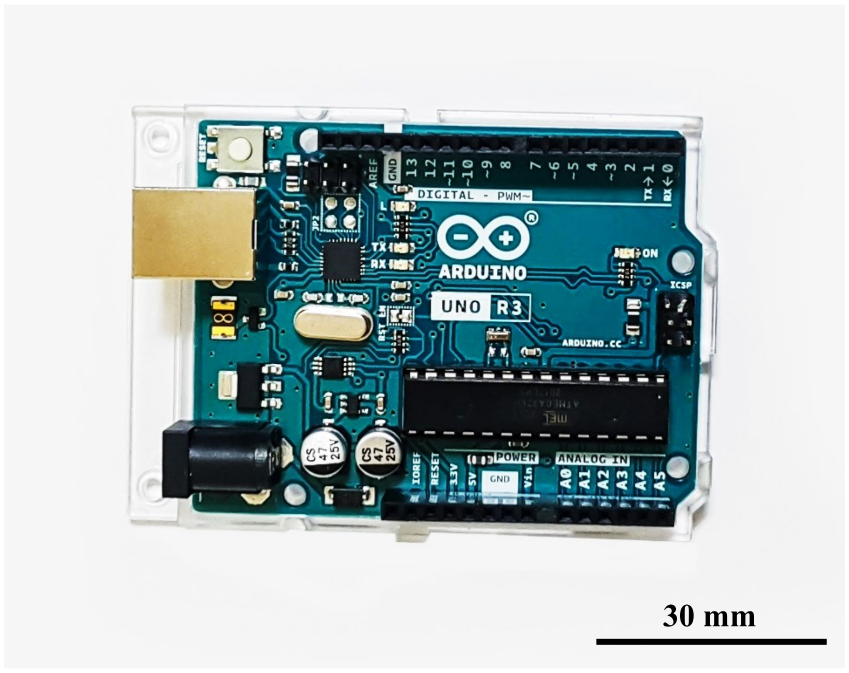


Figure 16: Arduino Uno board.

The Arduino USB port connected to a computer supplies power and allows communication with the Arduino. The Power input wire of NeoPixel is connected to the 5V output pin, the Ground wire of the NeoPixel is connected to the GND pin of the Arduino, and the Data-in wire of the NeoPixel is connected to one of the digital pins of the Arduino Uno board. The USB connection can nominally handle 500 milliamps, and thus considering the nominal 60 milliampere current draw up of each LED, having 8 lit LED at the same time could reach the maximum current power. A separate power supply should be used if more LEDs need to be lit up all together. Rather than direct programming of the Arduino boards through the Arduino IDE, which requires compiling of the programs, the hardware of this study is controlled via codes written in MATLAB. MATLAB has a support package for controlling Arduino boards as well as built-in NeoPixel libraries. MATLAB also allows for control of the camera using built-in libraries as well as real time viewing of the captured images. This process is explained in section 3.4.

3.3. Camera and Microscope

While the ultimate goal is to image MEMS devices using the MEMS lab microscope a number of different microscope setups were built and tested, partly due to 2020/21 COVID restrictions and partly to start at larger scales and move to smaller scales.

To investigate RTI at the mm scale, a USB microscope was used. Figure 17 shows a Celestron digital portable USB microscope. A 5MP digital camera captures images with a range of resolution (640 x 480 → 2592 x 1944). Working distance varies from 10 mm to several cm.

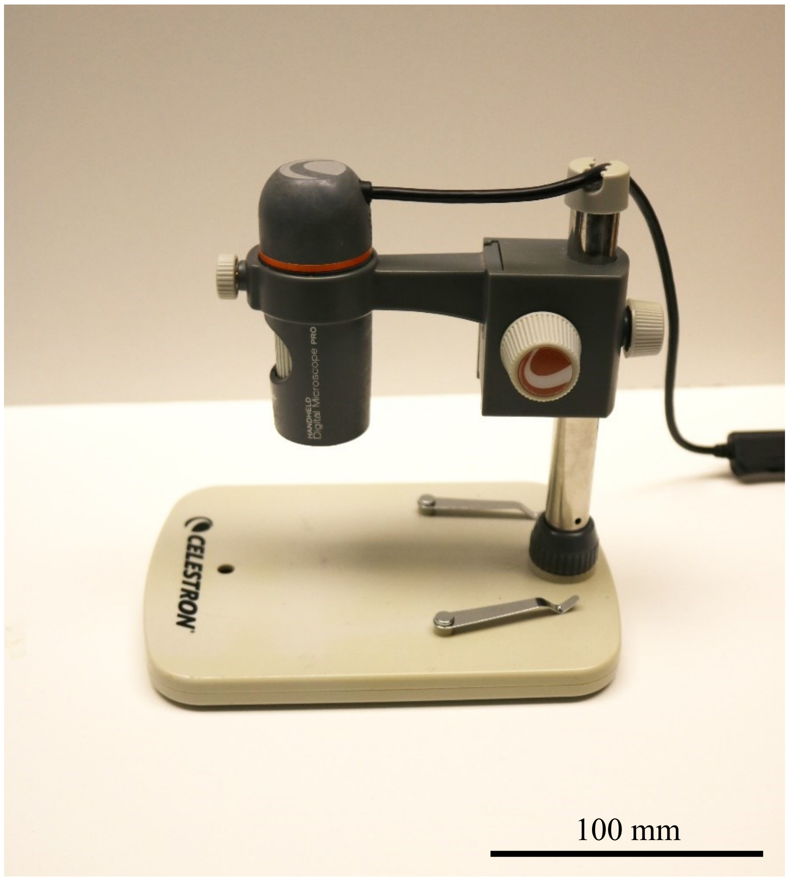


Figure 17: Celestron handheld USB digital microscope.

By installing the MATLAB support package for USB webcam add-on MATLAB is capable of identifying the USB microscope as a webcam and taking pictures. The details of the capturing process are explained in section 3.4.

The USB camera was combined with the 3D printed NeoPixel hemisphere described above (see Figure 18). The cylindrical microscope frame was placed in a 35mm diameter hole at the apex of the 3D printed hemisphere and multiple mm-scale objects were imaged (see Chapter 4).

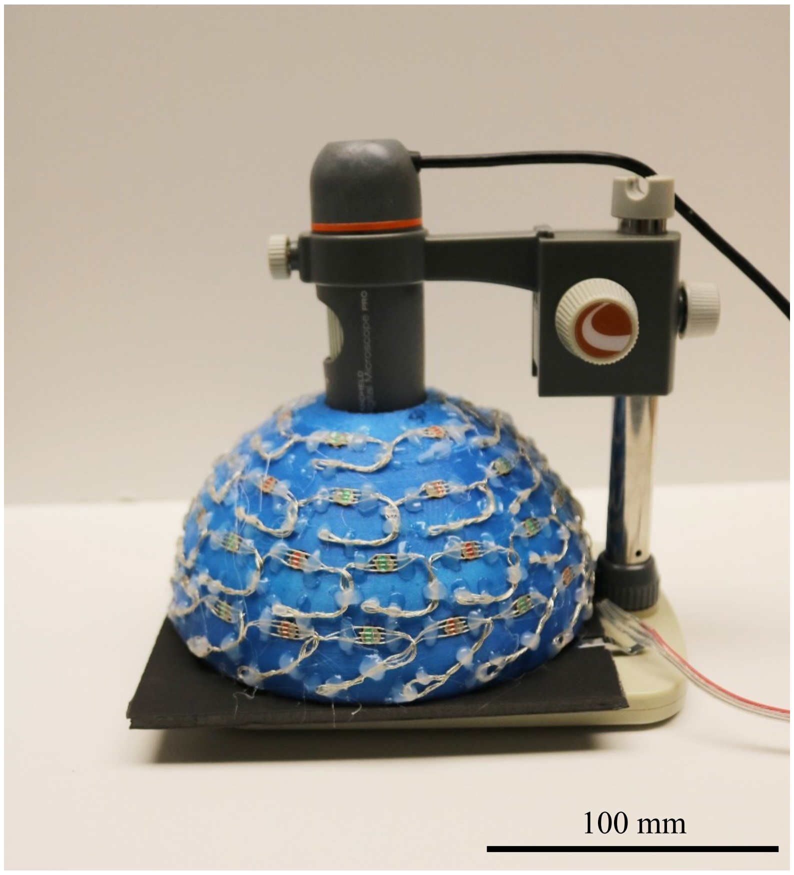


Figure 18: The 3D printed hemisphere and the USB camera.

3.4. MATLAB

Programming of the USB Camera and NeoPixels in MATLAB begins with the installation of the relevant support packages. These packages and relevant information can be found in MATLAB through the Home Tab → Environment section → Add-Ons option. The desired package for Arduino is called "MATLAB Support Package for Arduino Hardware" and has been extended to work with NeoPixels through a second Add-on called "NeoPixel Add-On Library for Arduino". Having these two packages properly installed, the Arduino will be ready to communicate with NeoPixels.

Figure 19 shows parts of the written code to light up a ring NeoPixel with 12 LEDs through Arduino Uno board. First, the user initializes the Arduino hardware by determining the USB serial port connected to the Arduino supply connector, followed by the name of the used Arduino board. The desired library needed to control the NeoPixel hardware, Adafruit/NeoPixel, is also addressed. The NeoPixel is then initialized by mentioning the name of the Arduino pin to which the NeoPixel data wire is connected, as well as the number of NeoPixel LEDs. The subsequent lines determine the brightness level of the LEDs (range is from 0 to 1), turn on the 12 existing LEDs on the Ring NeoPixel in red light and turn them off after a tenth of a second pause.

```
Arduino_Uno =arduino('COM5','Uno','Libraries','Adafruit\NeoPixel');
NeoPixel =addon(Arduino_Uno,'Adafruit/NeoPixel','D6',12);
NeoPixel.Brightness = 0.8;
writeColor(NeoPixel, 1:12 , 'red');
pause(0.1);
writeColor(NeoPixel, 'off');
```

Figure 19: MATLAB program to set up Arduino and NeoPixel.

The webcam support package needs to be installed so that the image viewed by the USB webcam camera can be read by MATLAB. This support package is called "MATLAB Support Package for USB Webcams" and can be downloaded through the previously stated path in MATLAB home tab. Figure 20 represents an outline of the program that was written in MATLAB to image specimens through a USB microscope.

```
my_camera = webcam(1);
my_preview = preview(my_camera);
 pause;
closePreview(my_camera)

my_image = snapshot(my_camera);
imshow(my_image);

file_extension = 'png';
image_name =  'Test_image_';

imwrite(my_image,[image_name,file_extension])
```

Figure 20: MATLAB program to set up a USB camera, capture and save an image.

Assuming there is only 1 camera connected, then MATLAB assigns it to number 1. The “webcam()” function is to define the selected camera as a new object to MATLAB. After using the “webcam()” function, properties of the camera can be manipulated, such as resolution, brightness and contrast. Next, the camera image is previewed; the user can manipulate the position and focus of the camera, press any key and the window will be closed. The “snapshot()” function captures the specimen under test. The “imwrite()” function can then save the captured image with a specified name and format in the current directory.

A combination of the programs written to activate NeoPixel and the USB camera can be used in the RTI imaging technique. First, the hardware needs to be connected properly as stated in previous sections.

As shown in Figure 21, first the camera, Arduino and NeoPixels are initialized using the proper ports and pins. The NeoPixel used is the flexible strip with 50 LEDs, connected to the pin “D6” of an Arduino Uno. Next, a save directory, a filename and extension are specified. Then the resolution of the camera is selected. The available resolutions can be accessed using the “webcam()” function in command window. Next, the “preview()” function shows a live view of the camera image, and thus the user can manipulate the specimen, the focus of the camera and/or the working distance. Then, a counter is defined and a *for i* loop can be used to turn on each LED on a NeoPixel. Another *for j* loop takes one or more pictures while the LED is on and then saves each. Note that for image numbers below 10, a leading 0 is added to ensure the images are sorted properly. The current LED is turned off at the end of the loop and then the process is repeated for the next LED on the NeoPixel. Because the LEDs are not turned on at the same time, the current consumption remains below the maximum current supplied by the Arduino.

```
arduino = arduino ('COM4', 'Uno', 'Libraries', 'Adafruit\NeoPixel');
    disp(['Connected to:  ',arduino.Board,'  on ', arduino.Port]);
NeoPixel = addon(arduino, 'Adafruit/NeoPixel', 'D6', 50);

% Set up Connection to Webcam
    my_cam = webcam(1);

    disp('Select directory to save images...')
    path = uigetdir;
    cd(path)

file_extension = 'png';
image_name =  'Pix_#';

modes = my_cam.AvailableResolutions;
 my_cam.Resolution ='1280x720';

my_preview = preview(my_cam);
  pause;
closePreview(my_cam);


 for i = 1:50
    for j = 1:num_pics
    writeColor(NeoPixel, i, 'white');
    pause(0.2);

    my_image = snapshot(my_cam);
    imshow(my_image);

 if j<10

imwrite(my_image,[image_name,'_0',num2str(j),'.',file_extension])
   else
       imwrite(my_image,[image_name,'_',num2str(j),'.',file_extension])
   end
    writeColor(NeoPixel, 'off');
    end
 end
```

Figure 21: The RTI_Capture MATLAB program.

When the program is properly executed, a set of images is produced. Using the strip NeoPixel and the hemisphere, 50 images are taken from each specimen, in each of which the lighting angle is different. Figure 22 shows 4 sample images of 3 small rounded beads ranging in size from 4 mm to 6 mm diameter. Browsing the images simulates a visual relighting of the object. These images are the input of a series of post processing and Virtual-Relighting MATLAB programs that provide information about the 3D structure of the surface under study. The Virtual-Relighting and post-processing phase is explained in Chapter 4.

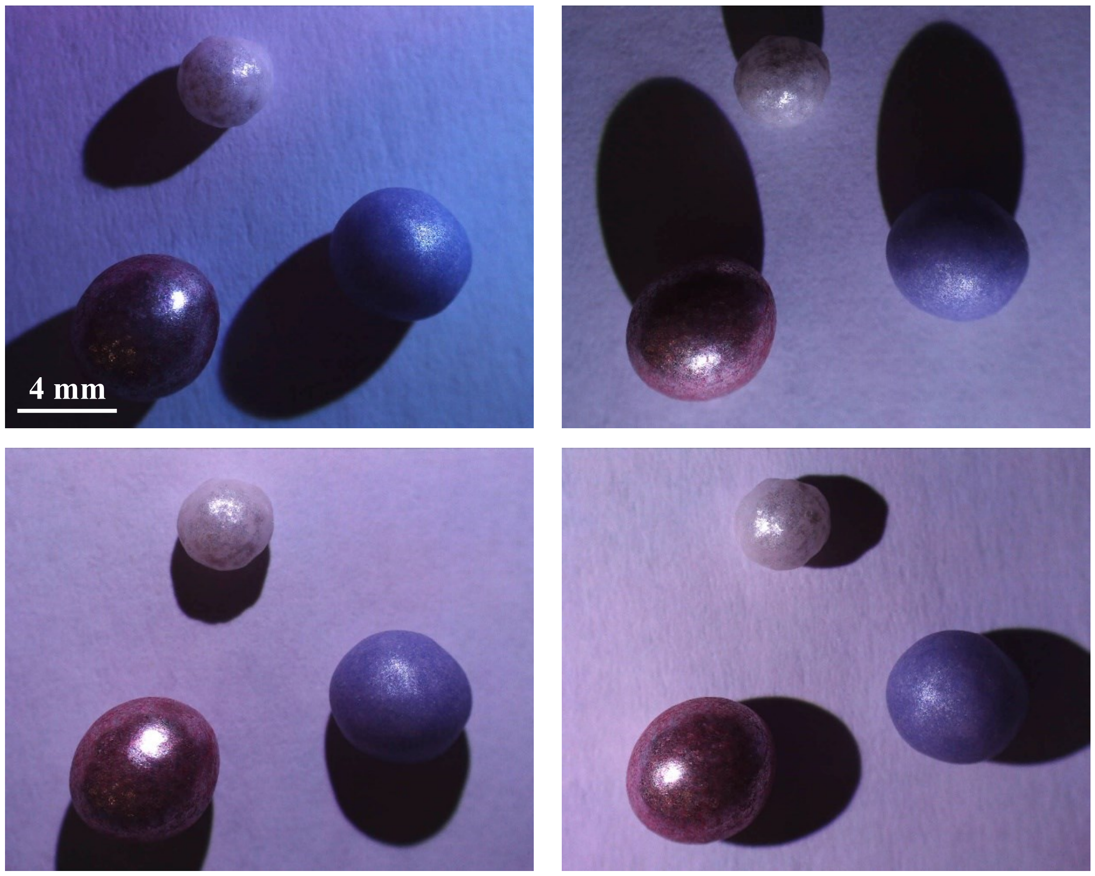


Figure 22: Four balls under test while the illumination angle varies.

4. RTI Viewer

In this chapter, the third stage of RTI workflow, post-processing of captured images, is discussed. The idea behind this stage is to identify the information within a sequence of two-dimensional images and detect relationships between the three-dimensional properties of a surface and their two-dimensional representation in an image. Note that the diagnostic methods presented in this chapter are primarily descriptive.

This stage is divided into two general categories; firstly, the virtual reillumination, whereby a sequence of images captured in varying lighting directions can be browsed, and secondly composite images each of which can provide specific information about the three-dimensional structure of a surface.

4.1. Virtual Reilluminating

In the previous chapter, we have seen how by employing a hemispherical lighting frame and a USB microscope, a sequence of images captured in different illumination directions can be obtained. Each of these images are arrays where each element in the arrays holds intensity information that can be used in surface interpretation. These images stacked together can simulate the multi-directional illumination of a surface under observation. As described in chapter 3, the hemisphere lighting frame used in this study defines the position of each LED in three-dimensional space based on two angles: ϕ is defined to be the polar angle or colatitude between 0° to 90° (in this case 4 bands of ϕ between 25° for the top band of 6 LEDs and 73° for the lower band of 18 LEDs), and θ is the longitude or azimuth angle which ranges between 0° and 360°. The top band will have 6 different θ and the bottom band will have 18 different θ.

As described in the previous chapter, the RTI capture program numbers the images according to the order the LEDs light up and the microscope takes pictures. Although each image has its ϕ and θ index, images in the directory are stored as image numbers from 1 to 50.

To browse captured images, the user is first required to specify a directory in which the captured images are stored. Then the “dir” function lists the contents of the selected directory with a *png* extension. A variable is then assigned to the number of images, and the name of images is extracted from the structure, as shown in Figure 23.

```
path = uigetdir;
list_of_images = dir(strcat(path,'\*.','png'));
num_pics = size(list_of_images,1);
file_names(1:num_pics,:) = char(list_of_images(1:num_pics).name);
```

Figure 23: MATLAB program to list the name of images from a directory.

The next step is to read the images one at a time and place them in a stack. A stack is a multi-dimensional array in which the content of each color image along with its illumination direction information is stored. A single RGB image is comprised of an $m \times n \times 3$ array which contains the information of the 3 color channels. Each illumination direction corresponds to the azimuth (θ) and colatitude angle (ϕ) of the LEDs associated with each image. To stack the images, first a zero 5-dimensional array ($m \times n \times t \times p \times 3$) is defined to reserve memory for the $m \times n \times 3$ RGB images at the specific lighting direction of each image (t denotes θ theta locations and p denotes ϕ phi locations). Then as shown in Figure 24, a *for i* loop fetches image files one at a time from the predefined list of names and stores each image in the corresponding θ and ϕ number of the pre-defined zero multidimensional array.

```
phi_val = 1;
theta_val = 1;

theta_table = [18 15 11 6];
theta_max = max(theta_table);
phi_max = 4;

image_stack =zeros(y_size,x_size,theta_max,phi_max,3);

for i=1:num_pics
    ith_image = imread(file_names(i,:));
    image_stack(:,:,theta_val,phi_val,:)=ith_roi;

    theta_val = theta_val+1;
    if theta_val > theta_table(phi_val)
        phi_val=phi_val+1;
        theta_val = 1;
    end
end
```

Figure 24: Part of the MATLAB program to store RGB images.

As mentioned earlier, the images are named according to the order of capture, and our lighting frame consists of 4 classes of colatitude, each of which has respectively 18, 15, 11 and 6 LEDs at different angles. To store images, once the maximum number of images per colatitude (ϕ) is reached, the ϕ number increases by one, and then θ number is reset to one. When the loop reaches the end, a 5-dimensional array of RGB images is acquired as shown in Figure 25, which can be used in the following steps to display each image independently.

Throughout this chapter, frequently used term are as follows: 1. High LEDs: the six most elevated LEDs at a polar angle of 25 degrees (ϕ_4). 2. High Images: the six images captured under the illumination of Top LEDs. 3. Low LEDs: the 18 least elevated LEDs at a polar angle of 73 degrees (ϕ_1). 4. Low Images: the 18 images captured under the illumination of Low LEDs.

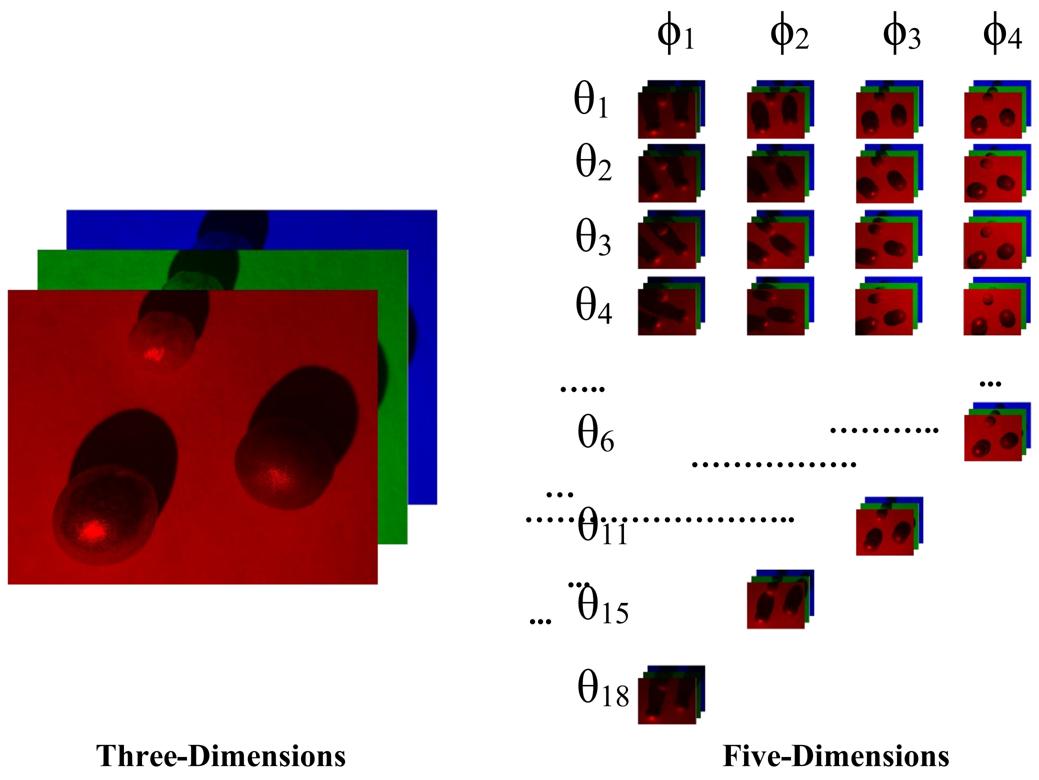


Figure 25: Illustration of a five-dimensional array of RGB images.

Images can be accessed from the image stack and then be displayed individually by specifying their ϕ and θ numbers. The algorithm used for this purpose is shown in Figure 26. *Theta* and *Phi* are the assigned longitude and colatitude of the image to be displayed. The "squeeze" function is used to collapse the dimensions of the final array from 5 to 3 as only one image is displayed at a time.

```
display_image = squeeze(image_stack(:,:,theta,phi,:));

imshow(display_image)
```

Figure 26: Part of the MATLAB program to display images separately.

To interact with the user and enable image browsing, a variable is defined in a *while* loop, in which the key pressed by the user is stored and a command is executed accordingly. In Figure 27, the "get" function contains the character pressed by the user, and the "double" function converts the character to an integer according to the ASCII codes defined for each key on a keyboard. Then the *key_value* is set as a switch expression that executes a statement depending on the key pressed by the user. The ASCII codes for the left and right arrow keys on the keyboard are 28 and 29, respectively. Therefore, pressing these keys executes one of the two cases shown in Figure 27. Initially, *Theta* and *Phi* are set to one. Each time the user presses the left arrow key, the *Theta* number decreases by one, and upon reaching 1, the *Theta* number jumps to the maximum value which is the last image in the same *Phi*. In a similar manner, pressing the right arrow key increases the *Theta* value by one, and as soon as the *Theta* reaches the maximum, the value is reset to one.

```
key_value = double(get(gcf,'CurrentCharacter'));
  switch key_value
         case 28
            if theta > 1
                theta = theta-1;
            else
                theta = theta_max;
            end
        case 29
            if theta <theta_max
                theta = theta+1;
            else
                theta = 1;
            end
```

Figure 27: MATLAB code to update the Theta value.

The value of *Phi* can also be altered in a similar way to the *Theta* algorithm by user control over the up and down arrow keys, as depicted in Figure 28. One requirement of *Phi* and *Theta* control is to update the *Theta* number when moving between different *Phi*-s. Since the number of LEDs in each elevation of the hemisphere is different, the *Theta* number should be interpolated when the *Phi* number is adjusted.

Being in a *while* loop, each time an arrow key is pressed, the previous image closes and the new image is opened which corresponds to the new *Phi* and *Theta* value. The time interval between the closing of the previous image and the opening of the later one is brief, and thus the images appear to be browsed sequentially.

```
case 30
    if phi <phi_max
        old_theta_max = theta_max;
        phi = phi+1;
        theta_max=theta_table(phi);
        theta=1+round((theta-1)/old_theta_max*theta_max);
    end
case 31
    if phi > 1
        old_theta_max = theta_max;
        phi = phi-1;
        theta_max=theta_table(phi);
        theta=1+round((theta-1)/old_theta_max*theta_max);
    end
```

Figure 28: MATLAB code to interpolate new Theta value when changing Phi value.

4.1.1. Image Interpretation

Both RTI_Capture and RTI_Viewer programs have been tested on multiple mm scale objects. Each group of data sets contains 50 images, and it takes less than a minute to run the program and review all the captured images. Due to the descriptive nature of post-processing stage in this study, the process of interpreting the images may vary depending on the characteristics of the subject. Common identifiable properties can be summarized as follows:

Orientation of facets. Assuming that most surfaces exhibit spread reflection, in order for the camera to capture the specular highlights, the reflection beam and the viewing angle must overlap or be as close as possible (see Phong model, Chapter 3). As Figure 29 illustrates, with respect to the law of reflection, the angle of the surface normal at points whose spread reflection is toward the camera will be half the angle of the incident light (ϕ). Hence, the surface at those points would make the same angle of $\phi/2$ with the horizontal, which in this case, with four groups of light on the hemisphere at polar angles of ϕ_1=73°, ϕ_2=57°, ϕ_3=41°, and ϕ_4=25°, it would be possible to detect surfaces with approximate angles of 36.5°, 28.5°, 20.5° and 12.5°, respectively.

Note that the accuracy of the above results is limited by: the LED position accuracy, the change in radial distance between each LED and different parts of the surface, and LEDs light intensity variation.

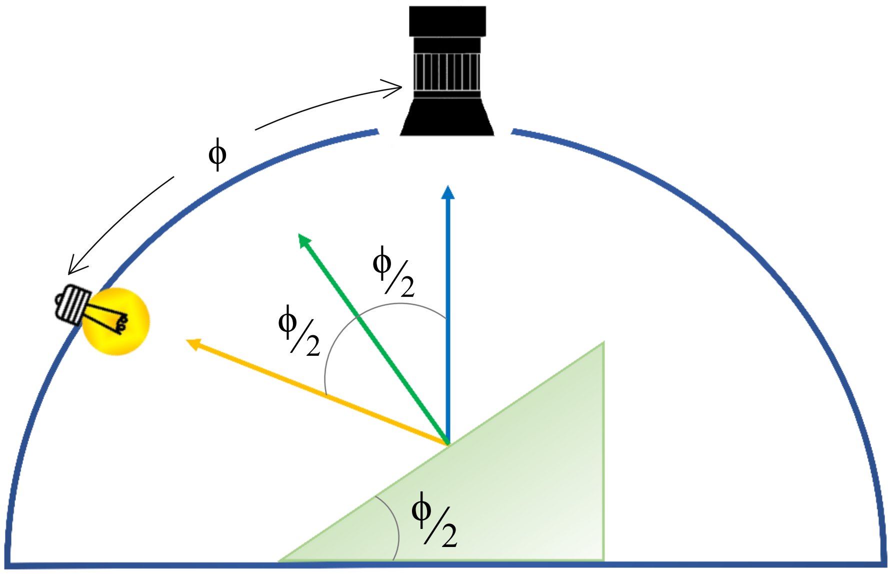


Figure 29: Perception of the surface slope from highlight reflections in the images and the polar angle of the incident light.

Depression and Elevation. Elevated features cast shadows over the base surface when illuminated from angles. The shadow length varies in direct proportion to the polar angle of the incident light. On the other hand, depressions cannot be identified by casting shadows. A simple approach could be to trace the shadow that the rim of the depression casts over the interior region of it, and thus depending on the depression width and depth, as the polar angle of the incident beam increases, less area of the depression lights up, and vice versa. One approach might be to identify depressions by comparing them to detected elevations. In general, when the westerly light illuminates the west coast of an elevation, the same incident light illuminates the east coast of a depression, this difference can also be used to infer these features. The interpretation procedures used to identify depression and elevations may vary depending on the 3D characteristics of the subject under test.

One of the millimeter-scale objects under test is a red square plastic brick (Nanoblocks^{TM}) with dimensions 7.8 × 7.8 × 3.1 mm, as shown in Figure 30. There are 4 cylindrical studs on the top face of the brick and a 4 mm ball attached to the left side. The RTI_Viewer program creates a sequence of images, as shown in Figure 31, in which the reflectance properties of the surface, specular highlights and shadow effects reveal the geometric characteristics of the surface. The images show the subject under illumination from three different θ directions and two ϕ directions. Each of these images can be examined to obtain important information about the geometry of a surface, for example the shadows are longer on the higher ϕ set of images.

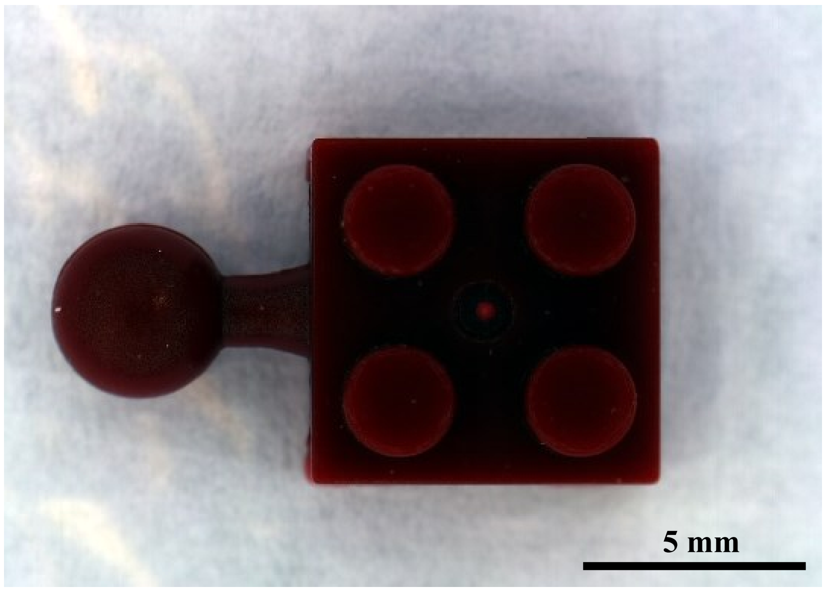


Figure 30:An image of a plastic brick under standard illumination.

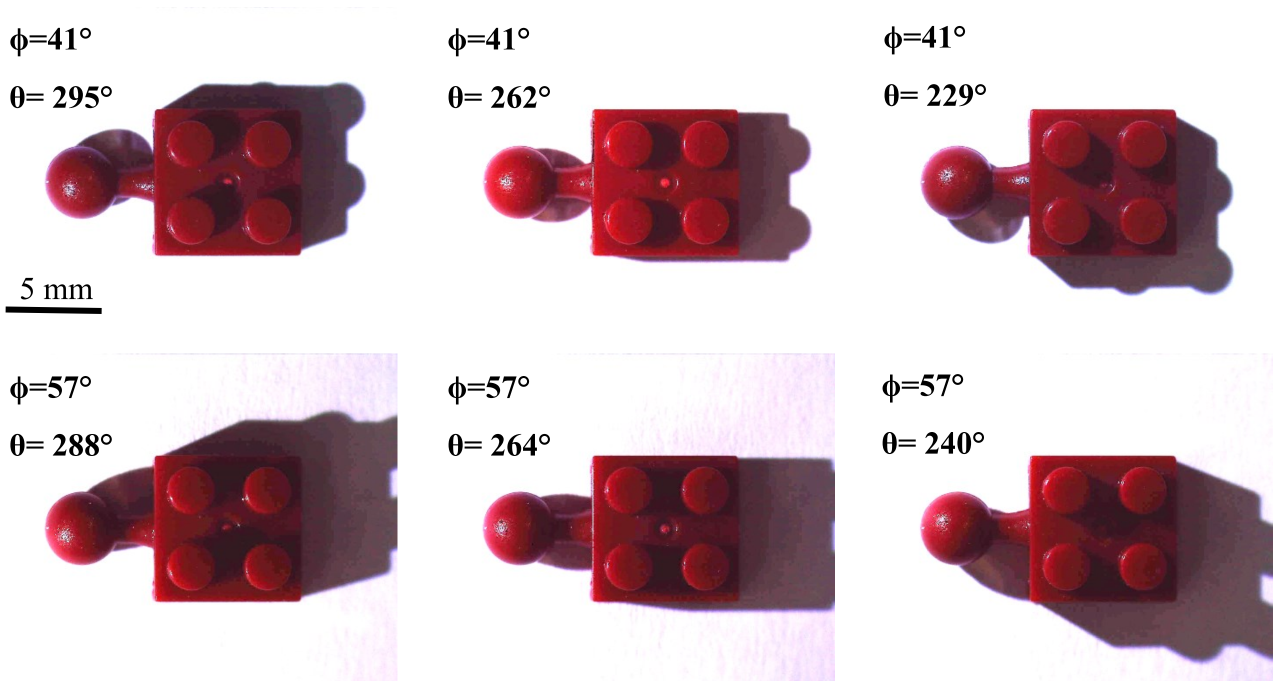


Figure 31: 6 samples of a plastic brick observed through the RTI program.

As mentioned earlier, our visual perception of a subject depends on various elements. Two of these elements on which the RTI technique is based are the shadows and highlights. Upon comparing conventional digital photos with RTI images, our visual inferences of a surface are improved due to the shadows and highlights associated with RTI technique.

Figure 32 demonstrates the type of details that an RTI image can discern. Our perception of elevations and depressions on a surface can be markedly enhanced through shadows and highlights. While conventional photography illuminates these features evenly and therefore shows no depth and looks flat, RTI images emphasize the 3D feature of a subject. As can be observed in the northerly illuminated Figure 32 (a), the southerly shadow cast by the stud indicates that the studs are at a higher elevation than the top face of the brick. There are also highlights on the north edges of the studs that indicate a faceted or rounded edge. Conversely, the same incident light highlights the south side of the central hole and thus differences in the direction of the highlights can discern the visual differences between elevations and depressions. At specular highlights such as the surface of the ball (Figure 32 (b)) the angle between surface normal and the incident beam approximately equals the angle between the surface normal and reflected beam that reaches the camera, and thus the orientation of these points can qualitatively be estimated with respect to the direction of the incident beam. In addition, Figure 32 (c) shows how a vertical facet can be detected through the shadow it casts on the background when being illuminated by the light source in the opposite direction.

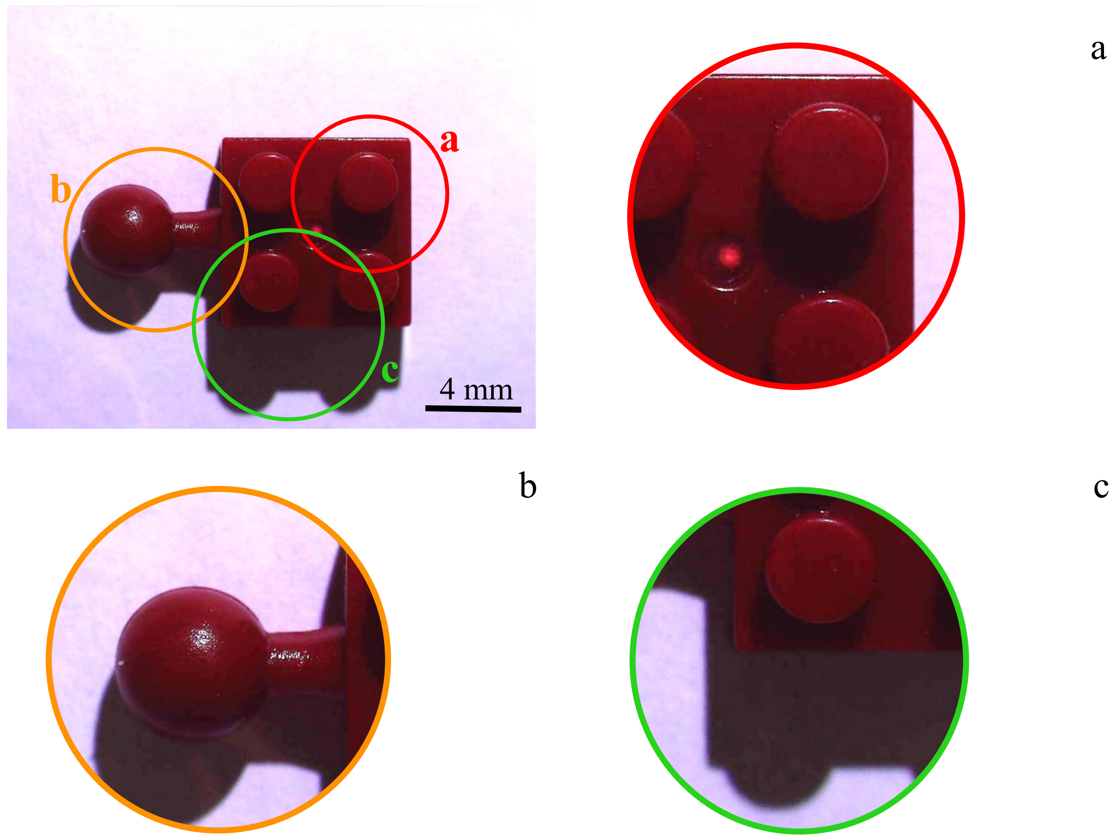


Figure 32: Representation of the information that an RTI image can provide. Illumination is from the north.

4.1.2. Noise

One of the points connected with images in RTI is the exposure to noise introduced by photography and lighting equipment. Noise in images can generally be perceived as the lack of light and color uniformity in the images, especially in the dark regions. Camera settings, sensor type and temperature, and the inherent properties of light are considered the main sources of image noise. However, one of the environmental factors involved in noise formation, especially in this study, is the light source.

The brightness intensity of NeoPixels used in this study as light source is estimated to be at 550 to 700 Millicandelas range for red LED, 1100 to 1400 Millicandelas for green and 200 to 400 Millicandelas for blue.

Hence, the nominal light power calculated for each LED of the NeoPixel is estimated to be approximately 6.5 lumens across an angle of 120°, which is less than the power of conventional microscope LED light sources [31]. Therefore, in a lower level of incident light, a decrease in the input signal can reduce the signal-to-noise ratio and thus the image appears to have more noise.

To reduce the problems introduced by noise, one approach is to weight the captured images. Since the noise present in the images is randomly distributed, averaging over multiple illuminations can reduce the appearance of the noise and also balance out the brightness. In this case, only one image has been recorded each time the scene is illuminated from a direction. Therefore, the conventional averaging technique cannot be implemented. To combat this, first assume that each noisy captured image *"f"* is represented as:

$$f_i(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix} \tag{10}$$

Where *(x,y)* represents the pixel coordinates, the image number is denoted by i, and the three red, green and blue channels are respectively given by *r(x,y)*, *g(x,y)* and *b(x,y)*. Next, instead of taking the average of multiple images captured from the same viewpoint, the average of all the stacked images, each of which has been recorded under a different lighting direction is calculated. The average image can be represented by "G" as follows:

$$G(x,y) = \frac{1}{N}\sum_{i=1}^{N} f_i(x,y) \tag{11}$$

Where N is the total number of images, which in this case it is equal to 50. Figure 33 shows the MATLAB algorithm to acquire the average image. To this end, first the total intensity of the all the images in all 3 color channels for each individual pixel is determined, and then the values are divided by the number of images.

```
avg_image=zeros(y_size,x_size,3);
total = 0;
 for j=1:4
    for i = 1:theta_table(j)
        avg_image = avg_image + squeeze(image_stack(:,:,i,j,:));
        total =total+1;
    end
end
avg_image = avg_image/total;
```

Figure 33: Part of the MATLAB program to average all the catured images.

The idea is to produce a composite image *h* consisting of a weighted sum of the (noisy) individual image *f* and lower noise average image *G*:

$$h_i(x,y) = weight * f_i(x,y) + (1 - weight) * G(x,y) \tag{12}$$

Here h_i is the *i*-th composite image, and *weight* is a user-defined variable to specify the sharing fraction of the average and the noisy image in the final denoised image. This definition modifies the display image in Figure 26 and the new algorithm is given in Figure 34. The *weight* variable could be specified using the same key press algorithm defined for Theta and Phi in Figure 27 and Figure 28. The highest value the weight can hold is 1, which thereby no portion of the mean image is added and the composite image is 100% the individual image.

```
add_image = squeeze(image_stack(:,:,theta,phi,:));
display_image =  weight * add_image + (1-weight) * avg_image;
imshow(display_image)
```

Figure 34: Part of the MATLAB program to add the average image partially to the images called from the image stack.

Figure 35 shows: (a) the nineteenth photo taken from three spherical beads, (b) the average of 50 recorded images and (c) the 50/50 weighted image of the average image and the captured image. Using only the individual image (a), parts of the surface are not visible and the surface is noisy. Using only the average image (b), the 3D details of the subject are not fully displayed, and the shadows and highlights present in the individually captured images have been removed. The more average the user adds to the captured image, the less noise will appear in the displayed image; however, the appearance of shadows and highlights begins to decrease.

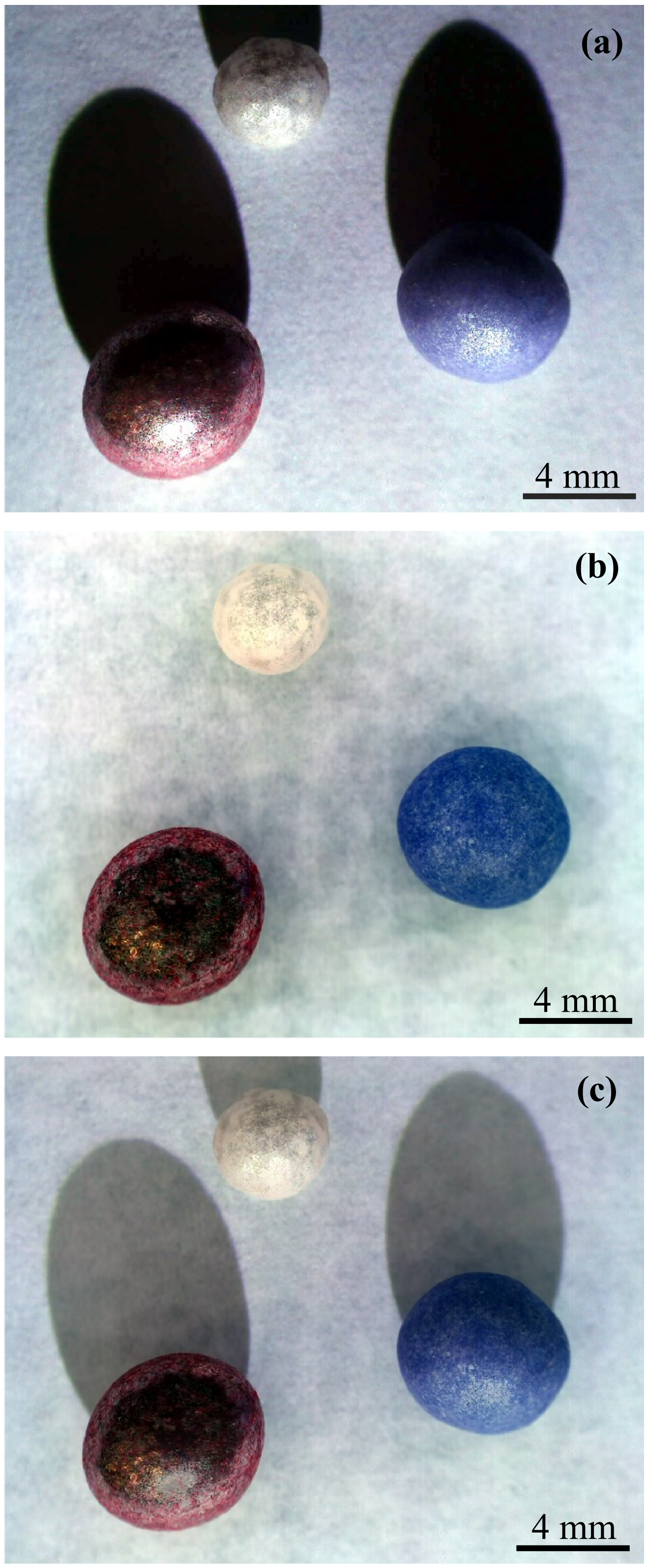


Figure 35: (a)A single captured image of three spherical beads. (b) Average of all the captured images. (c) 50/50 contribution of the captured image and the average image.

4.2. HSV Image

The brightness of Lambertian surfaces is reliant on the angle of the incident light relative to the surface normal. Therefore, by determining the brightest state of each point on a surface, and the angle of the incidence light at which the maximum brightness occurred, a pictorial estimate of the orientation of the points on the surface can be made. The proposed method to interpret a three-dimensional structure from two-dimensional images is derived from the assumption that 1) the LEDs have the same luminous power and 2) the relative distance of the LEDs from the subject is approximately the same. In the case of ~8 mm Nanoblock from the previous section, using a 15cm dome, the maximum radial distance change is ~ ±4%.

4.2.1. Maximum Brightness Angles

The first prerequisite is to identify the brightest state of each pixel among the captured images and determine the corresponding direction information of the incident light. To this end, first the MATLAB program in Figure 24 is modified to store the grayscale intensity of the images into a stack called *gray_stack*, as depicted in Figure 36.

```
image_stack =zeros(y_size,x_size,theta_max,phi_max,3);
gray_stack =zeros(y_size,x_size,theta_max,phi_max);

for i=1:num_pics
    ith_image = imread(file_names(i,:));
    image_stack(:,:,theta_val,phi_val,:)=ith_roi;

    gray_stack(:,:,theta_val,phi_val)=rgb2gray(ith_roi);

    theta_val = theta_val+1;
    if theta_val > theta_table(phi_val)
        phi_val=phi_val+1;
        theta_val = 1;
    end
end
```

Figure 36: Part of the MATLAB program to store grayscale version of the images along with RGB images.

The *gray_stack* is a $m \times n \times t \times p$ array that contains the grayscale intensity of all the $m \times n$ captured images along with their illumination direction information. The number of rows and columns of $m \times n$ arrays corresponds to the resolution of the images. The 3rd & 4th indices of the grayscale matrix (t & p) correspond respectively to θ (longitude index) and ϕ (colatitude index) values. Next, the ϕ and θ that produce the maximum grayscale intensity at each pixel can be found using the "max()" function, as shown in Figure 37. The "max()" function returns 2 items: an $m \times n$ image (*max_image*) with each pixel containing its maximum intensity, and the location of the maxima within the gray stack (*theta_phi* stored as linear indices). The individual maximum θ and ϕ values can be extracted from the linear indices using the "ind2sub()" function. This results in two $m \times n$ arrays (*theta_max* & *phi_max*) which respectively store which θ and ϕ values produce the maximum intensity for each pixel.

```
theta_max =18;
phi_max=4;

[max_image, theta_phi]= max(gray_stack,[],[3 4],'linear');

[~,~,max_theta,max_phi]=ind2sub([y_size x_size theta_max phi_max],theta_phi);
```

Figure 37: The algorithm to extract maximum values of pixels along with the direction info of the incident beams of light.

The acquired arrays of *max_theta* and *max_phi* store the LED indices (1,2, 3, etc.) and not the angular values corresponding to each LED. Therefore, the next step is to convert the indices in *max_theta* and *max_phi* arrays to the corresponding angular coordinates. As shown in Figure 38, angles assigned to θ are calculated according to the number of LEDs at each ϕ level and the distribution of the LEDs over a 360 degrees range of angles, and the result is stored in an array called *lookup_theta*. The angles assigned to ϕ are pre-measured according to the built hemisphere in an array called *lookup_phi*. Through the proposed arrangement, the previous two arrays of *max_theta* and *max_phi* can be modified so that each of their elements represents the azimuth angle and polar angles corresponding to each pixel's maximum intensity (*max_theta2* & *max_phi2*).

```
phi_table =[73 57 41 25];
theta_table = [18 15 11 6];
for i =1:phi_max
  for j = 1:theta_table(i)
      lookup_theta(j,i)=round(360*(j-1)/theta_table(i),1);
      lookup_phi(j,i)=phi_table(i);
 end
 end

  for i =1:y_size
  for j = 1:x_size
      ii = max_theta(i,j);
      jj = max_phi(i,j);
      max_theta2(i,j)=lookup_theta(ii,jj);
      max_phi2(i,j)=lookup_phi(ii,jj);
 end
 end
```

Figure 38: Converting angular indices to angular coordinates.

4.2.2. HSV Image

Regardless of the maximum value of each pixel, the angles at which the maximum occurs can reveal the geometric properties of the subject. The primary purpose of this section is to provide a visual perception of the orientation of different points of a surface based on the angle of the incident light.

While an RGB image is composed of three channels of red, green and blue, and each channel contains 256 tones of each primary color, an HSV image is defined using three dimensions of hue (H), saturation (S) and value (V). Hue represents the color of the image from a continuous 360-degrees spectrum around a circle, where the colors start from red, passing through yellow, green and blue and come back to red again at the end. Saturation indicates the extent to which colors are diluted with white, maximum saturation is when the colors have the highest purity and low saturation corresponds to a whitish representation of the colors. Value corresponds to the brightness of the colors. (for more information regarding HSV color spaces see [32]).

For each pixel the hue (H) will be set to the maximum azimuth angle (θ) and the saturation (S) will be set to maxima polar angle (ϕ). The HSV colormap in MATLAB assigns numerical information of pixels ranging from 0 to 1. To convert angle information to visual concepts, first the numerical range of angles is scaled to 0 to 1 as shown in Figure 39. There is a 360° range of azimuth angle in θ and there is a 48° range of polar angles in ϕ (from ϕ_1=73 to ϕ_2=25). Here, the third component of the image value (V) is to define by a $m \times n$ array named *val_image* and is set to one (maximum brightness).

```
max_theta2 = max_theta2/360;
max_phi2 = (max_phi2-25)/ 48;
val_image =ones(y_size,x_size);
```

Figure 39: The three components of the HSV image: hue, saturation and brightness.

The arrays depicted in Figure 39 are the three required components to build a HSV image where each pixel exhibits pictorially its spatial orientation with respect to the location of the light sources. To convert these components into a three-dimensional array, it is required to concatenate the arrays that are assigned respectively to hue, saturation and brightness and then convert the resulting array to an RGB image as depicted in Figure 40.

```
HSV_image =cat(3, max_theta2, max_phi2, val_image);
RGB_image= hsv2rgb(HSV_image);
imshow(RGB_image)
colormap hsv
```

Figure 40: Part of the MATLAB program that represents how the HSV image is built.

The corresponding output image, called the HSV image, provides inferences about the orientation of a surface. Subjects with known geometrical orientations were tested to examine and evaluate the proposed HSV image. Figure 41 shows (top) the three spherical beads ranging in size from 4 mm to 6 mm imaged previously, and (bottom) their HSV image. A small HSV spectrum ring is shown in the upper left corner to indicate the direction of incoming light. According to the distribution of LEDs over the hemisphere, the surface of the beads is expected to be divided into colored slices. The saturation decreases moving to the higher parts of the spheres, because smaller polar angles (higher LEDs) indicate lower saturation and greater polar angles (lower LEDs) are assigned to higher saturation.

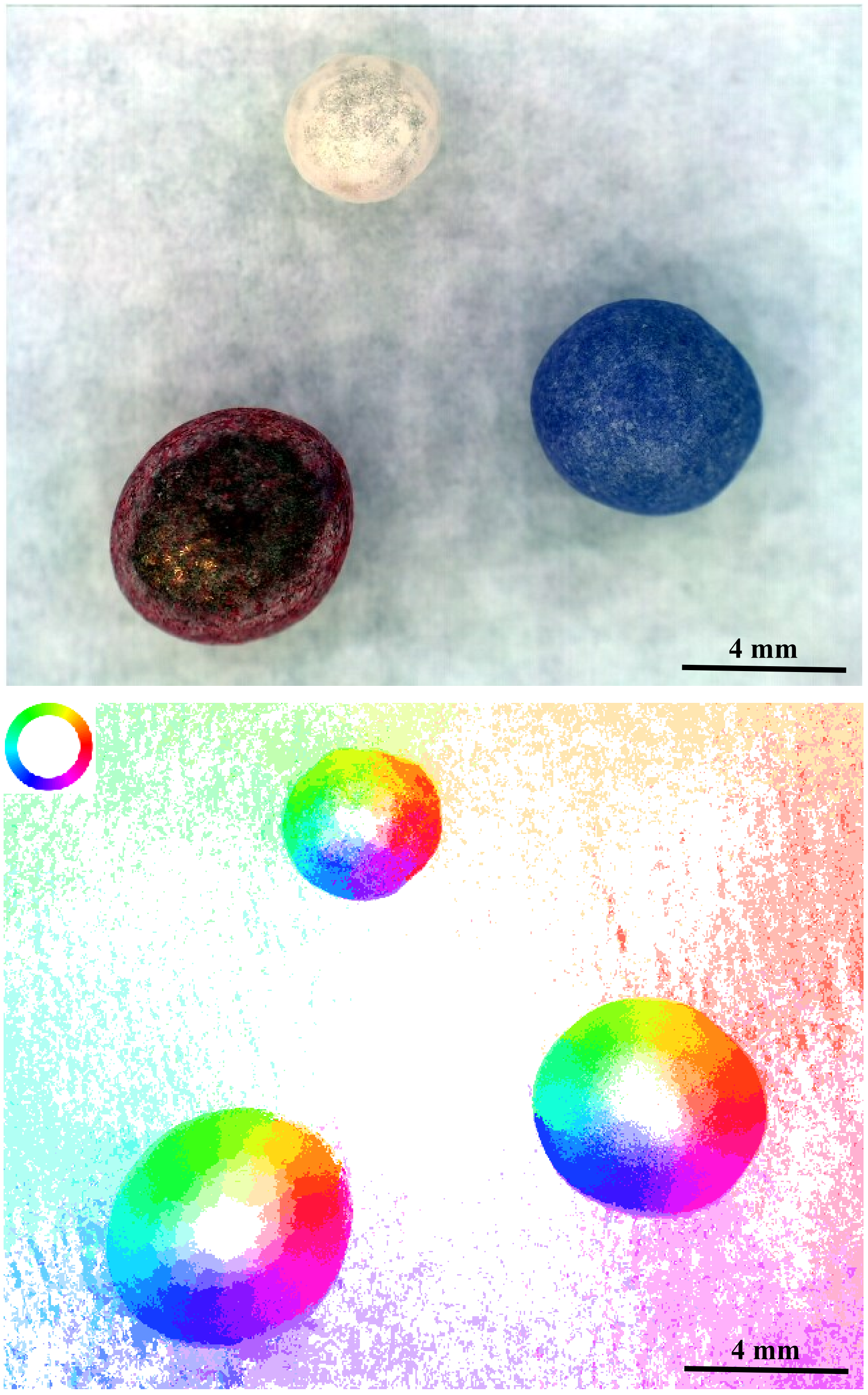


Figure 41: Top: The average image of three spherical beads. Bottom: The HSV image of three spherical beads, used to represent the surface orientations. A small HSV ring in shown in the upper left corner of the bottom image to indicate illumination direction.

As expected, the surface of the beads is segmented into colored slices that face different LEDs. The number of the slices is proportional to the number of LEDs. Looking at the largest bead, the most central part of the sphere in the image, which is actually the top region of the bead, is exposed to the light of the High LEDs on the hemisphere (ϕ_4), whose light hits the surface at the polar angle of 22.5°. This is the minimum angle ϕ that is assigned the least amount of saturation, and therefore these regions appear white. By moving towards the outer rings of the sphere, the saturation level of the colors as well as the number of color slices increases.

The image background is a flat surface whose normal is perpendicular to the image plane and therefore is assumed to meet the maximum value when illuminated by LEDs closest to the pole (High LEDs). As expected, most of the background surface is white; however, background areas that are closer to some of the light sources appear in color since nearer light sources have more light hitting the surface. This difference arises from varying distances of surface points from light sources and is not related to the orientation of the surface.

The HSV image of a small chain with 8 mm oval links is presented in Figure 42. We begin the image interpretation process with the white regions of the subject that are maximally illuminated by the High LEDs on the hemisphere (ϕ_4). Therefore, these points on the surface are close to parallel to the image plane. The surfaces in pure green and purple that are located on two different sides of the white regions are illuminated by the Low LEDs (ϕ_1) from the north and the south. The pure purple surfaces correspond to incident lights coming from the LEDs on the south-southwest of the hemisphere with respect to the viewpoint. Similarly, the green surfaces have been illuminated by the LEDs on the north-northwest.

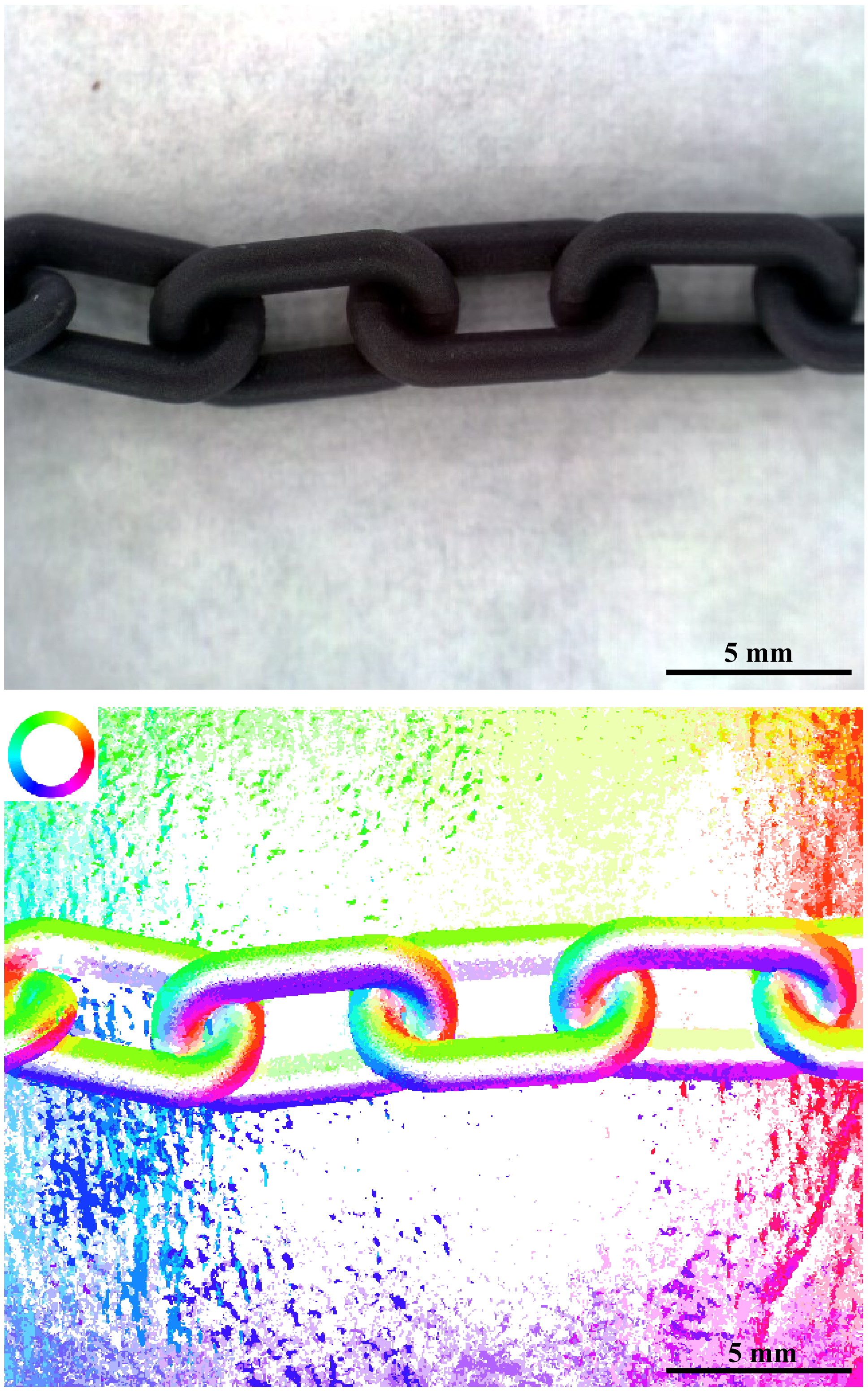


Figure 42: Top: The average image of a chain. Bottom: The HSV image of the chain, used to represent the surface orientations.

4.2.3. Thresholding

In attempting to recover information from the HSV image, an undesirable image artifact is that the flat areas on the image that should be displayed as white, are brightly colored. This occurs due to the surface roughness of flat regions or the relatively closer distance of the points to some LEDs.

Flat surfaces would be brightly illuminated by light sources at or near the vertical, while sloped surfaces would be less illuminated from these directions. The 6 grayscale images corresponding to the 6 High LEDs, are extracted from the created grayscale stack, and stored in a new array named *top_stack*, as shown in Figure 43, which is a $m \times n \times 6$ array. The reason for converting RGB images to grayscale images is that the mask algorithms used only function on grayscale images.

Thresholding is an image segmentation technique that is used to segment the image into two regions; in this case, one is the pixels with maximum brightness, and the other is pixels with reduced brightness. To apply the thresholding to this case, first, the maximum value of each pixel out of 6 images of ϕ_4 is determined and is stored in a $m \times n$ array called *max_top_image*. Flat surfaces tend to exhibit high values of brightness when illuminated from a polar angle close to 0° which is close to the surface normal. A user-settable variable, *threshold*, is defined to provide the criteria for evaluating the intensity of the pixel. Any pixel where *max_top_image* > *threshold* causes that pixel's saturation to be set to 0 (turned white). In MATLAB a Boolean $m \times n$ array mask is used to selectively desaturate all image regions with high top brightness above the threshold as shown in Figure 43:

```
top_phi = 4;
top_theta= theta_table(top_phi);
top_stack = gray_stack(:,:,1:top_theta,top_phi);
max_top_image =imadjust(max(top_stack,[],3));

mask = max_top_image > threshold;
val_image(mask) = max_image(mask);
sat_image(mask) = 0;
```

Figure 43: Thresholding to reduce the brightness of flat points on a surface.

Figure 44 illustrates the effect of thresholding on the HSV image of the chain. The image at the top is the original HSV image with threshold variable at 1. In this case, no pixels are desaturated. The image in the middle is when the threshold variable is at 0.7 whereas the brightness of the flat surfaces is reduced without compromising the brightness of the subject. The image at the bottom is the HSV image with an excessive threshold variable at 0.2, which omits HSV information.

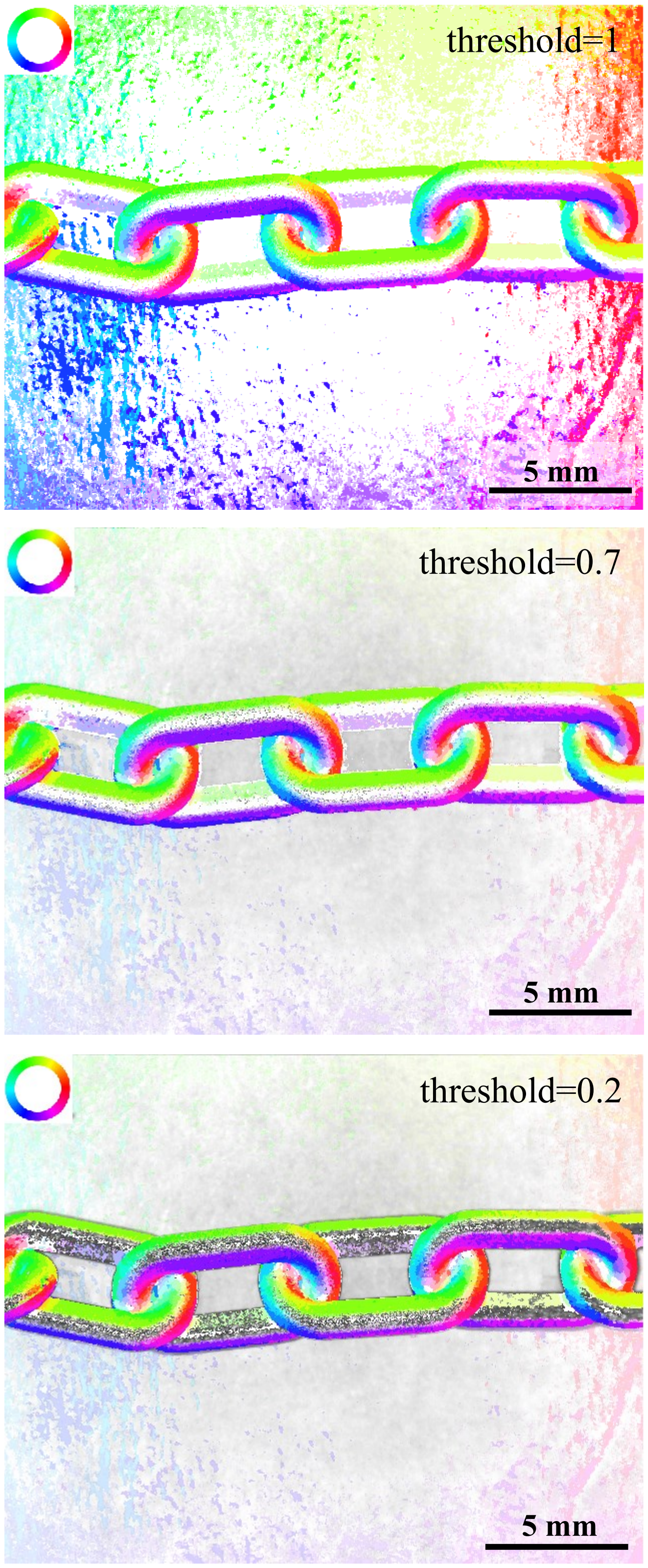


Figure 44: The effect of thresholding on the HSV image. Top: high threshold. Middle: reduced threshold. Bottom: low threshold.

5. MEMS

This chapter presents a general introduction to Micro Electro-Mechanical Systems (MEMS), a categorization of MEMS devices, and examples of MEMS applications. Micro-fabrication processes are reviewed and commonly used materials for MEMS technology are summarized. Next, the optical behavior of MEMS devices is presented, and lastly the experimental setup required to observe MEMS devices is examined.

5.1. MEMS Background

Micro electro-mechanical systems are micro-level devices that generate or sense physical quantities that can affect the macro-level world. MEMS devices incorporate components with dimensions between 1 to 100 μm and are fabricated using micromachining processes in which thin layers of materials are selectively etched away to release a three-dimensional structure. In a similar manner to integrated circuit technology, silicon is the primary material used in MEMS fabrication due to its superior intrinsic properties.

Since the mid-1950's silicon has formed the foundation of microelectronics and during the last few decades has extended its boundaries to include fabrication of micro electro-mechanical systems [33]. The substrates upon which microsystems are built are large wafers made from the pure single-crystal form, typically a few hundred micrometers thick (see [34]).

The advent of silicon micromachining made it possible to manufacture complex three-dimensional components of MEMS devices. Shifting towards micro-scale devices yields a number of advantages. There is a significant reduction in weight and volume, and as a result the material requirement is minimized. The power requirement is also reduced in comparison to macro-scale counterparts. Micro-scale fabrication is also adapted to mass-production, and as a result, the cost for a single device could be markedly reduced [35].

MEMS micromachining process falls into two main categories: The process of micromachining MEMS devices from thin structural layers deposited on the silicon substrate is called “surface micromachining”, and when the micro-structure is machined by etching away the silicon substrate, the process is defined as "bulk micromachining". Surface micromachined MEMS devices will be examined in this thesis.

The first MEMS device dates back to a microscopic tuning device for radios invented by an American engineer, Harvey C. Nathanson, in 1967 and with the passage of time has evolved into a broad categorization of MEMS devices [36]. MEMS devices now encompass a variety of functions, mechanisms and applications. With respect to their interaction with surroundings, MEMS devices are classified into two general groups: actuators and sensors. A sensor converts a non-electrical quantity to an electrical quantity (e.g. motion or displacement to change in capacitance), while an actuator converts electrical input into non-electrical output (e.g. current to thermal). MEMS sensors and actuators form a broad spectrum of devices from pressure sensors, accelerometers, ink-jet printers and gyroscopes to radio-frequency MEMS, optical devices and micro-fluidic components.

As a simple example, Figure 45 shows the top view of a two-arm MEMS thermal actuator which is a commonly used component in MEMS devices such as microsensors or microswitches. The thermal actuator makes use of the fact that the thermal expansion of a narrow arm will be relatively greater than the thermal expansion of a wide arm when a same voltage is applied (this is due to the higher current density and thus higher temperature). The narrow arm causes the structure to deflect towards the wide arm as shown in the Figure 45. Voltages of a few volts and currents of a few milliamps produce several microns of motion.

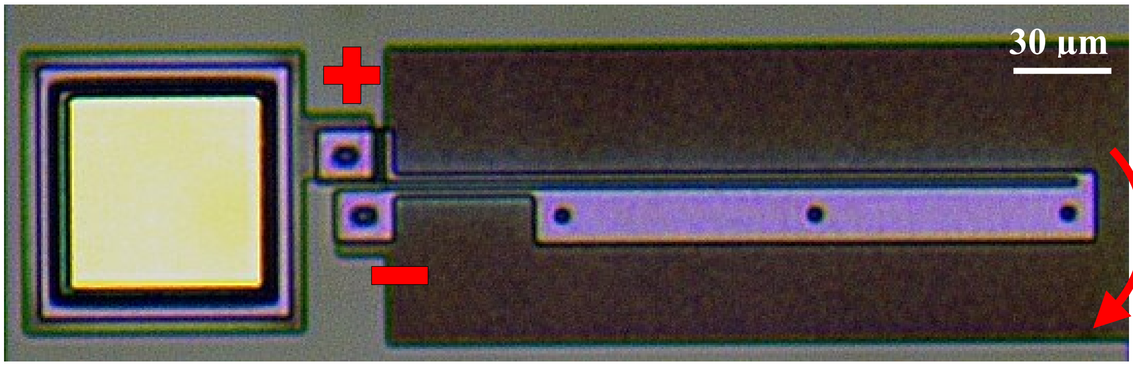


Figure 45: Microphotograph of an asymmetric MEMS thermal actuator.

5.2. Surface Micromachining & MUMPs

Depending on the desired output microstructure, surface micromachining can generally entail patterning techniques and additive and subtractive processes. Patterning concerns the process of coating a thin uniform photosensitive polymer film on a structural surface and subsequently etching the film selectively based on the pattern of a mask layer exposed to lighting. The pattern of the photosensitive film is reproduced in the layer underneath through an etching agent exposed to the entire workpiece. The remaining photosensitive film which protected the structural layer is inert against the etching agent and later requires an additional etching stage to be removed.

Multi-User MEMS Processing system (MUMPs) is a popular fabrication technology that eases the design and fabrication process by supplying documented standardized design rules on shared silicon wafers that insure the reliable performance of the devices. PolyMUMPs is one of the subsets of MUMPs technology that can be implemented in a similar manner as surface micromachining (see PolyMUMPS Design Handbook [37]). PolyMUMPs utilizes an established micromachining procedure in which the alignments and thickness of layers along with patterning, masking, depositing and etching stages are predetermined.

5.2.1. Process Overview

PolyMUMPs is an additive fabrication processes consisting of stacking up to seven successive layers of thin films with a maximum thickness of 7.85 μm, as shown in the cutaway illustration in Figure 46. There are two types of layers: structural layers made from polysilicon and sacrificial layers made from oxides. These are analogous to structural and support layers in 3D printing.

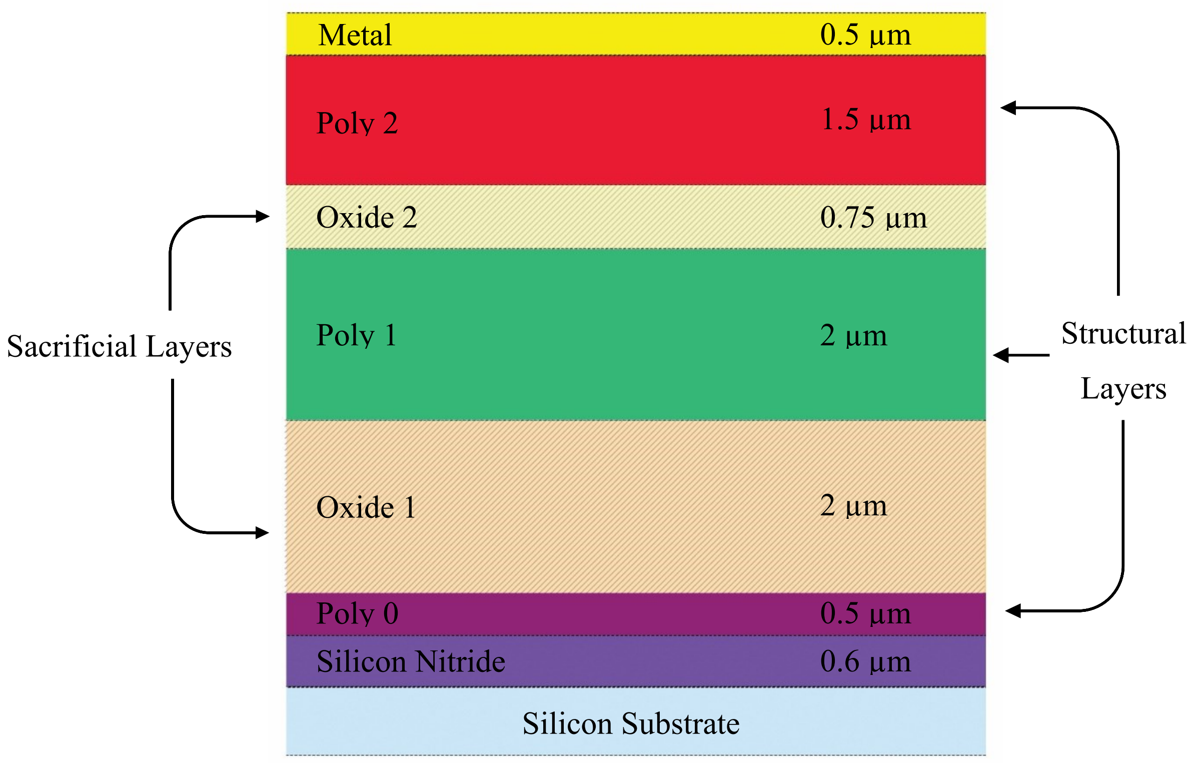


Figure 46: PolyMUMPs Layers.

To illustrate the process, the two-arm thermal actuator shown in Figure 45 is used as an example. The cutaway views of the process when the thermal actuator is cut along the dotted line in Figure 47, are shown in the following images.

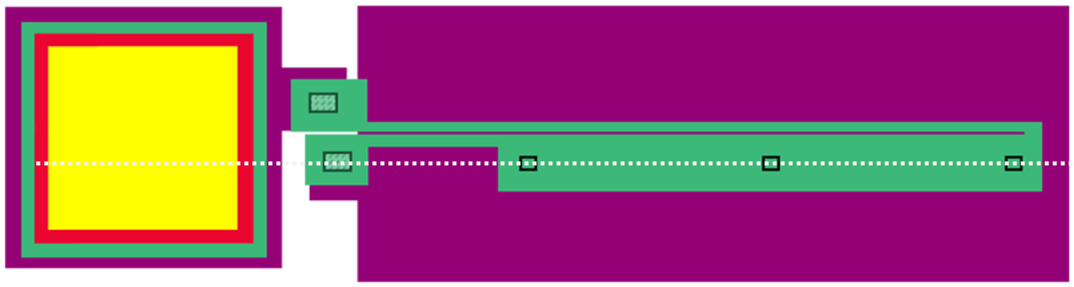


Figure 47:Two arm thermal actuator illustration.

A n-type (100) single crystal silicon wafer substrate is the first prerequisite, and it forms the lowermost structure (>100 µm thick). Then as shown in Figure 48 (a), a silicon nitride thin film (0.6 µm) is deposited on top of the silicon wafer to electrically isolate and mechanically support the silicon wafer. Using the same deposition process, a thin layer (0.5 µm) of polysilicon (POLY0) is grown on the nitride film to be used as an electrical ground plane (Figure 48 (b)). A photolithography technique is used then to pattern the POLY0, and subsequently POLY0 is etched according to the created pattern.

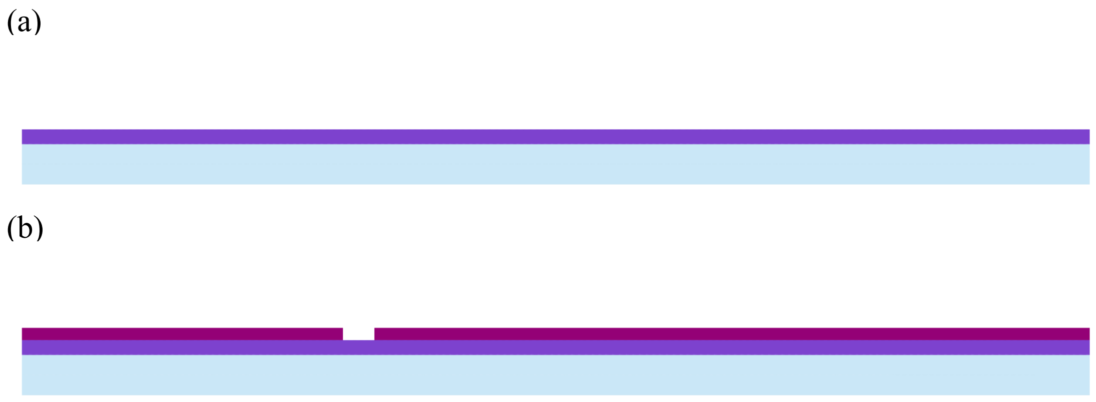


Figure 48: (a) Nitride (violet) deposition. (b) POLY0 (magenta) deposition and patterning.

The first sacrificial (space-holding) layer is silicon dioxide (OXIDE1) (2 µm) which is a common material in surface micromachining. OXIDE1 is deposited on the POLY0 layer using a deposition technique and is subsequently patterned in a second photolithography step as shown in Figure 49. Patterning and etching OXIDE1 contributes two purposes: to form dimple and anchor (ANCHOR1) holes to be filled with the subsequent polysilicon structural layer. Dimples are 0.75 µm deep etched pits that will be filled with polysilicon with the purpose of reducing the large adhesion force at microscale between two flat surfaces. Anchors are formed by etching completely through the oxide layer so that an electrical connection will be made between polysilicon layer and the substrate.

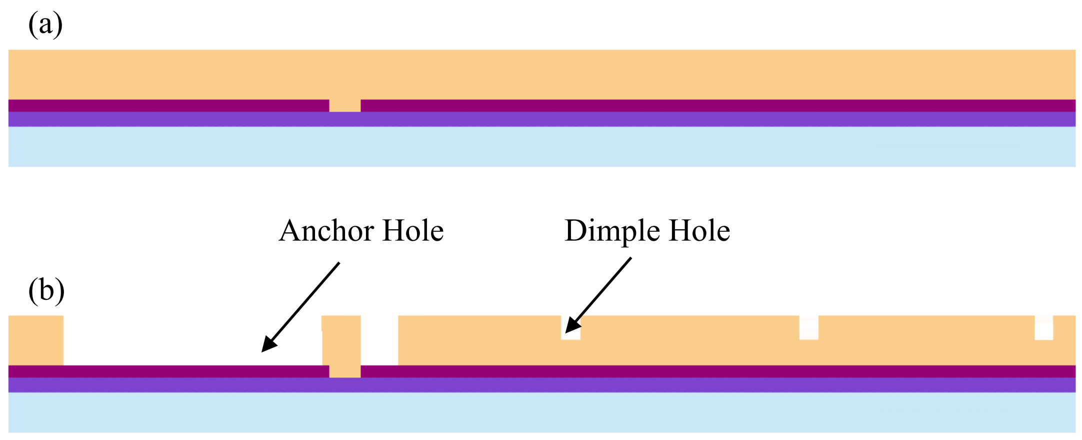


Figure 49: First silicon oxide layer (OXIDE1) deposition and patterning.

POLY1 is the first moveable structural layer (2 µm) which is deposited upon the OXIDE1, as shown in Figure 50 (a). POLY1 fills the anchor (ANCHOR1) and dimple holes, and since the sacrificial layers are to be removed at the end of the process, the dimples and anchors will emerge as depicted in Figure 50 (b).

Figure 50 shows a simple cantilever that is made by anchoring one end of a long beam of POLY1 (green) to the POLY0 (purple). Where there is no anchor, the POLY1 will be 2 µm above the surface after the OXIDE1 is removed. There are also three dimples that form bumps on the bottom of the POLY1 and lessen the contact area between POLY1 and POLY0 underneath, and as a result prevent the long beam from sticking.

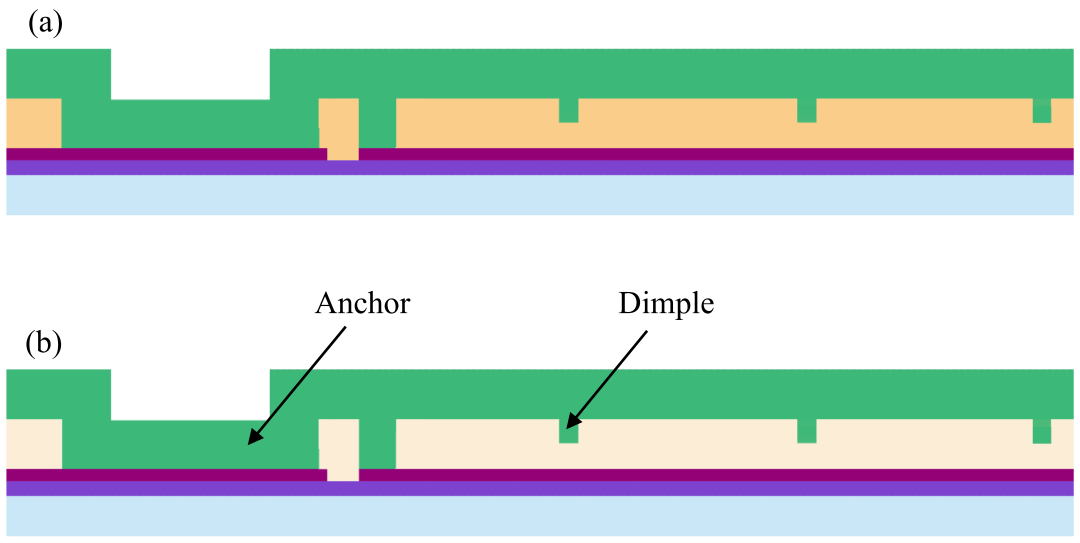


Figure 50: (a) POLY1 (green) deposition. (b) How POLY1 looks like when OXIDE1 is removed.

Next, in a third photolithography step, POLY1 is patterned and etched away selectively, as shown in Figure 51 (a). Then, upon the POLY1, a thin layer (0.75 µm) of silicon oxide (OXIDE2) is deposited, as illustrated in Figure 51 (b), and then patterned (Figure 51 (c)).

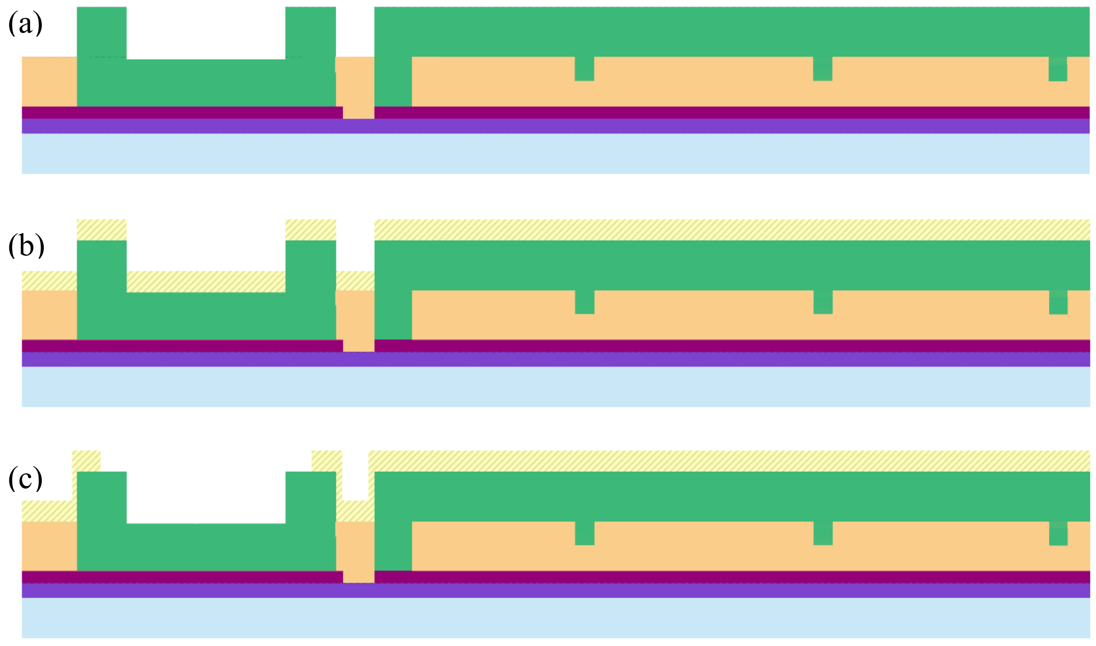


Figure 51: (a) POLY1 (green) patterning and etching. (b) OXIDE2 (yellow) deposition. (c) OXIDE2 patterning & etching.

Depending upon the particular design requirements of a MEMS device, the OXIDE2 is etched to serve two functions: POLY1_POLY2_VIA and ANCHOR2. These functions are fulfilled with respect to the next layer POLY2. POLY2 is the last movable structural layer (1.5 µm) and is deposited and then patterned using photolithography, as shown in Figure 52. POLY1_POLY2_VIA allows the POLY2 to contact the POLY1 while ANCHOR2 allows the POLY2 layer to contact the substrate. Where the OXIDE2 remains intact, the POLY2 will be suspended at a distance (0.75 µm) from the POLY1. In this particular example, OXIDE2 is just patterned to provide the connection between POLY1 and POLY2 (POLY1_POLY2_VIA), and POLY2 has no need to be anchored to POLY0 (ANCHOR2).

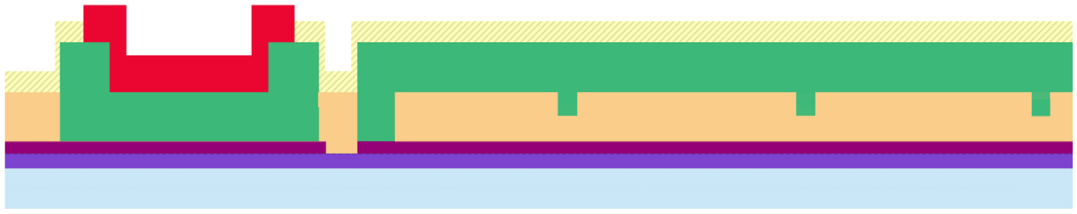


Figure 52: POLY2 deposition and selectively etching.

What remains to be deposited is the electrical connections through a thin metal (yellow) layer (0.5 µm), as shown in Figure 53. The metal layer can also be deposited to be used as reflectance layers required in optical MEMS. The micromachined device with suspended structures will be released by removing the sacrificial OXIDE layers using Hydrofluoric acid (HF) solution (which does not remove the polysilicon and metal layers). This is analogous to dissolvable supports in 3D Printing.

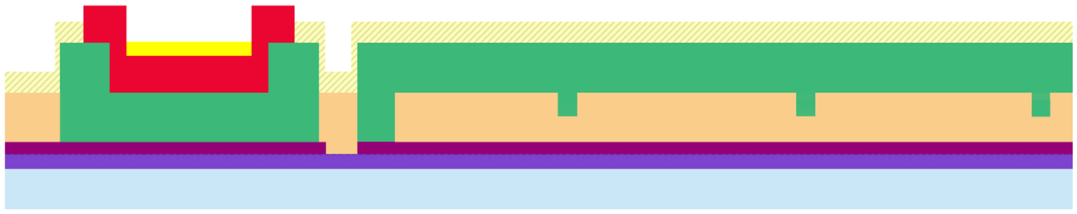


Figure 53: Metal (yellow) deposition.

To design effective and successful MEMS devices, design guidelines introduced by companies offering the MUMPs process must be followed. For example, in PolyMUMPs the minimum anchor size is 3.0 µm, and the ANCHOR1 boundary must be at least 4 µm away from the POLY0 boundary that enclosed ANCHOR1.

The figures above show pictorially the cross-section of MUMPs process for fabricating an asymmetric thermal actuator. In this case, only POLY1 is used as a structural layer. The cutaway illustration of the thermal actuator (Figure 53) shows the cold arm with three dimples and the probe pad with no electrical connection to the cold arm (the connection is made through the hot arm which cannot be seen in the cutaway plane).

5.3. MEMS Chip

The available design area varies depending on the capabilities of the foundry service. In the case of MUMPs, the MEMS devices are designed in a 5 mm × 5 mm area. The dicing or sawing stage separates each designed die after fabrication from the shared wafer.

Figure 54 shows the 68 PGA (Pin Grid Array) MEMS chip packaging used in the Dalhousie MEMS lab mounted on a ZIF (Zero Insertion Force) socket. The purplish PGA ceramic chip carrier with matte surface finish has dimensions of 27.9 mm × 27.9 mm × 2.0 mm and encompasses the 5 mm × 5 mm diced chip at its center. The gold wire bonding around the die provides an interface between the contact pads on the chip and electrical connecting pins on the ceramic package. Each electrical connecting pin of the ceramic package is routed to a ribbon cable for the external power supply.

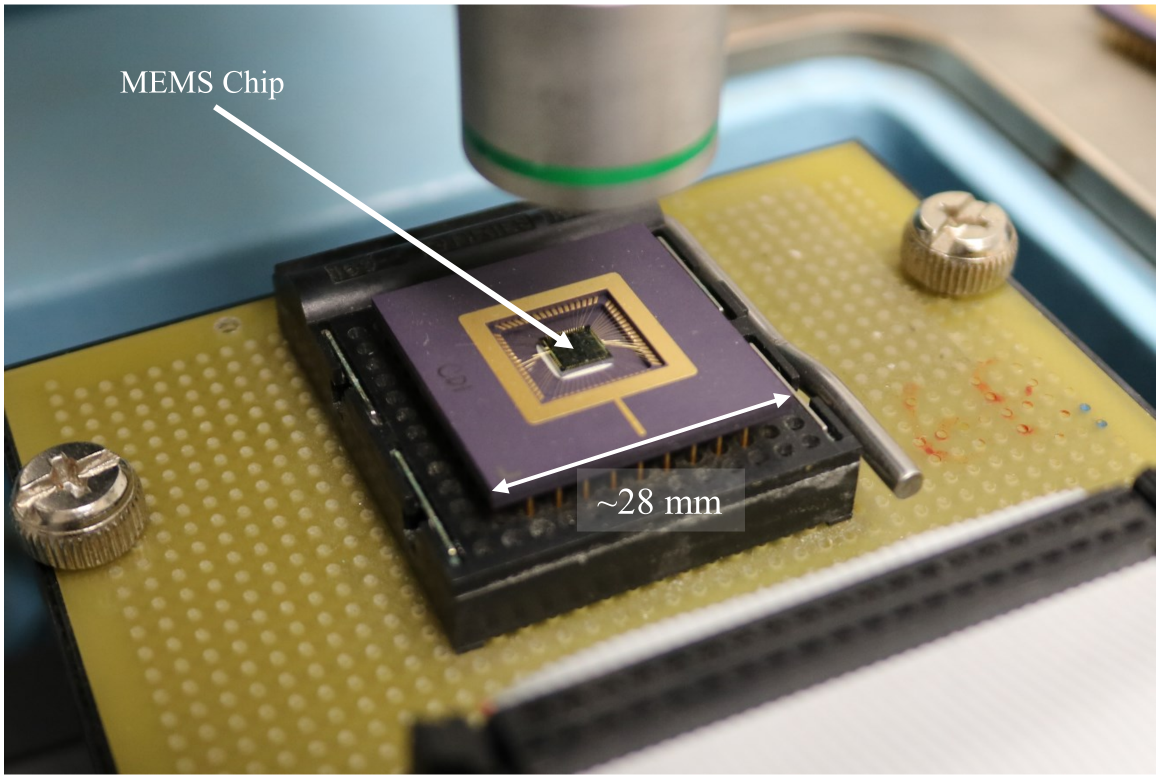


Figure 54: The 5×5-millimeter MEMS die inside a ceramic package (purple) mounted on a ZIF socket (black).

The most conventional approach to observe MEMS devices and their operation and performance is to use optical microscopy with light sources illuminating the chip at normal incidence. Hoping to find visual cues and information about the three-dimensional characteristic of MEMS devices, the RTI technique will be applied, however visual output similar to mm-scale RTI cannot be achieved due to the optical properties of MUMPs surface which are discussed in the following sections.

5.4. Optical Properties of MUMPs

As discussed in the previous sections, polysilicon is the primary structural layer of PolyMUMPs. Therefore, from the perspective of an optical microscope, MEMS devices are comprised of polysilicon structures that unfortunately exhibit similar optical properties and similar depths of field (a few μm), as well as optically different metal connections with higher reflectivity.

To apply the RTI imaging technique to the MEMS chip, light sources need to illuminate the chip at different incident angles other than vertical angle, as explored in Chapter 3 using a dome lighting structure. Attempts to illuminate the MEMS surface from different angles, using several 3D printed structures, revealed that an image as bright as coaxial illuminated images could not be achieved while the camera remains above the MEMS surface. Figure 55 shows a conventional image of a MEMS device selected to demonstrate different PolyMUMPs structures. This particular device was designed by Bruno Barazani [38] and is comprised of periodic suspended structures manufactured using POLY1 that drive a squeezer jaw (made of POLY2) to the left and right. The MEMS device shown in Figure 55 is labeled with examples of POLY1, POLY2, dimples, anchors. Note that only flat surfaces appear bright as they will specularly reflect light directly into the microscope objective. Conversely, sidewall facets from either the dimples, the anchors or layer transitions appear dark in coaxial illuminated image. This is because the angled surfaces specularly reflect light away from the microscope objective.

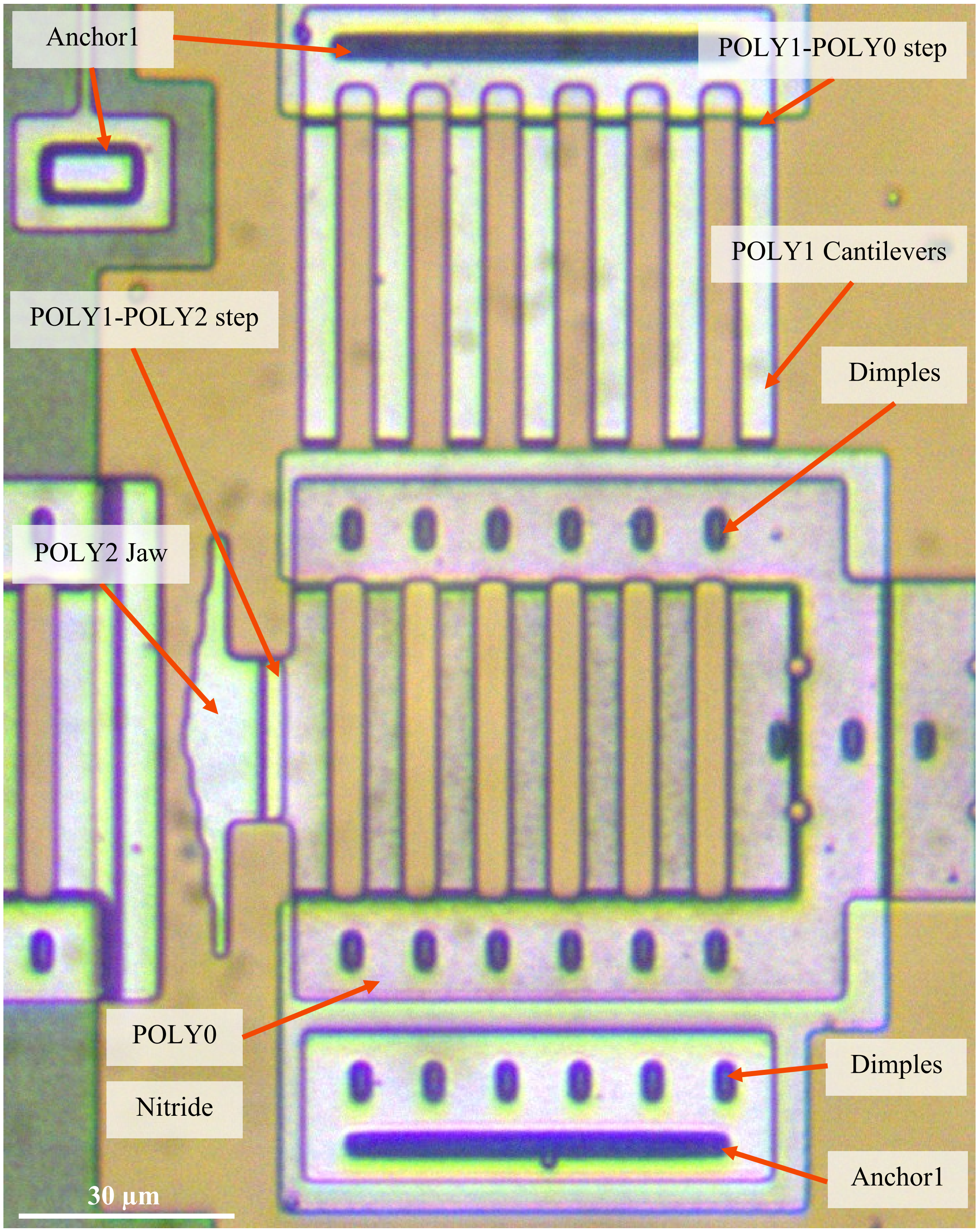
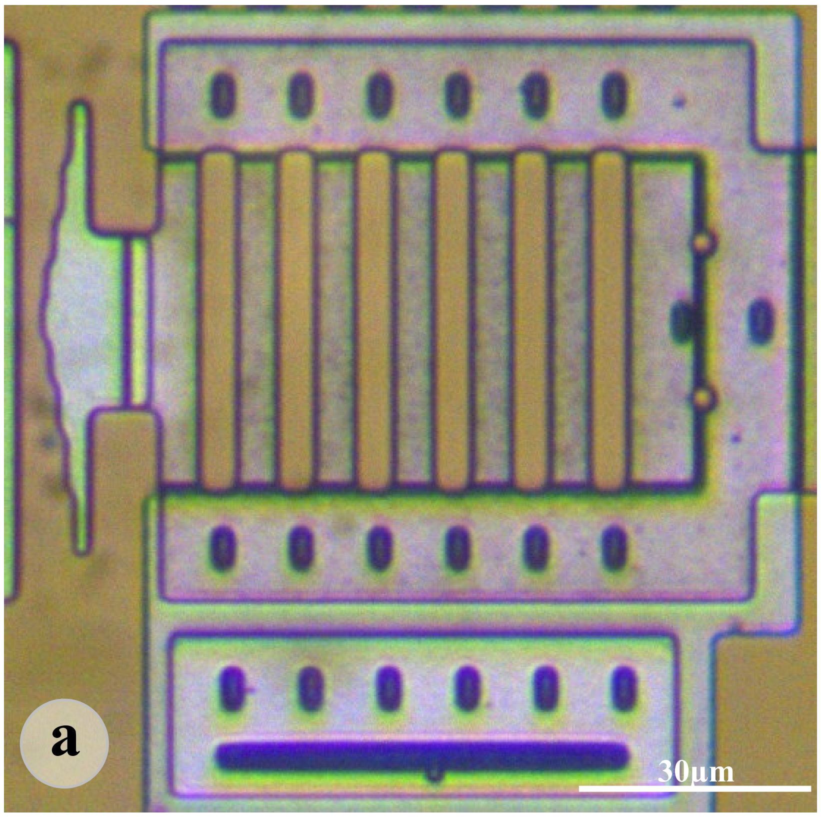


Figure 55: Conventional optical image of a MEMS device (cell squeezer) showing different structures. Note that non-flat features appear dark.

Figure 56 (a) shows a zoomed-in region of the same cell squeezer, and Figure 56 (b) is when the light source illuminates the same region at a relatively large polar angle (≈ 50°). The point of observation for both images (a & b) is the same. The primary difference between the two images is a significant reduction in overall image brightness, when the illumination angle has slightly tilted. This brightness reduction mainly arises from the dimness of flat surfaces that are entirely bright in conventional images. The only bright regions of the second image are sidewall facets from either the dimples, the anchors or a layer transition that appear dark in coaxial illuminated image. With angled illumination, these sidewall facets now reflect specularly light into the microscope objective, while flat regions reflect it away.

Figure 56 (b) and similar observations when the light source is located at different polar angles reveal the reflectance characteristics of the polysilicon surfaces. As mentioned in Chapter 3, diffuse surfaces scatter the incident beam in many directions, leading to the visibility of the subject from all points of view, while mirror-like surfaces have only a specular spike in a single direction and as a result observing mirror-like surfaces depends on the point of observation. From the viewpoint dependent reflectance characteristic of MEMS chip, we can conclude that the polysilicon thin films are shiny and reflect light specularly.



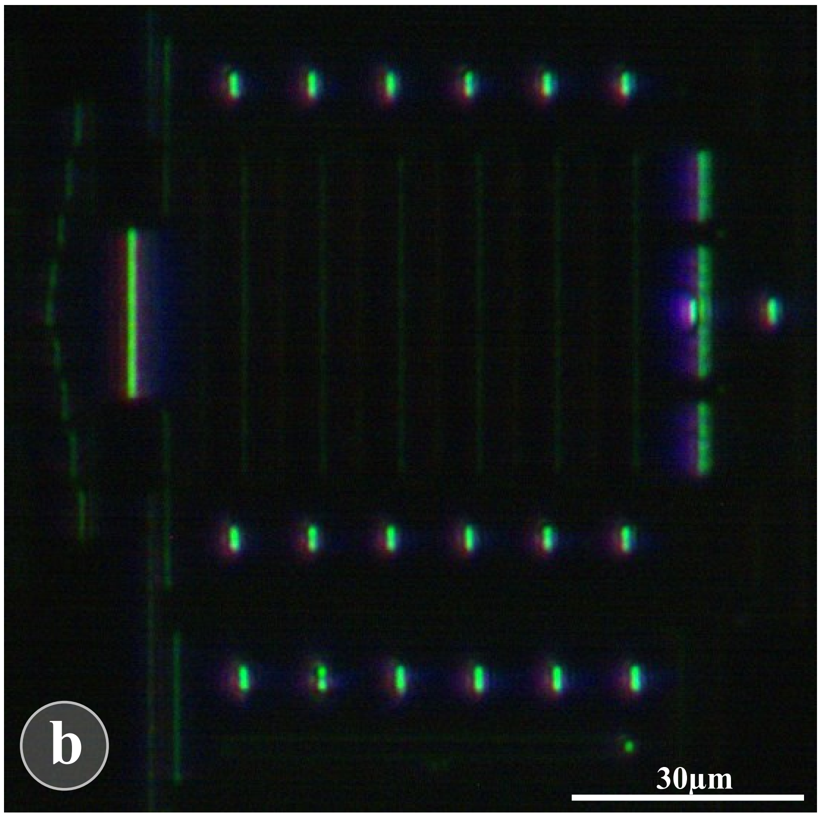


Figure 56: (a) A Zoomed-in view of the cell squeezer under coaxial illumination (b) The same device under a lateral illumination from the left. Note that non-flat features appear bright.

Shiny polysilicon surfaces pose two major problems for RTI implementation: 1) The lighting directions introduced in milli-scale RTI (light sources at different grazing angles all round a subject) cannot be implemented for the MEMS chip. 2) Surfaces do not reflect light diffusely, and as a result, no shadows can be seen in the images.

There is evidence in the literature about the surface roughness of polysilicon thin films [39] implying that the specular reflection from polysilicon surfaces would not be as a perfect mirror, and thus instead of a single specular spike, there would be a cone of specularity (specular lobe) that can be captured partially if the observer is positioned nearly in line with the lobe (see Figure 10).

Since the point of observation is fixed orthogonally above the surface when viewing MEMS devices by an optical microscope, in order to illuminate the MEMS chip and partially capture the specular lobe, here we must restrict the illumination angles close to the surface normal (lower polar angles), as shown in Figure 57. In such a case, the specular lobe can be partially captured, and therefore it is expected to see brighter images in comparison to the illumination directions at a higher polar angle. However, the lighting angle is limited in how close it can approach the surface normal before it is blocked by the objective lens.

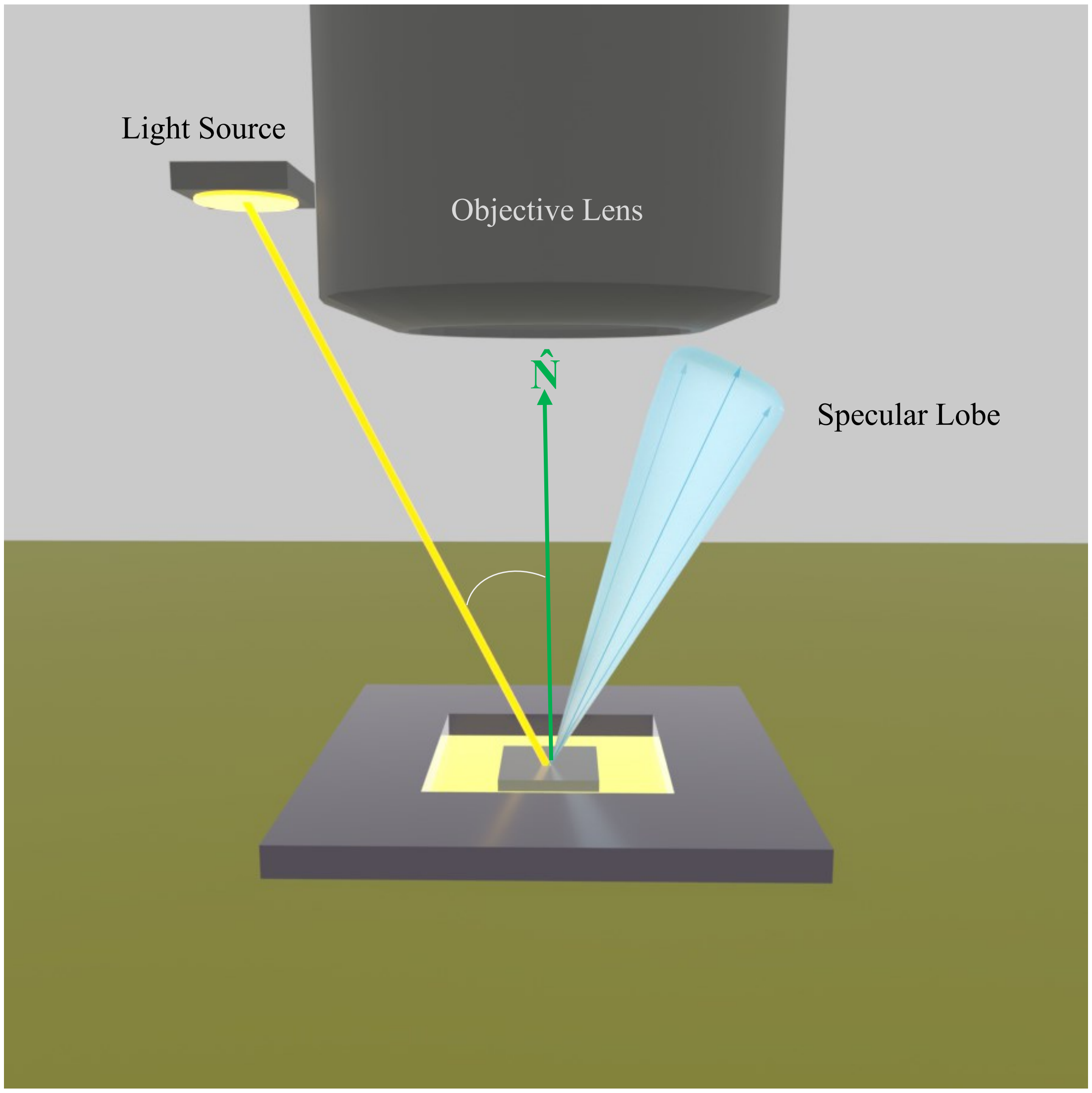


Figure 57: How the reflection of incident lights at smaller polar angles has a greater chance to be captured by camera.

One of the points connected with micro-scale RTI is that in conventional coaxial illumination, the incident light beam passing through the microscope objective lens is collimated and concentrated in a small region of high intensity. However, the external light sources we use in this study are omnidirectional and emit light in many directions rather than at a concentrated angle. This will reduce the image brightness.

Methods for mounting the light sources close enough to the objective lens, and how the image post-processing is modified to reveal three-dimensional information of the MEMS devices will be discussed in the following chapters.

5.5. MEMS Lab Microscope

At Dalhousie MEMS lab, a Bausch & Lomb MicroZoom microscope mounted on a Wentworth probe station is used to observe MEMS devices, as shown in Figure 58. The Microscope is equipped with a coaxial light source that illuminates the MEMS chip that is placed on a mechanical stage that can be adjusted along the X and Y axis using rotary knobs on the station. There are also fine and coarse focus knobs that move down the objectives along the Z-axis to bring the MEMS device into optimum focus. There are three objectives with 2.25×, 8×, and 25× magnifications mounted on a revolving nosepiece. Rotating the nosepiece enables aligning any of the objectives with the viewfield. For achieving higher magnification, this microscope also makes use of a zoom adjustment knob that provides an extended magnification range from 1× to 2×, which leads to higher magnifications of 4.5×, 16×, and 50×. Note that the numerical magnifications above apply to viewing with eyepieces, when viewing with a camera the actual magnification will be different depending on the camera sensor and pixel size.

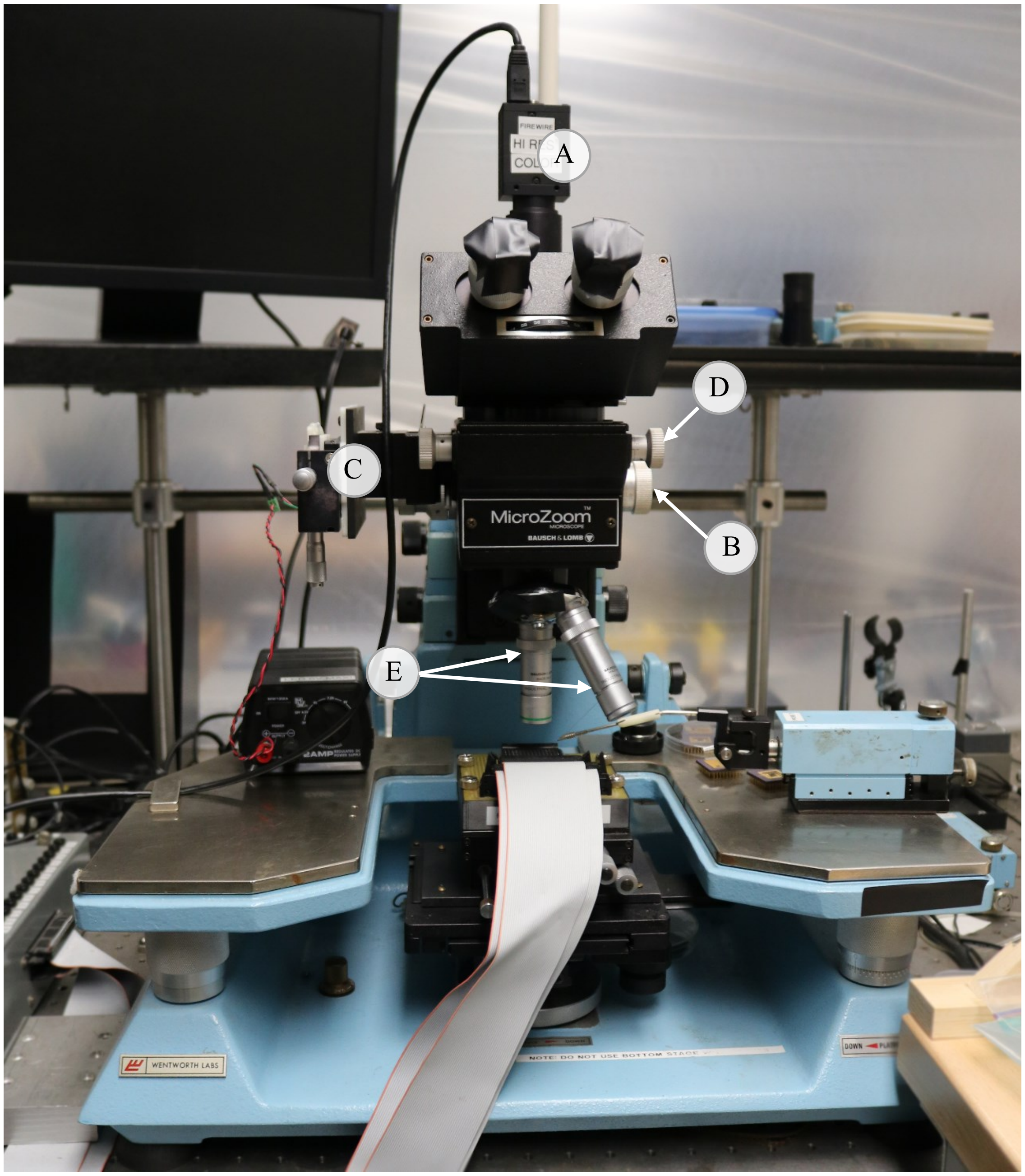


Figure 58: MEMS Lab Microscope: (A) camera, (B) focus knob, (C) light source, (D) zoom knob, and (E) objectives.

5.6. Camera

The micro-scale RTI were imaged using an ultra-compact camera (FLIR Blackfly® S USB 3) mounted on the microscope camera tube using a C mount adaptor as shown in Figure 59. The RGB images acquired by the FLIR camera have a resolution of

1536×1024 pixels and the size of the pixels at 16× and 50× magnification was 0.58 μm and 0.18 μm respectively.

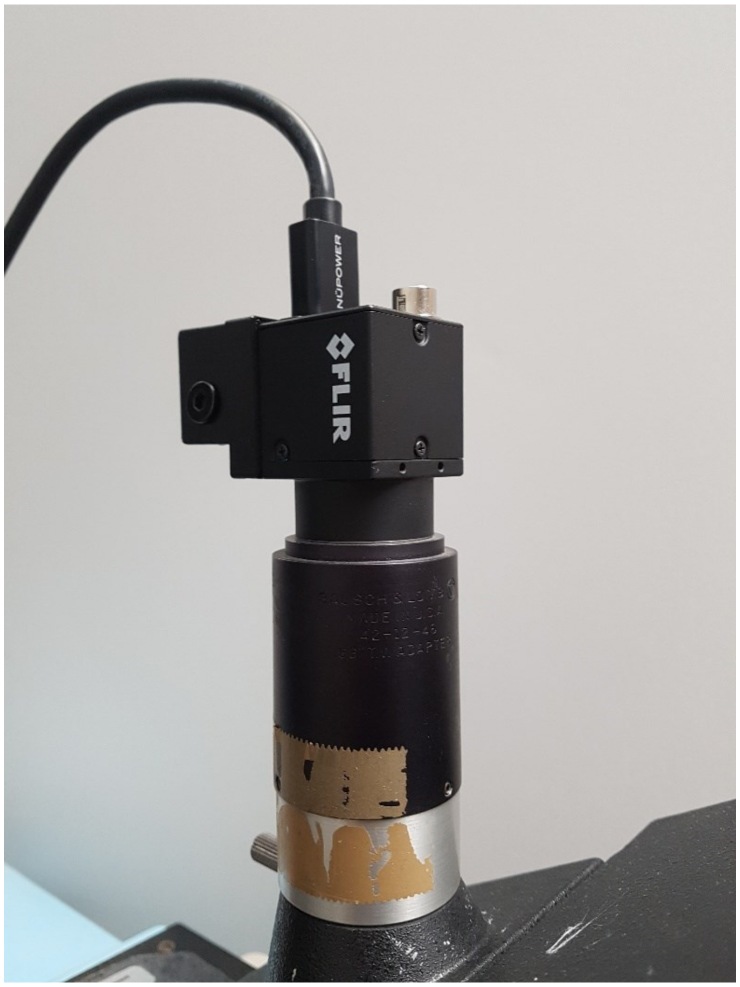


Figure 59: The FLIR camera mounted on the microscope.

MATLAB facilitates the image acquisition process by introducing functions that allow the user to control the camera and the image properties. To make use of these functions, first the “Image Acquisition Toolbox” and “FLIR Spinnaker support by Image Acquisition Toolbox” need to be properly installed. Image acquisition toolbox provides a graphical user interface that enables real-time imaging and can be found in MATLAB through the Home tab → Environment section → Add-Ons option. Having these toolboxes properly installed, through APPS → APPS section → Image Acquisition option, a MATLAB programmatic interface allows the user to identify the detected hardware, select a desired color space, preview a live video, and make use of the MATLAB built-in functions. This interface can be accessed also by entering the "imaqtool" command in command window.

Figure 60 shows the first three lines of the MATLAB capture program that allows the user to specify the FLIR camera and then preview a live video. The "videoinput()" function is used to define a connected camera as a particular object whose properties can subsequently be manipulated. The first argument is the adaptor name assigned to the FLIR Blackfly camera used in these experiments and can be found through the Hardware Browser section in image acquisition window. The second parameter is used to specify the connected device among other devices by a scalar value. The user can also determine the color space of the acquired images or videos using the device's supported color spaces that can be found in the image acquisition window. The second line indicates the video source that toolbox assigns to the created object in order to adjust specific properties of the device. Then the "preview()" function shows a live stream from the selected object.

```
vid = videoinput('mwspinnakerimaq', 1, 'BGR8');
src = getselectedsource(vid);
preview(vid);
```

Figure 60: Part of the MATLAB program to define the FLIR camera and open a live preview window.

The defined object has an ON and OFF running mode that is required to be set before any frame acquisition is accomplished. This can be implemented by "start()" function as shown in Figure 61. Then the "getdata()" function returns the frame captured by the object which can be stored subsequently using "imwrite()" MATLAB built-in function.

```
start(vid)
frame = getdata(vid);
my_image = frame;
```

Figure 61: Part of the MATLAB program to capture a frame.

One of the advantages of using the image acquisition toolbox interface is that manipulating the camera's properties in the image acquisition toolbox interface generates lines of code that can be readily cut and pasted as part of other MATLAB programs which will be discussed in the following chapters.

The aim of this section was to represent a description of the planar structure of MEMS devices and the micro-scale three-dimensional features that are perceived in 2D using optical microscopes. The reflectance properties of PolyMUMPs were also explained to be considered when the micro-scale RTI is applied. Here we restricted ourselves to the conventional optical microscopes and 2D captured images. The experimental setup required to observe MEMS devices are the Dalhousie MEMS lab microscope and a digital camera. In the following sections, the micro-scale RTI will be explained, and MEMS devices and other micro-scale subjects illuminated under lateral lighting are shown. What remains to be explored is the implementation of micro-scale RTI technique to MEMS failure detection.

6. Micro-Scale RTI

The objective of this chapter is to illuminate the MEMS surface so that a sequence of images containing information on three-dimensional characteristics of MEMS devices can be captured. Using the NeoPixel LEDs and Arduino board seen earlier in chapter 3, plus the FLIR camera (all interfacing with MATLAB programs), the micro-scale RTI can be set up. Two MATLAB programs: RTI-Capture and RTI-Viewer, are explained. RTI-Capture is responsible for communicating with the LED lights through the Arduino board and triggering the camera once an LED is lit. The micro-scale version of the RTI-Viewer program is discussed later in this chapter and deals with how the captured images should be displayed and post-processed to reveal the three-dimensional visual cues.

6.1. Light Frame: Ring

The proposed milli-scale RTI defined in chapter 4 used a dome lighting array of 50 LEDs that illuminated the subject at different polar angles and azimuth angles. As shown in chapter 4, milli-scale subjects scattered the incident beam in every direction. However, using a similar dome lighting array to illuminate the MEMS surface results in dark images with no visuals. MEMS shiny surface reflects light specularly and the outgoing light beams would never enter the objective lens unless the light source is aligned close enough to the objective lens. This limits the usable LED sources to the only the very smallest Neopixel form factors. The 12-LED NeoPixel ring is used as the light source in micro-scale RTI due to its compact size: it will fit around the objective lens without blocking its front lens, as shown in Figure 62. To fasten the NeoPixel ring tightly around the objective lens, a slightly undersized cylindrical 3D printed frame is used, which has a slit to allow a slight opening of the cylinder. The frame is positioned so that the LEDs are in line with the front lens of the objective.

For the 25× objective at a working distance of 20 mm, the 12 ring NeoPixel LEDs have the same polar angle of ~45° and illuminate the MEMS chip at 12 different azimuth angles from 0° → 360° in 30° steps.

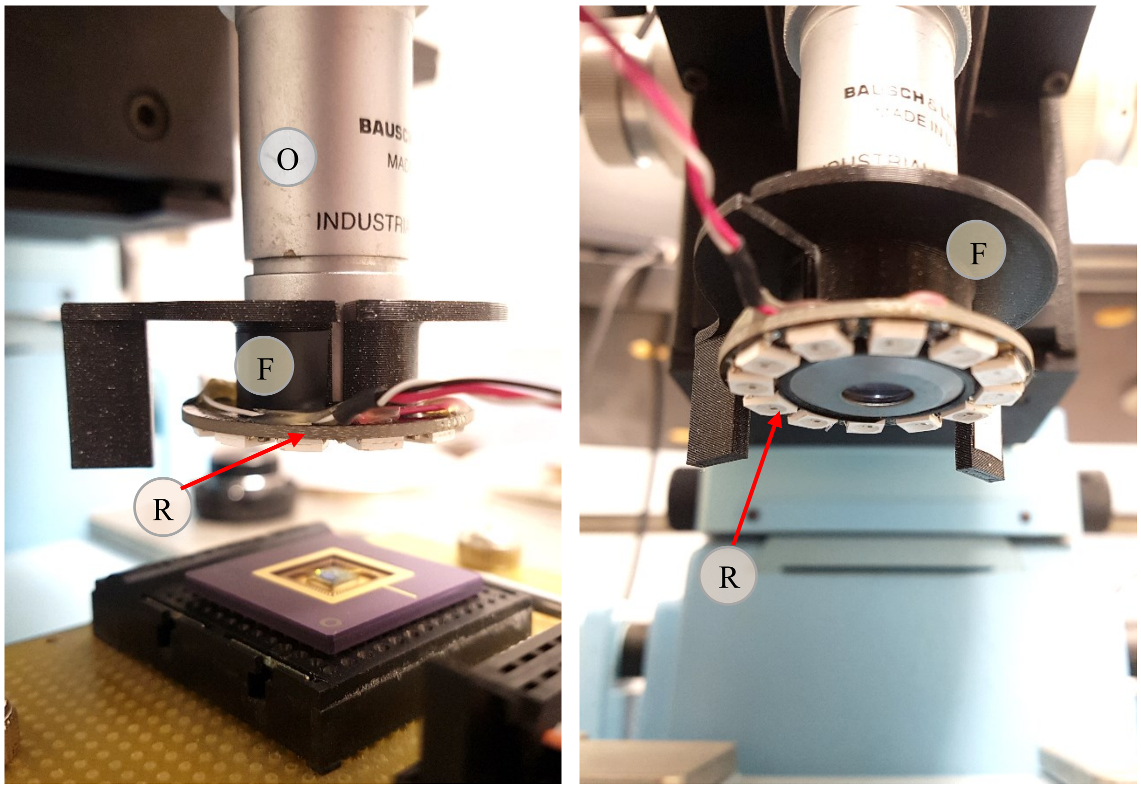


Figure 62: Micro-scale RTI lighting array: (R) NeoPixel Ring light source, (F) 3D printed cylindrical frame, (O) objective lens.

6.2. Micro-RTI Capture

A series of 12 images lit individually by 12 LEDs of the Ring NeoPixel are taken, where the light source and camera operate in coordination. To this end, a MATLAB program was written which in the first step initializes the Arduino board and the ring NeoPixel, as well as the FLIR camera (see Figure 19 in chapter 3). The Coordination between turning on the LEDs and capturing images using the camera results in the sequence of RTI images. For the sake of brevity, these 12 laterally illuminated images are referred to as “single images” throughout this chapter. For further reading regarding how these images are captured, see the Appendix A-1.

Prior to capturing single images, a brightfield image and a darkfield image are captured. A brightfield is the conventional MEMS imaging when the coaxial light source lights the MEMS surface vertically. To capture a brightfield image, the user is asked to turn on the microscope-integrated light source, and then the image is captured and stored in the defined directory. As stated in chapter 2, a darkfield image is generated by emitting a hollow cone of light using a ring-like illumination source and it aims to highlight the sidewalls while the flat surfaces remain dark. To this end, turning on the 12 LEDs of the ring NeoPixel can simulate the dark-field illumination. Thus, the imaging order is:

1. User turns on the microscope light source,
2. MATLAB takes one brightfield image,
3. User turns off microscope lamp,
4. MATLAB takes one darkfield image with all 12 LEDs "ON",
5. MATLAB takes 12 individual LED images.

By executing the micro-RTI Capture program, 14 images are stored in this order: a brightfield image, a darkfield image, and 12 single images. Figure 63 illustrates (a) the brightfield image of the cell squeezer device, (b) the darkfield image of the device with 12 LEDs lit, and (c) one of the single LED images. The darkfield image focuses the attention on sidewall facets and inclined features while flat surfaces remain dark. The overall illumination of the 12 LEDs darkfield image is higher than the single LED images.

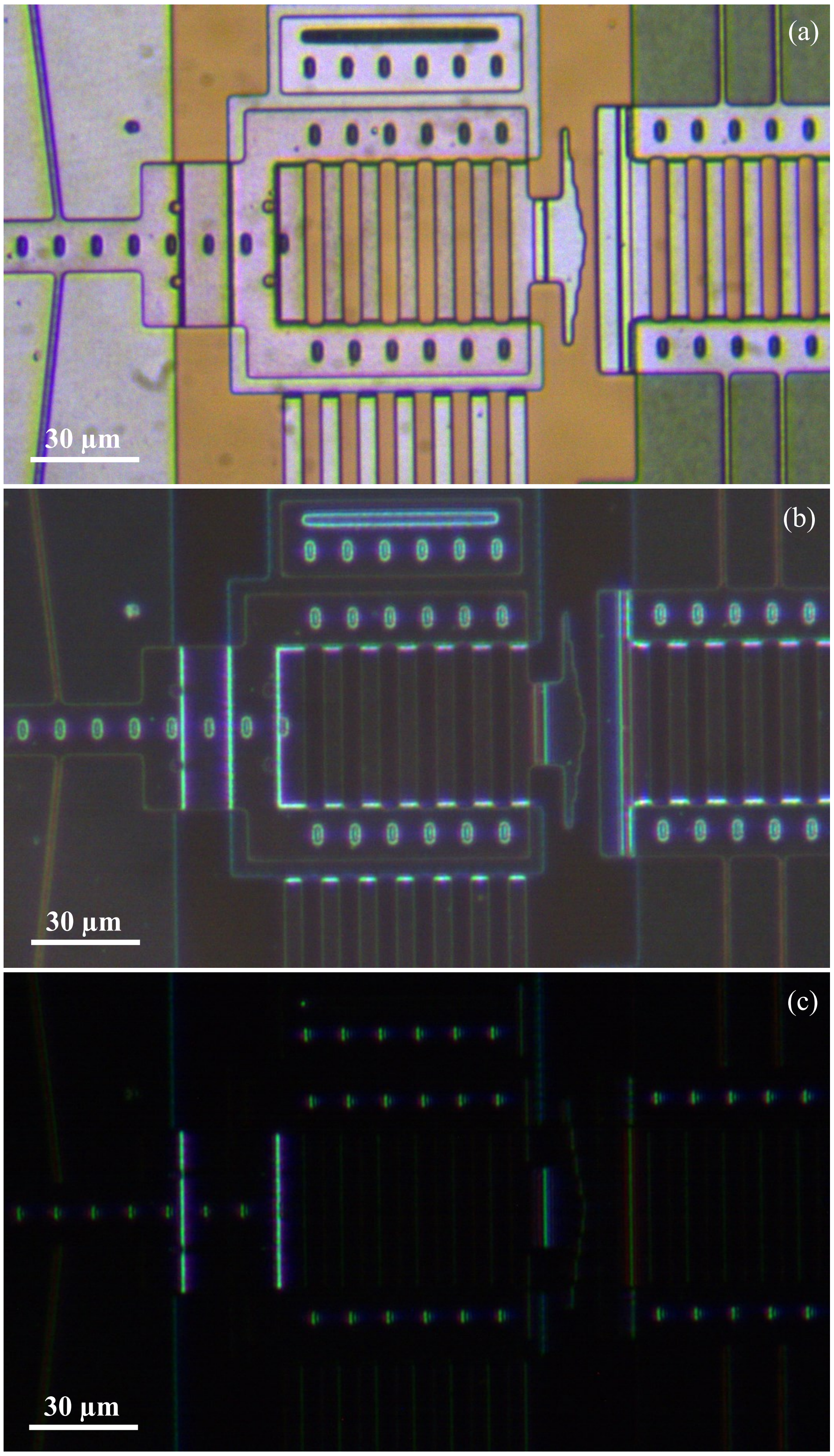


Figure 63: The cell squeezer device under (a) brightfield illumination, (b) darkfield illumination, and (c) one RTI lateral illumination (from the right side).

6.3. Micro-RTI Viewer

This section describes how the captured images are browsed interactively by the user command and how the three-dimensional aspects of the MEMS devices can be highlighted. As discussed in milli-scale RTI, the proposed diagnostic procedure in this chapter is primarily qualitative rather than quantitative.

6.3.1. Browsing Images

The primary focus of the Viewer program is to interactively display the 14 captured images of each scene and allow the user to browse the images and extract the information content of the scene by comparing different visual representations of features.

To this end, similar to the approach in the milli-scale Viewer program (see Figure 23) the user first specifies the directory of the captured images, and the 14 read images are stored in an array named "file_names". The next step is to stack the captured images in a four-dimensional array ($m \times n \times t \times 3$) in which each captured RGB image ($m \times n \times 3$) is assigned the index t (*theta*).

The brightfield and darkfield images are stored in theta values of 1 and 2 respectively, and the individual images are stored in *theta* values of 3 to 14.

As with the milli-scale RTI, user image navigation can be implemented by defining an infinite *while* loop that executes constantly and displays an image based on the user key command unless the "*q*" key on the keyboard is presses to quit the loop. By pressing an arrow key, the *theta* value varies, and the displayed image updates accordingly.

6.3.2. Image interpretation

To achieve a visual recognition of the cell squeezer as an example of MEMS devices, first, the MUMPs layout is shown to be used as a reference image that reveals the layer type and features that may be poorly recognizable in a brightfield image. Figure 64 shows the top-view schematic of the cell squeezer shown in Figure 63 that is composed of POLY0 (magenta), POLY1 (green), POLY2 (red), dimples, etch holes, anchors, and POLY1-POLY2_VIAs.

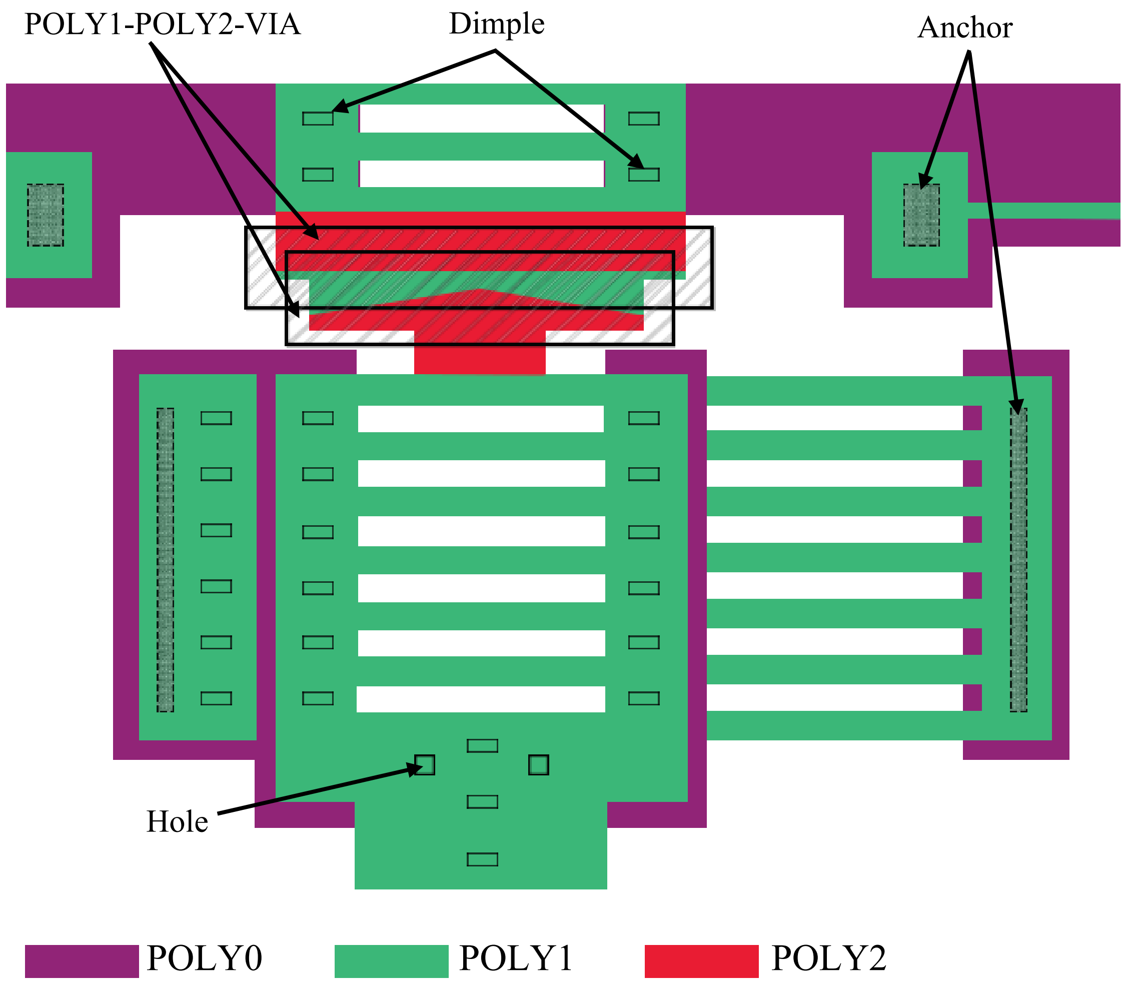


Figure 64: The MUMPs layers of the cell squeezer device.

To recover the three-dimensional information from the set of images, a magnified portion of Figure 64 is examined. The magnified insets in Figure 65 show (a) brightfield image, (b) the MUMPs layers, (c) the darkfield image of a selected region of the cell squeezer device, and (d) one of the single images.

The chosen region exhibits a complex micromachined structure that at first sight consists of two to three structural layers (based on the colors), dimples, and multiple stepped structures.

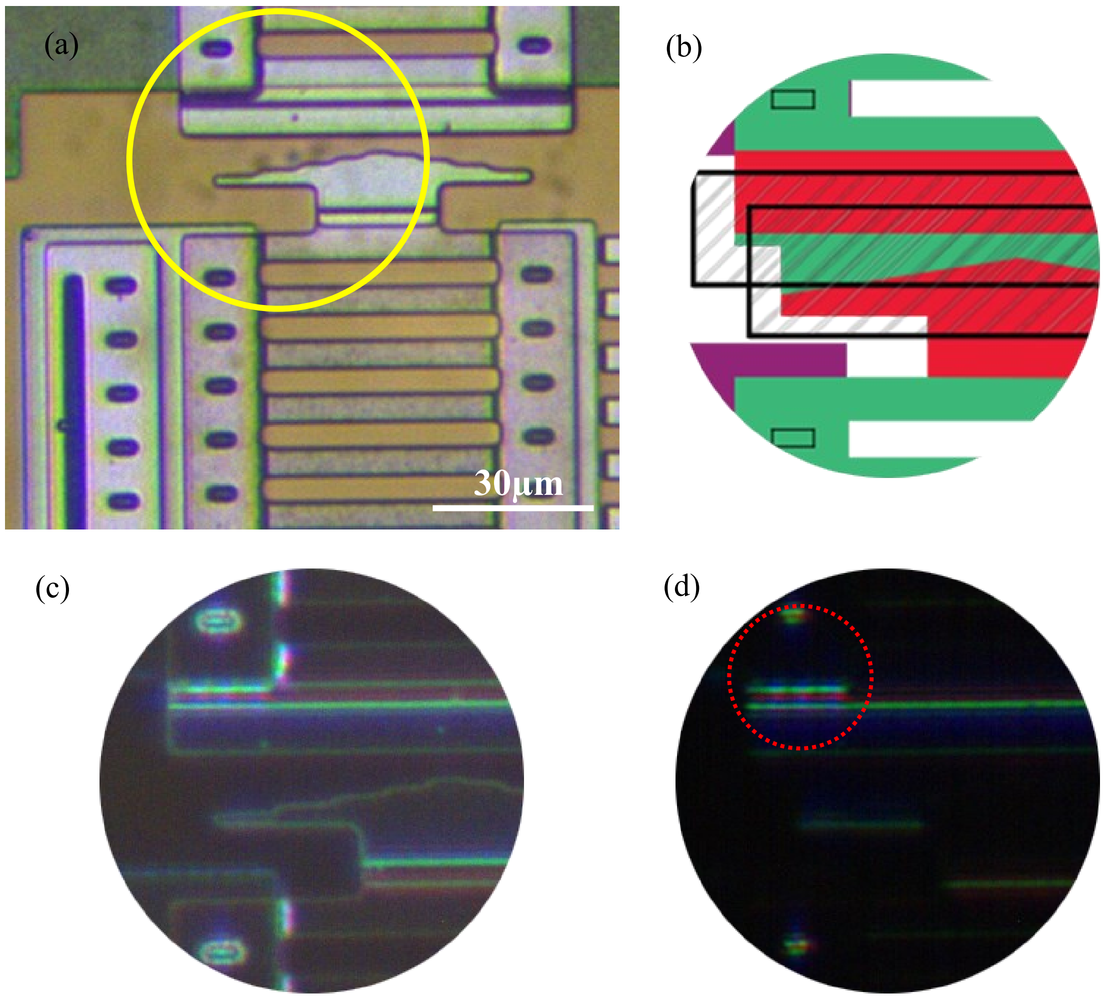


Figure 65: Magnified insets of the cell-squeezer image in (a) brightfield, (b) MUMPs layout, (c) darkfield image, and (d) single image illuminated from the south.

With no prior experience in interpreting the topography of MEMS devices, it would be challenging to draw conclusions about the MEMS surface based on the conflicting brightfield visual appearance of the features. Figure 65 (a) has no visual cues to reveal the step edges that are expected to exist. The MUMPs layout in Figure 65 (b) illustrates the constituent layers of the cell squeezer: POLY1 and POLY2, and thus reveals the fact that the device is made of two different layers superimposed on each other. Figure 65 (c) is the darkfield image that emphasizes sidewalls in any direction that 12 LEDs emit light. As a result, the sloped features stand out, while the lower flat surfaces seem so dark. When the device is illuminated solely from the south as in Figure 65 (d) the sidewall edges that are faced relatively towards the light rays coming from the south are highlighted. This indicates that the topography takes a step up as we climb onto the POLY1, then a second step up as we then climb onto the POLY2.

6.3.3. Composite Images

The post-processing stage aims to merge the information of two or more images in order to enhance the inferences on a subject surface.

The proposed post-processing stage contains the following images: (a) a RGB compound image that is formed by combining brightfield images with single images, (b) the modified HSV image. How these composite images are implemented and viewed is discussed below.

The RGB Compound Image:

The main idea involved in composite images is to partially add an image to another image. The variable that determines the fraction of each image added to the other image is known as "*weight*", as discussed in chapter 4.

Figure 66 shows (a) the brightfield image of the cell squeezer device, (b) the first single image, (c) the 50/50 weighted image of the single image and the brightfield image. The combination of these images allows us to observe the conventional image while a three-dimensional aspect is added to improve our visual perception of a MEMS device.

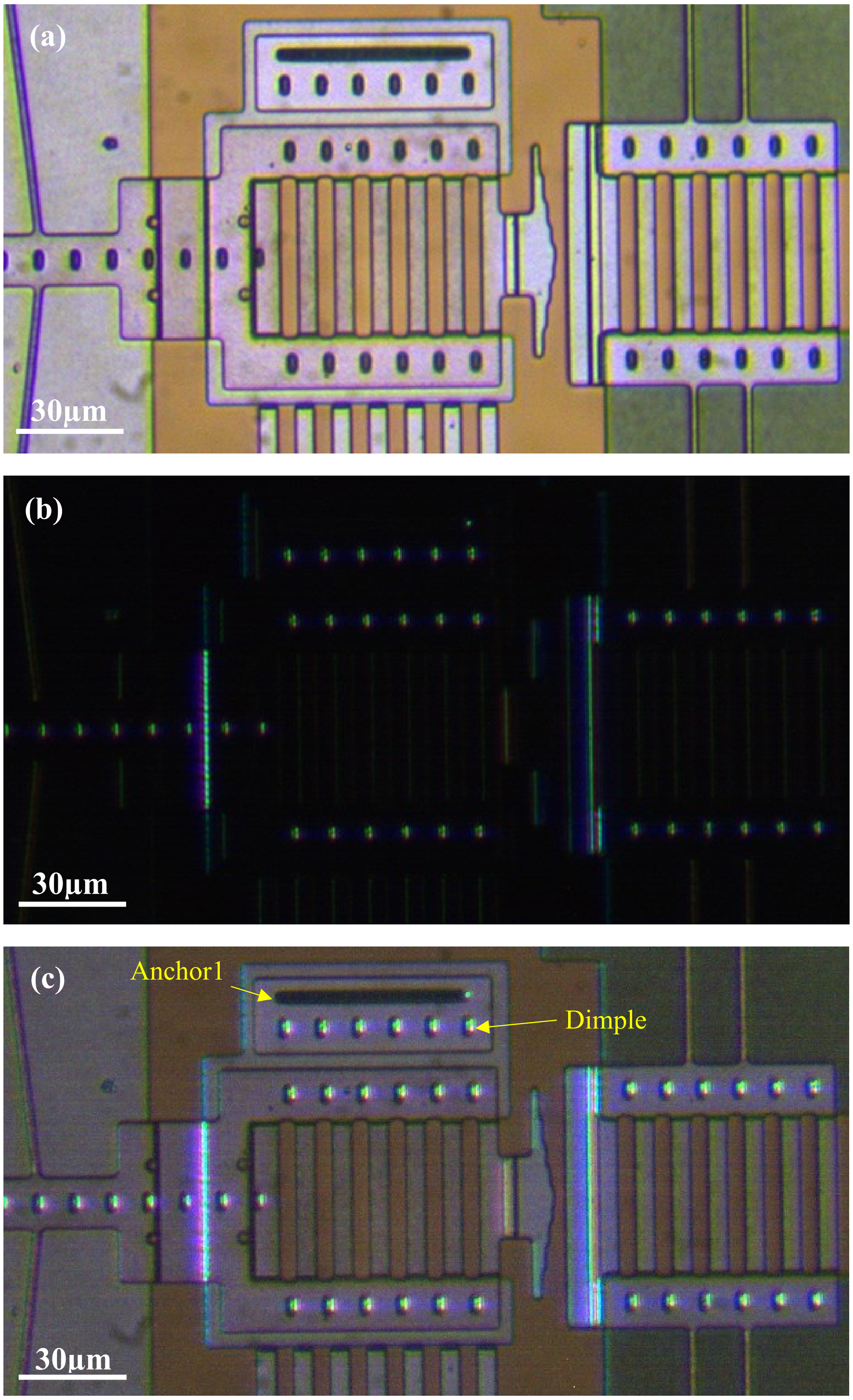


Figure 66: (a) The brightfield image of the cell squeezer device, (b) one of the RTI single images of the same device, and (c) 50/50 contribution of the single image and the brightfield image.

The RGB compound image in Figure 66 (c) can be used to discern the oval pit-shaped dimples with their inner right sidewalls highlighted when illuminated from the left side. Anchor1, the dark elongated ellipse in the top center of the image, can also be detected by a dot-shape highlight in the right corner. When side lighting highlights the inner edge of a shape, it implies that the structure is a depression and does not protrude outward. Conversely, highlights on the outer edge of a structure indicate that the edge is an elevated structure relative to the surrounding surfaces.

The HSV Image:

The HSV image is used to display which direction of illumination produces the brightest state. Unlike diffuse surfaces in milli-scale RTI, specular surfaces such as MEMS devices are highly view-point dependent, and thus a dome-based lighting frame is not compatible with a specular surface. Therefore, the HSV algorithm used in chapter 4 needs to be modified.

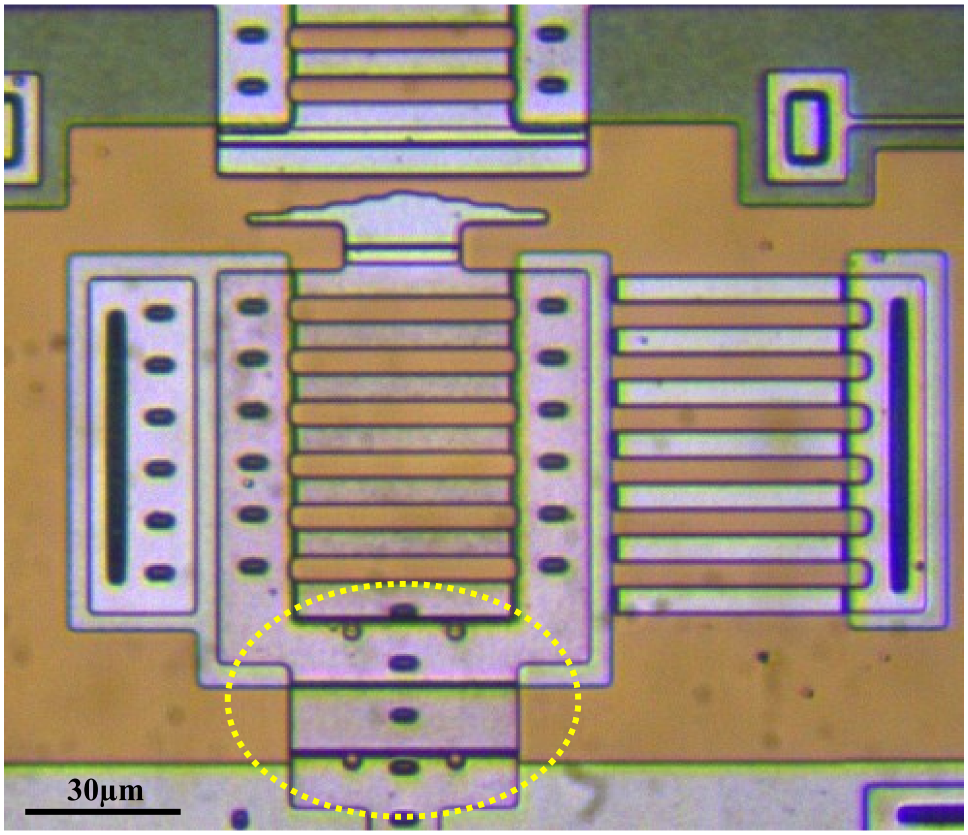
As mentioned before, HSV image contains three components: hue (H), saturation (S) and value or brightness (V). Hue is assigned to azimuth angle ($0 \rightarrow 360°$) of the LEDs in both milli-scale and micro-scale RTI-Viewer. Thus, the color of the pixels represents the spatial direction of the emitting light with respect to a continuous 360° spectrum around a given point on the surface. In the milli-scale RTI, LEDs were mounted not only at different azimuth angles but at different polar angles, the saturation channel was assigned to varying polar angles of LEDs. However, in micro-scale RTI, all the 12 LEDs of the ring NeoPixel have the same polar angle. Therefore, the saturation channel is employed to show the maximum intensity of each pixel among all the captured images. It implies that the higher the intensity value of a pixel, the more saturated the pixel is when the HSV image is displayed. The third channel of the HSV color map is assigned to the brightness which in micro-scale RTI-Viewer is defined by the pixel intensity of the grayscale version of the brightfield image.

To obtain the HSV image, first the grayscale versions of single images are stacked in a three-dimensional array ($m \times n \times t$). The first and second elements of the array correspond to the image dimensions, and the third element *t* is the lighting direction (*theta*). Then, the maximum value that each pixel carries and the lighting direction that corresponds to the maximum value are detected using the “max()” MATLAB built-in function, and are assigned to the saturation and hue elements of the HSV image, respectively (for more info regarding HSV image refer to chapter 4).

The flat regions are still not detectible in color in the HSV Normal image. Conversely, features with inclined sidewalls, such as dimples, anchors and step edges exhibit the sidewalls’ orientation through colors that can be perceived using the HSV color wheel.

Figure 67 (top) illustrates the conventional brightfield image along with (bottom) the HSV image obtained for the same cell squeezer device. The color wheel in the upper left corner of the HSV image represents the direction of the incident light relative to the MEMS surface.

For instance, red pixels had the brightest state when they have been illuminated from the east, and green regions are highly illuminated from the north. The selected region in the lower part of Figure 67 (top) shows three dark lines in a row, made of POLY1 based on the layout illustrated in Figure 64. However, neither the layout nor the brightfield image provides information on the surface topography of the given region. However, the HSV image shown in Figure 67 (bottom) provides visual cues that reveal the three-dimensional surface structure. Starting from the bottom, the alignment of the green line, then purple line, then green line illustrates three consecutive step edges that are formed by POLY1 that changes orientation from: a flat surface on POLY0 dropping to a lower flat surface on Nitride (green), then climbing up again to a higher flat surface on POLY0 (purple), and finally dropping to a lower surface on Nitride again (green). Looking at the same structure in Figure 67 (top), the complexity of the structure cannot easily be discerned without a pre-knowledge about the layout.



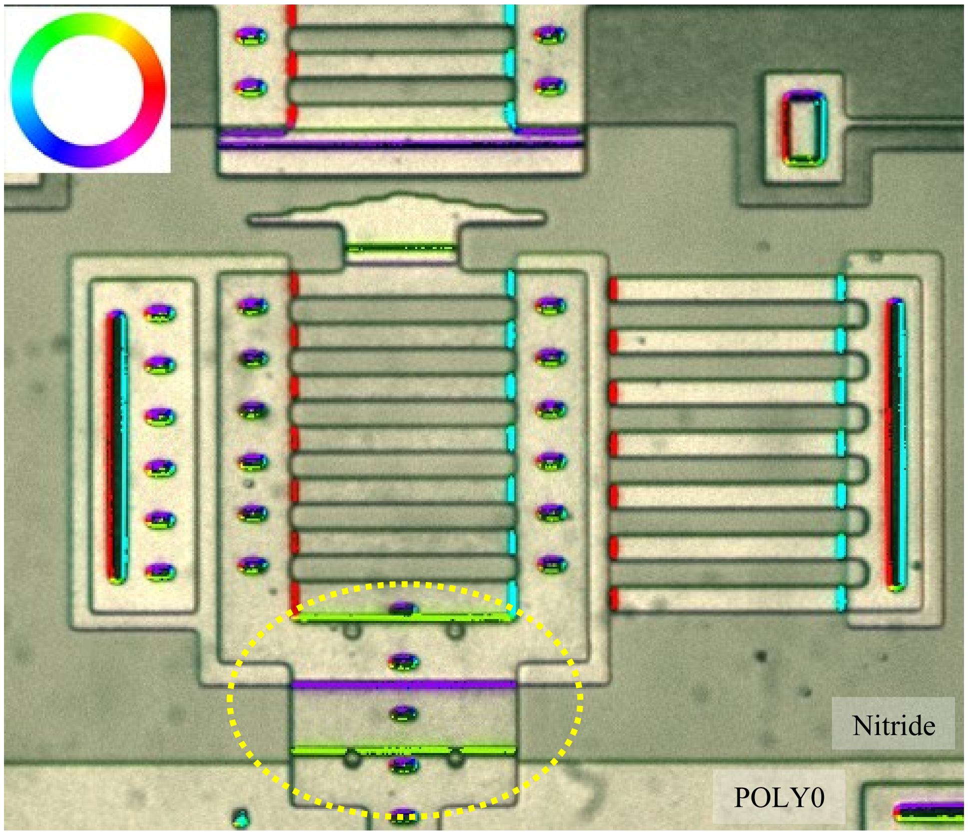


Figure 67:Top: The brightfield image of the cell squeezer. Bottom: the HSV image.

6.4. Light Frame: Cylinder

An integral part of RTI imaging is to illuminate a subject both from varying azimuth angles θ and polar angles ϕ. The NeoPixel ring fastened around the objective lights up the MEMS surface only from different azimuth angles θ. To address this problem, an additional lighting arrangement is required to complement the ring frame without blocking either the ring NeoPixel or the objective front lens. A 43 mm × 10 mm cylindrical frame was 3D printed to accommodate an Adafruit flexible lighting strip containing 12 LEDs (Figure 68 (left)). Then, using two binder clips, the new cylindrical frame was suspended below the NeoPixel ring frame, as shown in Figure 68 (right). For the sake of brevity, the original ring NeoPixel and its lighting frame is referred to as ring lighting array, and the added lighting array is referred to as cylinder lighting array.

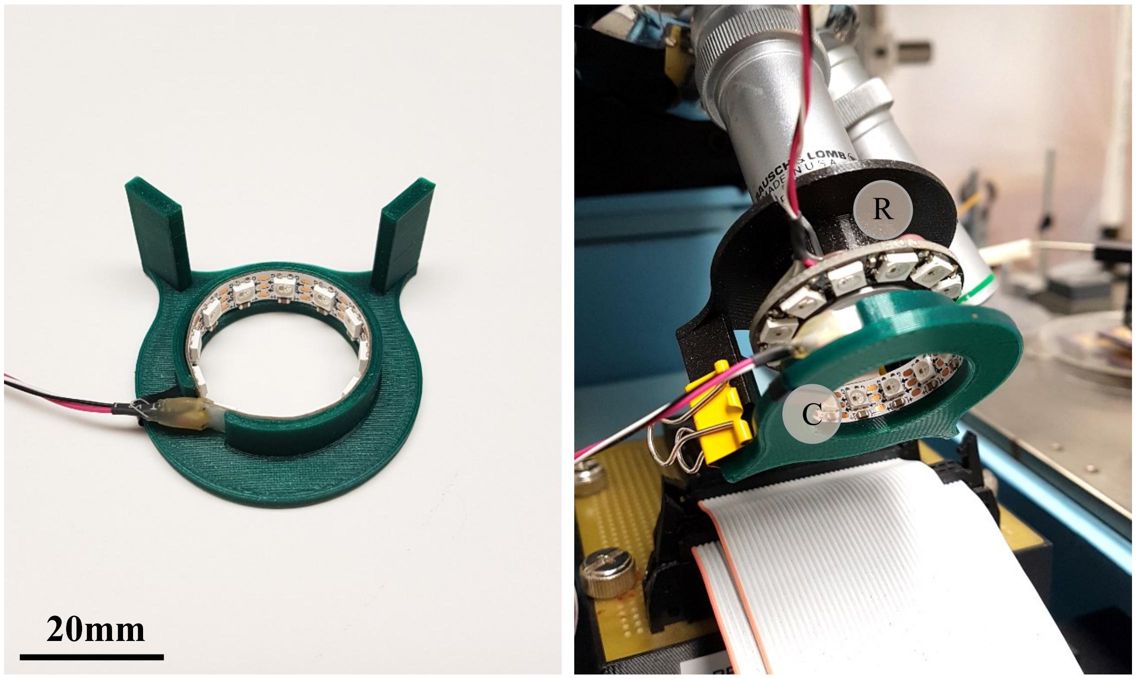


Figure 68: (Left) The flexible lighting strip mounted on a cylindrical frame, (Right) micro-RTI 3D lighting arrangement: (R) ring lighting array (C) cylinder lighting array with flexible lighting strip.

While the ring lighting array is constrained to a ~45° polar angle, the cylinder lighting array can provide incident polar angles between ~48° and ~60°, depending on its distance from MEMS surface. Figure 69 (left) illustrates a cutaway view of how lighting arrays are aligned with the microscope objectives and the MEMS surface, and (right) the implemented micro-scale RTI lighting arrangement. The ring NeoPixel is aligned with the front lens of the objective, and the cylinder is positioned as close to the MEMS surface as possible and emits incident light rays at a ~60° polar angle.

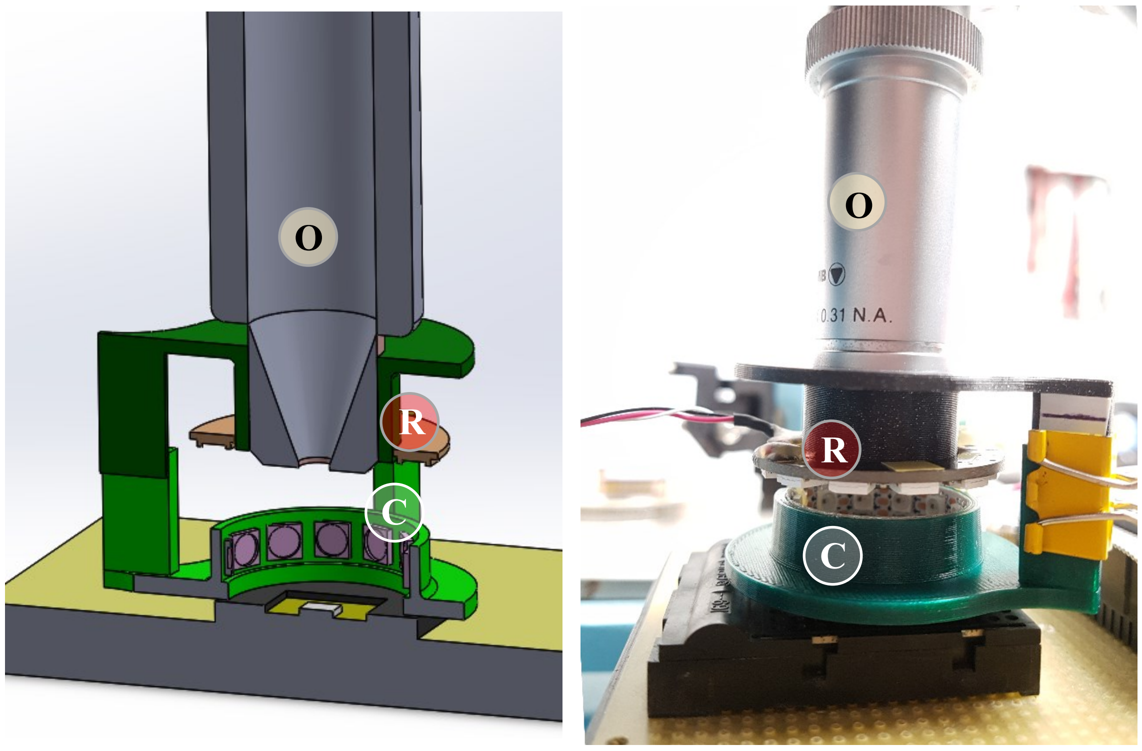
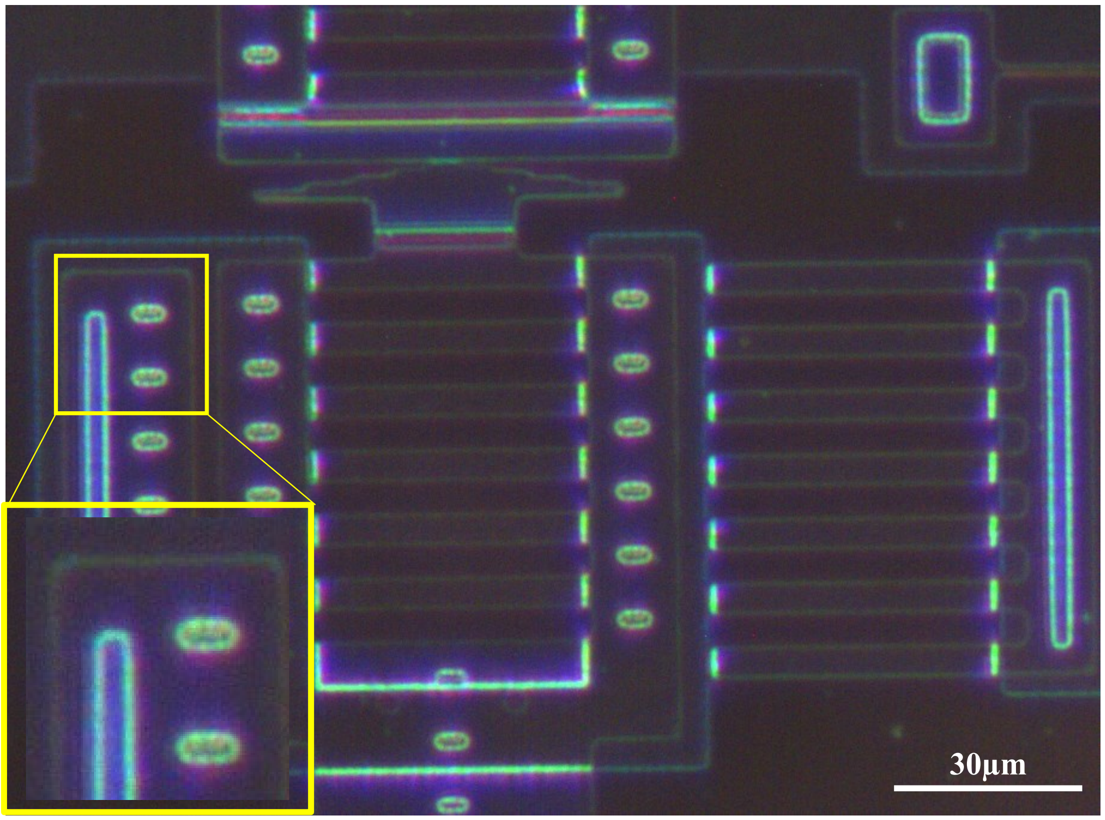


Figure 69: (Left) The cutaway illustration, and (right) the image of the implemented micro-RTI lighting arrangement: (O) the objective, (R) ring lighting array, and (C) cylinder lighting array.

According to the law of reflection and reflectance properties of specular surfaces, light rays with incident angles of ~60° expose surfaces at an angle of ~30° to view. Figure 70 shows the darkfield images captured using (top) the ring lighting array and (bottom) the cylinder lighting array. The process of capturing images using the cylinder lighting array is the same as that of the ring lighting array.



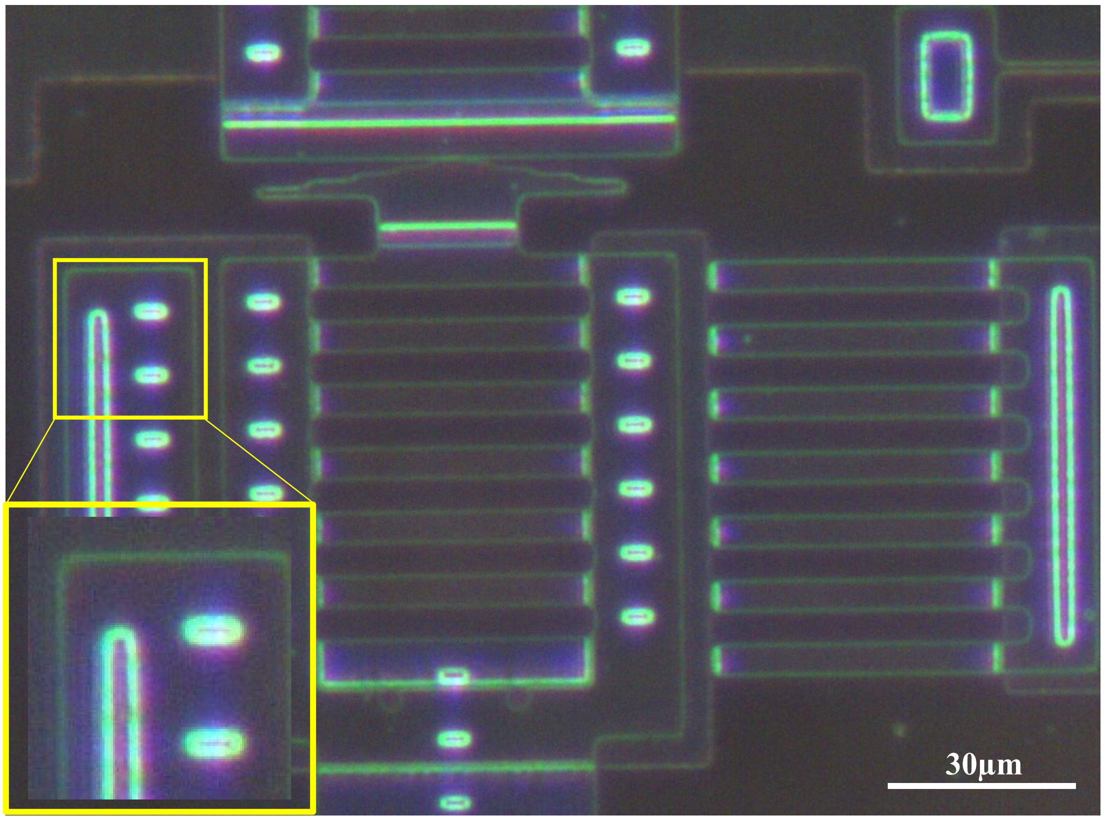


Figure 70: Darkfield image captured using the (top) ring NeoPixel lighting array, and (bottom) cylinder lighting array. Zoomed insets at lower left of each image.

The cylinder lighting array yields images with a higher overall brightness. Furthermore, as the magnified insets emphasizes, the cylinder lighting array better illuminates inner sidewalls of both dimples and anchors compared to the ring lighting array that illuminated the outer boundary of the feature. It implies that the sidewalls of the dimples and anchors are steeper within the feature than the edges. Therefore, the combined use of ring NeoPixel and cylinder lighting array allows us to explore the approximate slope of the sidewalls and the topography of the features.

The distinction between surfaces with a higher slope (~30°) and surfaces with a lower slope (~22.5°) can also be visually detectable by comparing the HSV images in two lighting modes: (top) the ring NeoPixel, (bottom) the cylinder NeoPixel, as shown in Figure 71. With respect to the HSV color wheel on the bottom left corner of the images and the magnified insets, the dimple depression can be discerned. Further, the fact that the ring NeoPixel illuminates the contour edges of the dimple, while the cylinder NeoPixel brings out the steeper interior sidewalls can be emphasized by the colored elliptical shape (Figure 71 (top)) that moves towards the inner sidewalls as the light source shifts to the cylinder (Figure 71 (bottom)). The same inferences can be made by observing the anchors and VIAs that their sidewalls exhibit markedly more intensity in the HSV image of the cylinder lighting mode.

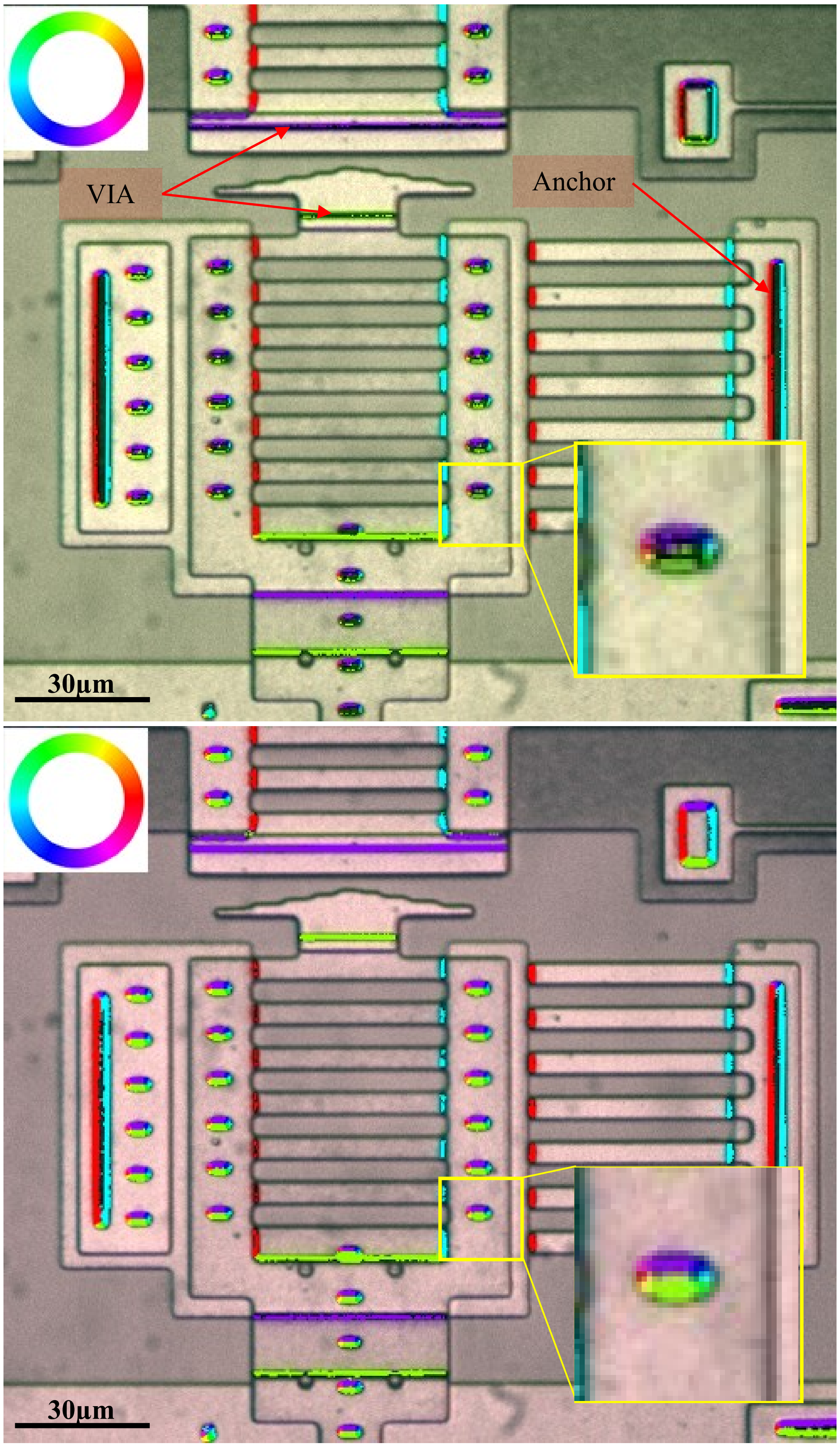


Figure 71: The HSV image of the cell squeezer: (top) under the illumination of the ring, (bottom) under the illumination of the cylinder. Zoomed insets at lower right of each image.

7. Case Studies

This chapter explores MEMS devices whose morphological features and mechanical performance are ambiguous in conventional brightfield images and where the micro-RTI technique can be employed to investigate the visual characteristics of the scene. Visual features of interest include the three-dimensional geometry and out-of-plane motion.

7.1. Micro-Bearing

Figure 72 shows the brightfield image of a MEMS bearing along with a magnified view of the central disk which is approximately 160 µm in diameter. For the bearing to be micromachined, a thin under layer of POLY0 is patterned to form a circular shape. Then OXIDE1 is etched subsequently to create the dimple masks. Then, a slightly smaller diameter circular POLY1 is deposited on OXIDE1 and a 25.5 µm in diameter central hole is etched on the POLY1 surface all the way through the layers underneath to form a donut shaped structure. An array of 6 µm in diameter square/circular holes and 5 µm in diameter square dimples are distributed on the surface of POLY1. The holes have been etched all the way through the structure, while the dimples are 0.75 µm thick. Next, a thin layer of OXIDE2 is patterned at the center of the hole to allow the next structural layer (POLY2) be anchored to the substrate. Lastly, the central hole is capped by POLY2 that constrains the bearing but also allows it to spin. As figure illustrates, a sequence of 8 radial spokes radiates from the POLY1 donut, as do multiple POLY0 radial spokes surrounding the donut structure.

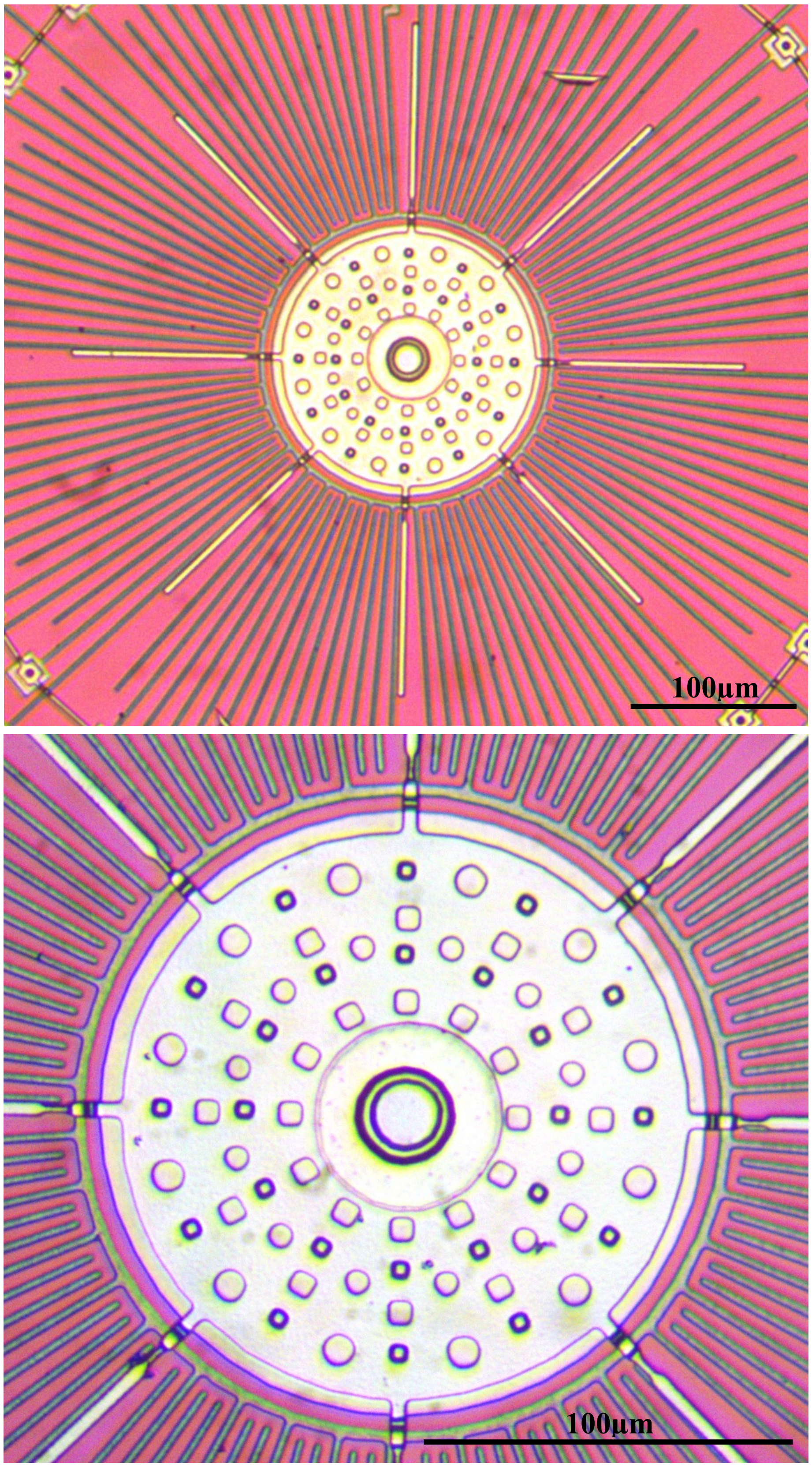


Figure 72: Top: Brightfield image of a 160 µm diameter MEMS bearing. Bottom: Zoom-in view of the perforated POLY1 disk at the center.

Figure 73 (top) illustrates the MUMPs layout of the MEMS bearing consisting of POLY0 (magenta), POLY1 (green), and POLY2 (red), and Figure 73 (bottom) shows a cutaway view of the device when the bearing is cut along the dotted line (yellow) on the top view layout after removing the sacrificial layers.

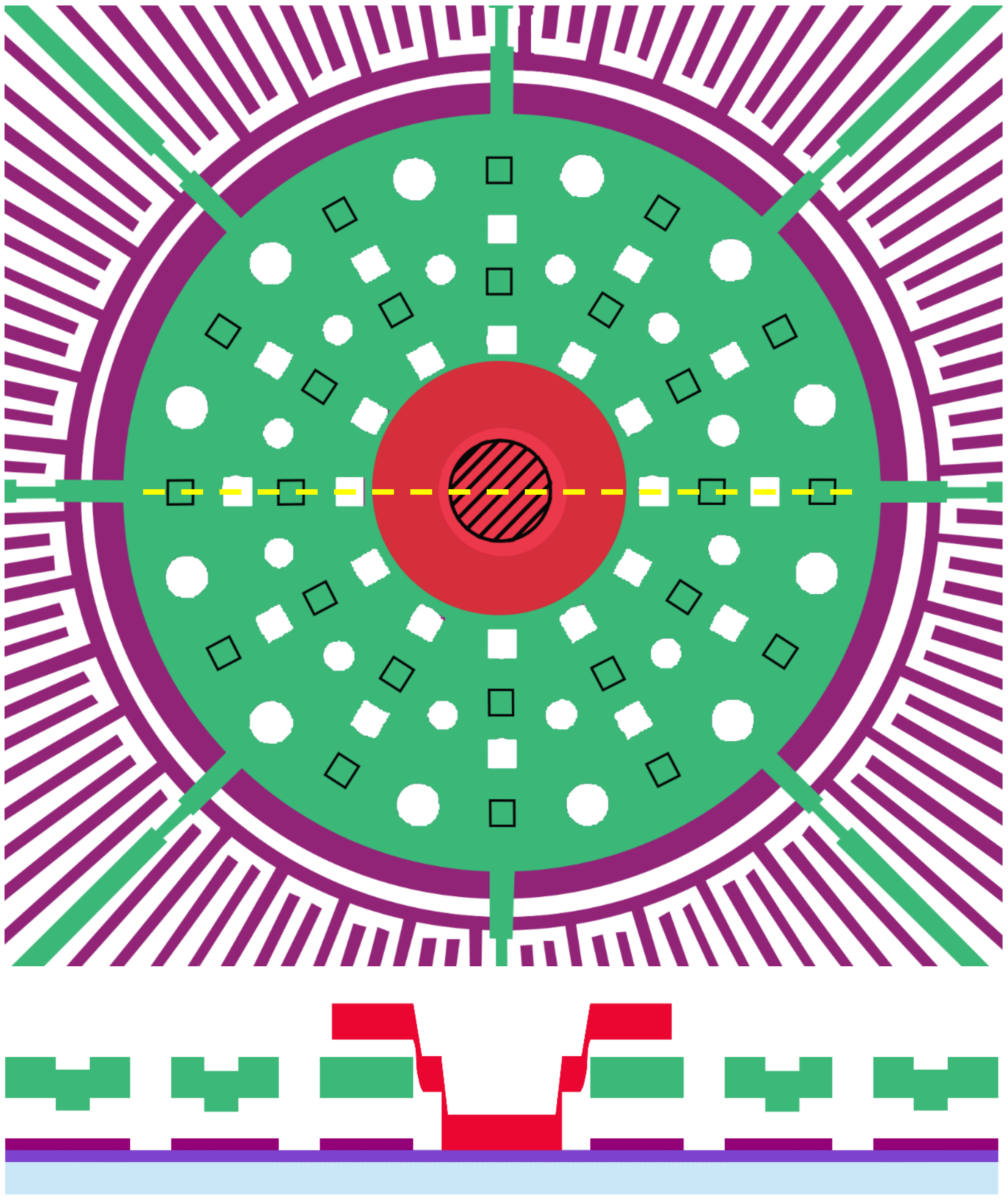
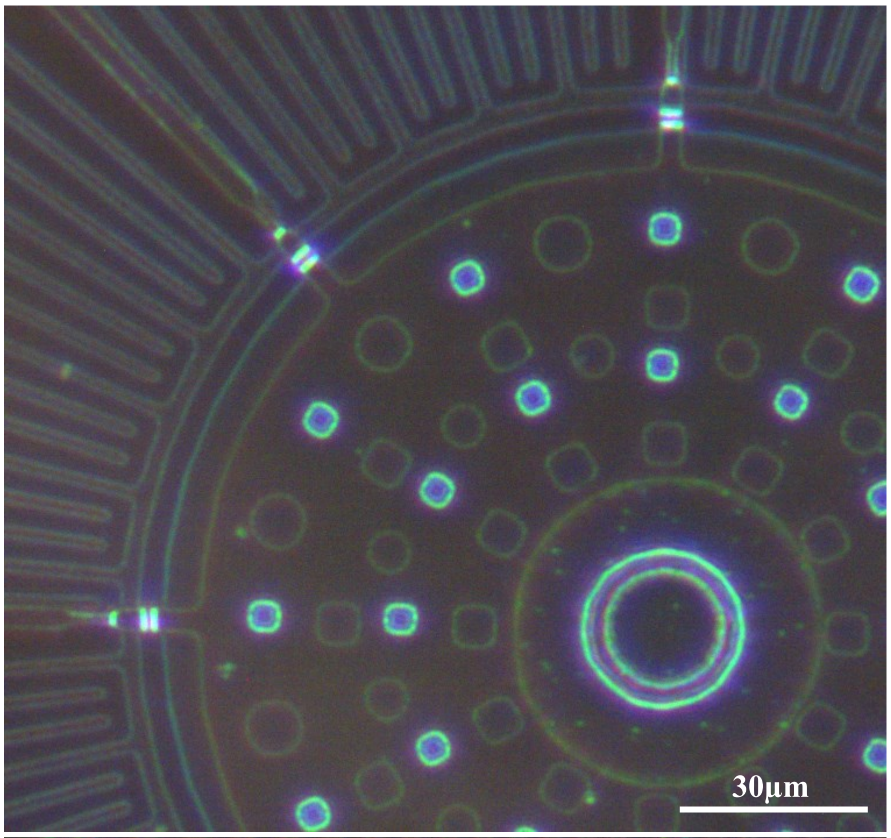


Figure 73: Top: The MUMPs layout of the MEMS bearing. Bottom: The cutaway view.

While the MEMS bearing surface appears completely flat in the brightfield image, the MUMPs layout illustrates how the device is made of multiple layers superimposed on each other, and the cutaway diagram reveals the intricate layered structure of the device that lies at different elevations. To study the geometrical complexities of the device, RTI captures a sequence of laterally illuminated images plus the darkfield image using the two lighting frames: ring and cylinder. As shown in the Figure 74, the darkfield images of the bearing reveals the surface anomalies by highlighting the inclined sidewalls. The dimples have the brightest pixels of the image due to their sloping sidewalls that reflect incident rays towards the objective lens. Moreover, the two bright rings at the center of the device (hub) correspond to the POLY2 layer that slides inwards and downwards on the surface of OXIDE2 to be anchored to the substrate. As expected, the vertical sidewalls such as holes or patterned structural layers cannot be easily detected.

The main difference between the two darkfield images is the intensity of the bright regions which is noticeably more when the cylinder has illuminated the surface. As stated in chapter 6, the cylinder lights up the scene at a higher polar angle and as a result, surfaces with higher slope appear brighter. It implies that the slope of the side walls of a dimple increases as it moves within the structure. Same holds true for two bright rings of the central hub. Therefore, using the two RTI lighting frames, the shallow versus steep surfaces can be differentiated.



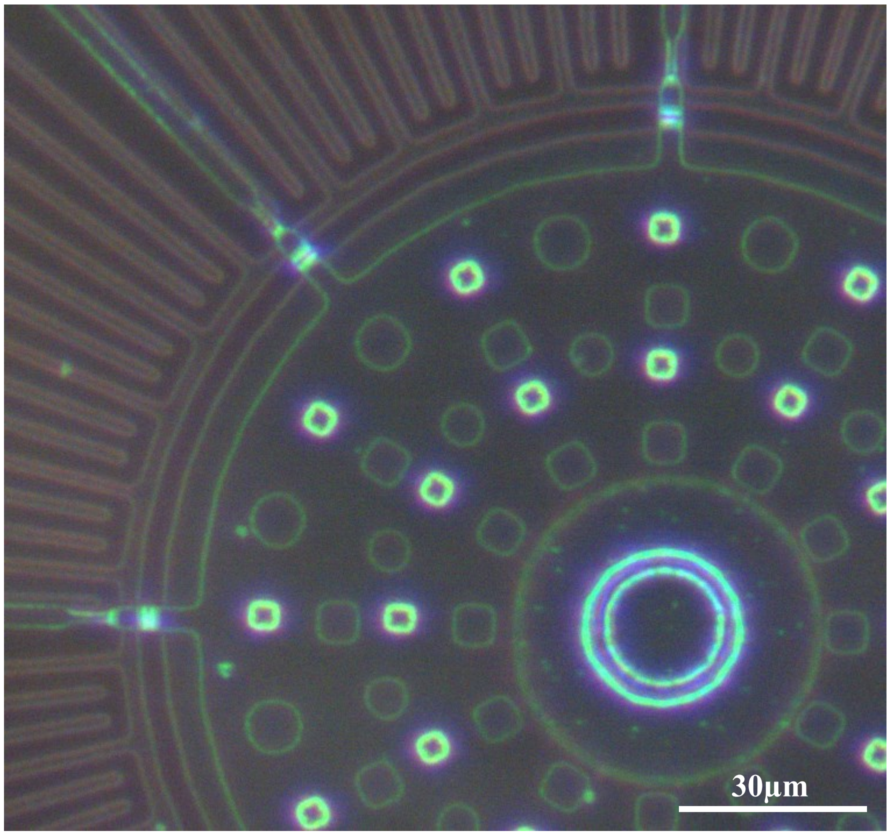


Figure 74: Zoomed-in darkfield images of the MEMS bearing. Top: ring lighting array. Bottom: cylinder lighting array.

To reveal the orientation of facets, the RTI single image with only one ring LED "ON" at a time is combined with the brightfield and is shown in Figure 75. The bearing in this figure is under the illumination of the LED on the right side of the image, and thus the bright regions could correspond to an upward step of an elevated feature on the right or a downward step of a depression on the left. According to the figure, the highlighted region on the bearing hub is on the left side and, considering its circular shape, it can be concluded that the highlights belong to the downward inclined internal sidewalls of the feature. Similarly, on the dimples brightest region is on the left side of the feature, and this would imply that the dimples are depressions and not elevations.

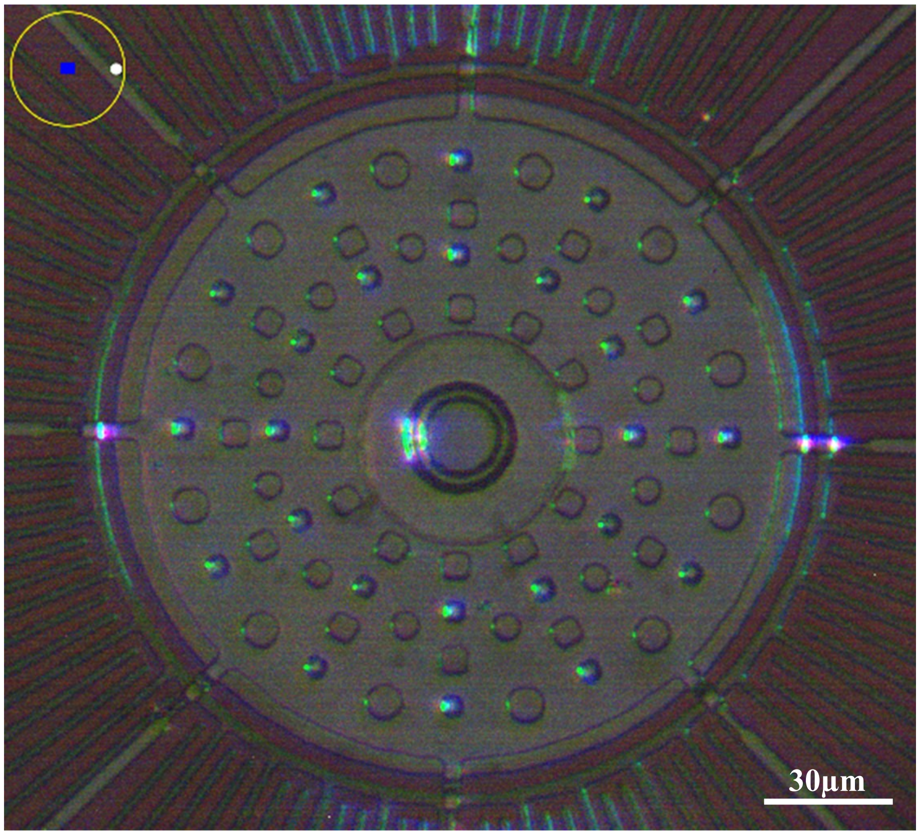
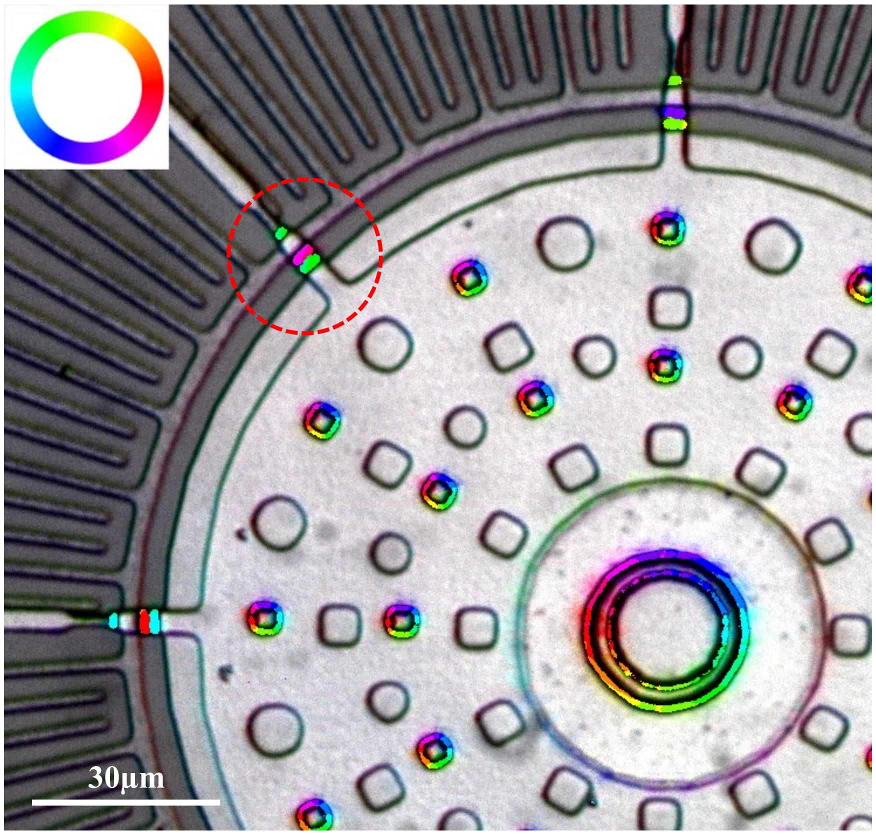


Figure 75: 50/50 composite of a single image and the brightfield image. LED illumination is from the right.

To investigate 3D structure of the MEMS bearing, the HSV images of the scene are shown in Figure 76. The top figure is when the ring illuminates the device and the bottom figure is the cylinder illuminated. Features of the device that correspond to the same color pattern as the HSV color wheel in the upper left corner are elevated features, while the structures that illustrate inverted color patterns correspond to depressions. For example, looking at the hub and the dimples, they both show inverted colors: it can be concluded visually that these features are depressions. As another example, looking at the highlighted region on the 45° spoke: starting from the outside it shows a green line, then a magenta line then a green line: This corresponds to a POLY1 line coming from the outside, rising above the thin POLY0 ring, then dropping down again and finally rising up over the POLY0 disc.



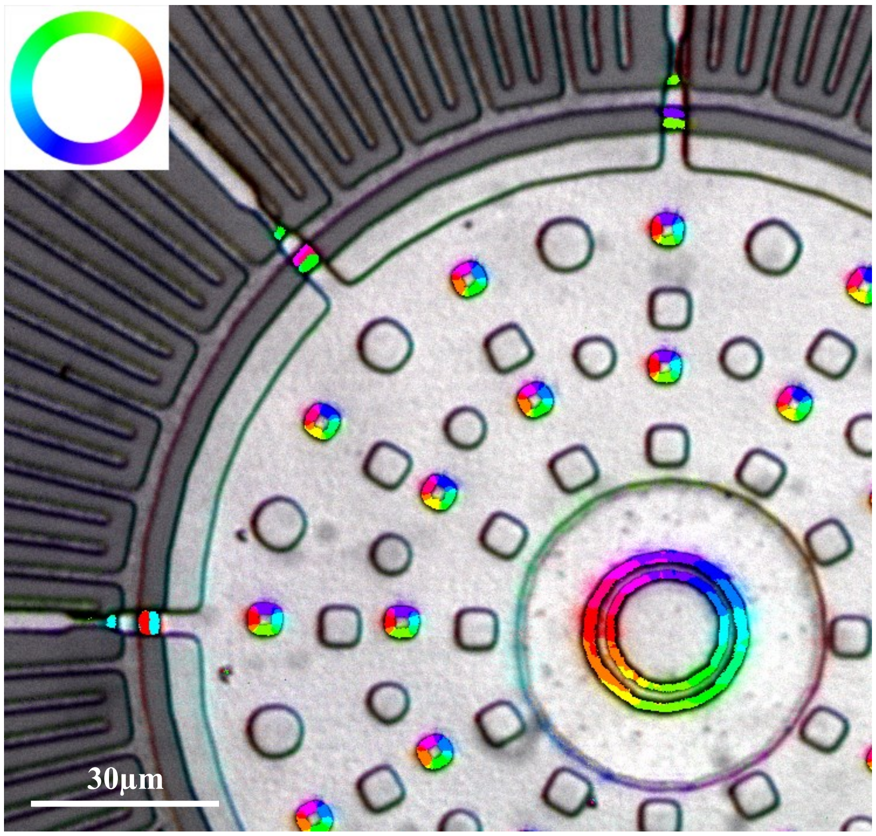


Figure 76: Zoomed-in HSV images of the MEMS bearing. Top: ring lighting array. Bottom: cylinder lighting array.

7.1.1. The Profile View

To facilitate the surface interpretation and visually discern the depressions and elevations, a new feature is proposed in this section. Note that the steep walled holes present few visual cues either in the darkfield images or in the brightfield images. To further develop the micro-RTI Viewer, a new post-processed stage is added to draw a profile view of a device that not only reveals the structures with inclined sidewalls such as dimples but also detects the vertical facets of the features such as holes that are ambiguous in other images.

The principal idea behind this feature is that the sidewalls in the brightfield image appear as dark regions and in the HSV image they appear as colored regions with color indicating slope direction. Therefore, by extracting the dark sidewalls on a brightfield scene and comparing them to their respective angles in the HSV image, it is possible to generate a pseudo-cross-section cutaway view of a scene. Note that this algorithm will only provide a qualitative view of the features by indicating up versus down steps, but will not indicate the relative depths.

The following algorithm is a continuation of the previous RTI-Viewer program, and the concepts such as the image stack and HSV image components (Hue, Saturation, and Value) remain intact.

As shown in Figure 77, firstly, the captured brightfield image of the scene is converted to a grayscale image. Then the grayscale image is plotted, and using "improfile()" MATLAB built-in function user is able to interactively draw a line to specify the cutting plane, as depicted in Figure 78. The coordinates of the endpoints of the drawn line are stored in two matrices of *x_vals* and *y_vals*, and subsequently are used to extract the corresponding intensity value of the pixels from the grayscale image using the "improfile()" function in reverse and store the intensity values in an array named *profile_gray* which is referred to as the grayscale profile.

```
gray_image = rgb2gray(bright_image);
imshow(gray_image)
[~,~,~,x_vals,y_vals]=improfile;
profile_gray = improfile(gray_image,x_vals,y_vals);
```

Figure 77: Part of the MATLAB program to draw a profile line and extract the intensity values across the drawn line.

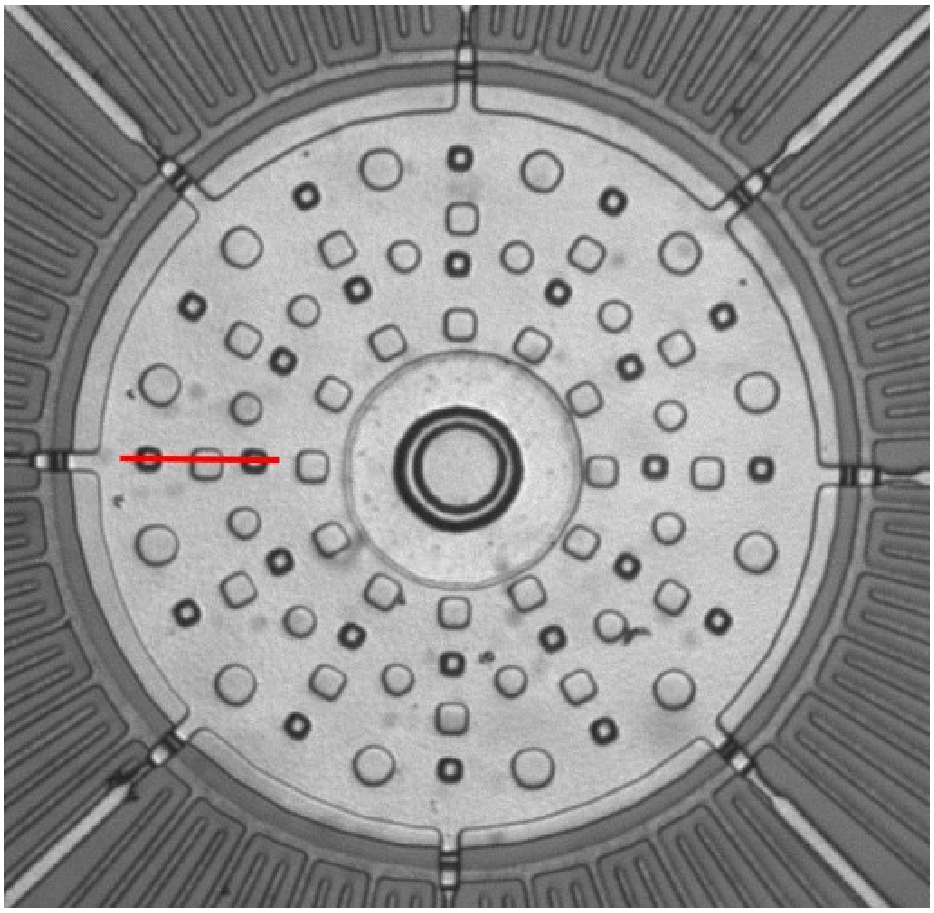


Figure 78: A line drawn by the user on the grayscale image of the MEMS bearing. Note that the line crosses: a dimple, a hole, then another dimple.

To extract and display the profile view of the device at the specified line, the following steps are carried out:

1. The grayscale profile is plotted. Then using the "findpeaks()" MATLAB function, provided by the Signal Processing Toolbox, minima of the grayscale profile are calculated and then illustrated on the grayscale profile figure. However

"findpeaks()" can only find maxima. As the darkest pixels carry the lowest intensity values, instead of using the profile_gray as the function input, *1-profile_gray* is used, as depicted in Figure 79. The second and third arguments are used to ensure the prominence of the selected peaks relative to the surrounding points. The intensity values of the selected pixels and their indices relative to the grayscale profile are stored in the *peaks* and *locations* arrays, respectively. Note that the peak values determined using the "findpeaks()" function, are then re-inverted to indicate the pixels with the minimum brightness. The grayscale profile with the darkest pixels indicated by orange dots is shown in Figure 80 (a).

```
plot(profile_gray)
hold on

[peaks,locations]=findpeaks(1-profile_gray,'MinPeakProminence',0.15);
peaks= -peaks+1;
scatter(locations,peaks)
```

Figure 79: Part of the MATLAB program to detect and plot the minima on the profile line.

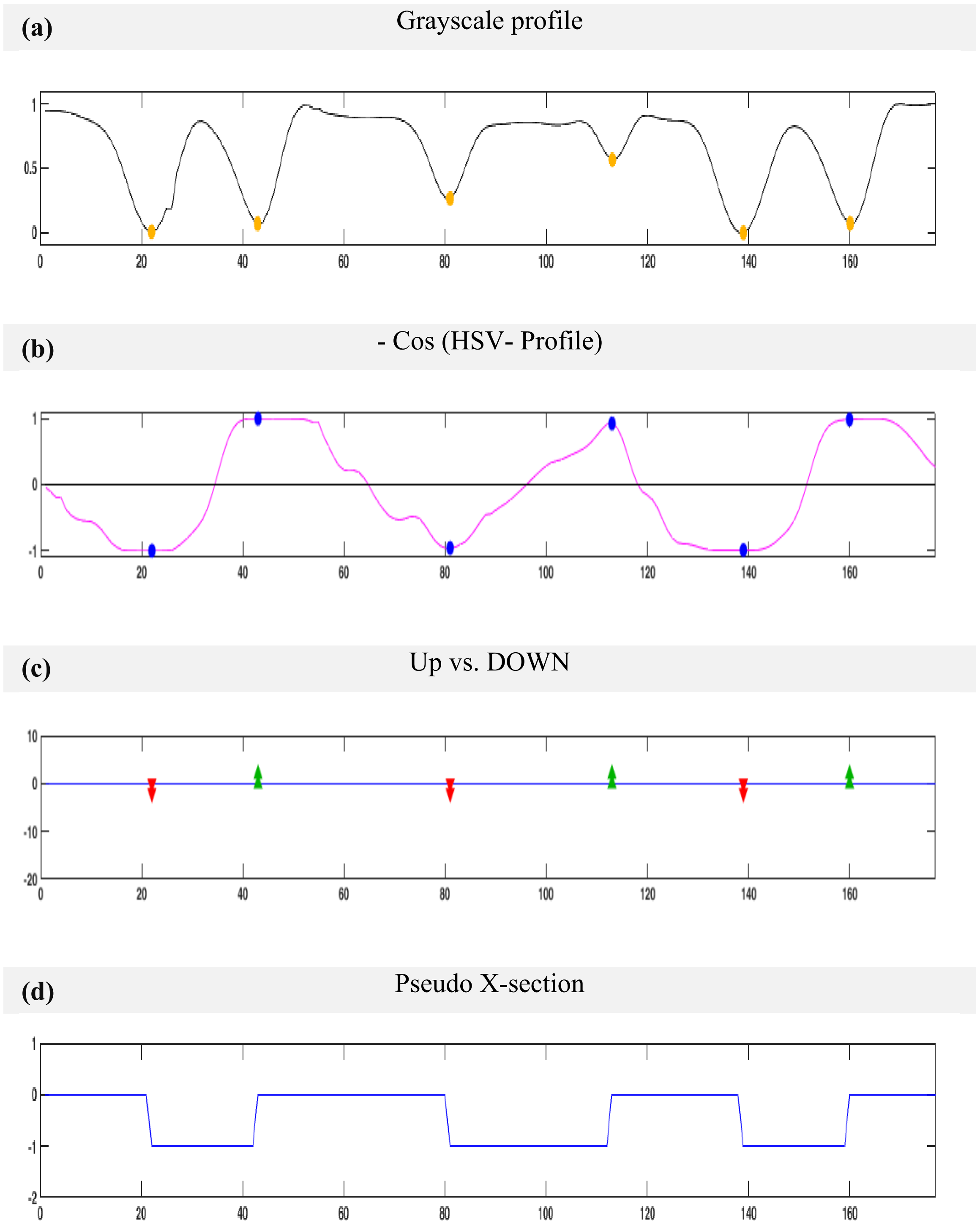


Figure 80: The intensity profile, cosine profile, up/down profile, and pseudo cross-section view, plotted based on the cutting plane line.

2. The same “improfile()” line coordinates will be used to generate a profile from the HSV image to indicate the orientation of the sidewalls where the minima occurs. To this end, firstly, the hue element of the HSV image (values between 0 and 1) are rescaled to lie between 0° and 360° and stored in the *max_theta_degs* array, as shown in Figure 81. As a result, each pixel of the *max_theta_degs* array carries the azimuth angle of the LED that brings the maximum intensity to the pixel. Then the angle of the profile line relative to the horizontal is calculated. Then an array is created in which each pixel represents the angle of the profile line (*profile_ang_image*). With this array the azimuth angle of LEDs at each pixel can be compared to the angle of the profile line. To determine if the hue element of the HSV image represents a step up or a step down, the cosine of (HSV angle - profile angle) is determined for each pixel on the profile line. The resulting vector is called the cosine profile. This is analogous to the dot product of two unit vectors in that its positive or negative sign indicates the directions of the two vectors relative to each other.

```
max_theta_degs = max_theta_hsv*360;
profile_ang = -180/pi()* atan2(y_del,x_del);
profile_ang_image = ones_image*profile_ang;
ang_difference= max_theta_degs - profile_ang_image;
cos_hsv= cosd(ang_difference);
cos_hsv =-1* cos_hsv;
profile_cos = improfile(cos_hsv,x_vals,y_vals);
```

Figure 81: Part of the MATLAB program to generate a cosine profile.

For instance, Figure 82 (top) illustrates the profile view of a hole with two opposite sidewalls: the left one is illuminated by the first LED whose azimuth angle is 0°, and the right one is illuminated by the seventh LED whose azimuth angle is 180°. The yellow arrows show the azimuth angle that the pixel has stored at the location of the walls, and the purple arrows illustrate the angle of a sample profile line. The cosine of the net angle (HSV angle - profile angle) in this case equals +0.8 and -0.8, corresponding respectively to Figure 82 (left) and Figure 82 (right). The +0.8 is to be assigned to a step down and the -0.8 to a step up, note the reversal of the signs. By following the pixels on the profile line, if the cosine of the net angle is positive, a depression occurs, and if the cosine of the net angle is negative, an elevation occurs. In MATLAB this cosine profile is calculated as: *-1*sign(cosine (HSV angle – profile angle)).*

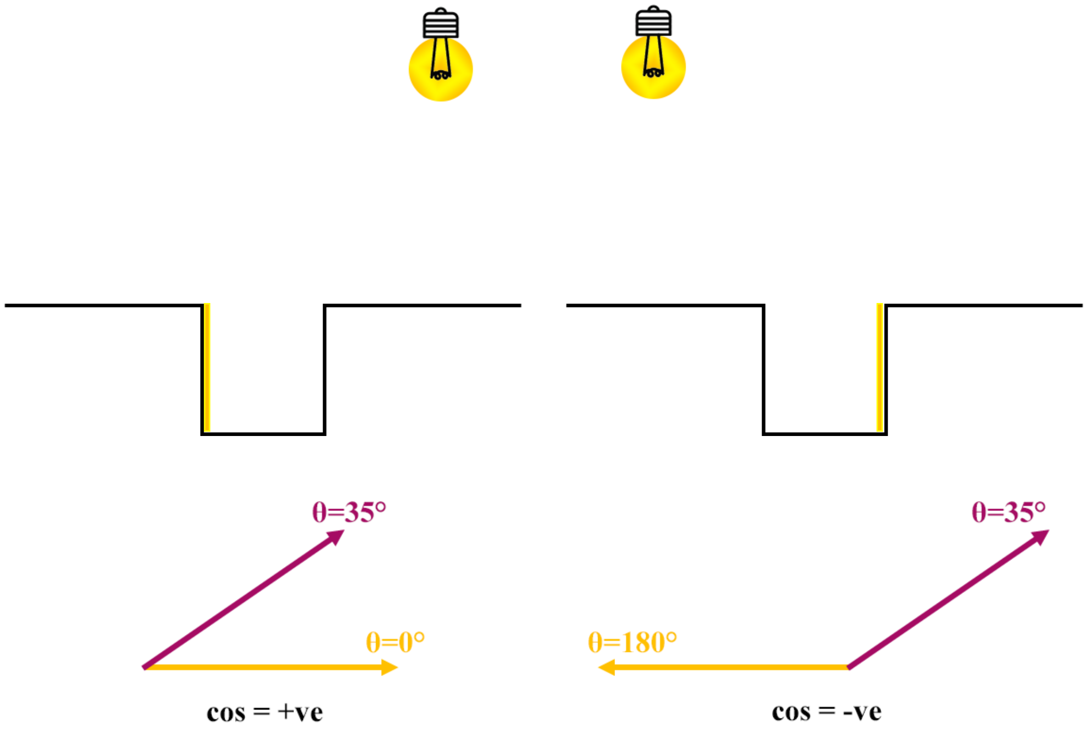


Figure 82: How the azimuth angle of the LEDs exhibits the up/down steps.

The cosine profile is plotted in Figure 80 (b), and the grayscale minima locations are indicated by blue dots. Figure 80 (c) represent an array called the up/down profile that stores only the sign of the cosine profile at the grayscale minima and represents a positive one with an up arrow, and a negative one with a down arrow.

Figure 80 (d) shows a pseudo cross-section generated by starting at zero and moving up or down ±1 at every minima point in the previous up/down plot. Comparing the results of the profile view with the original grayscale image in Figure 83: starting from the left the red line crosses a dimple, a hole and then another dimple, which is indicated in the profile view with three consecutive sets of falling and rising edges. Note that no information regarding the depth of the features is displayed: the profile view shows only the orientation differences on a surface.

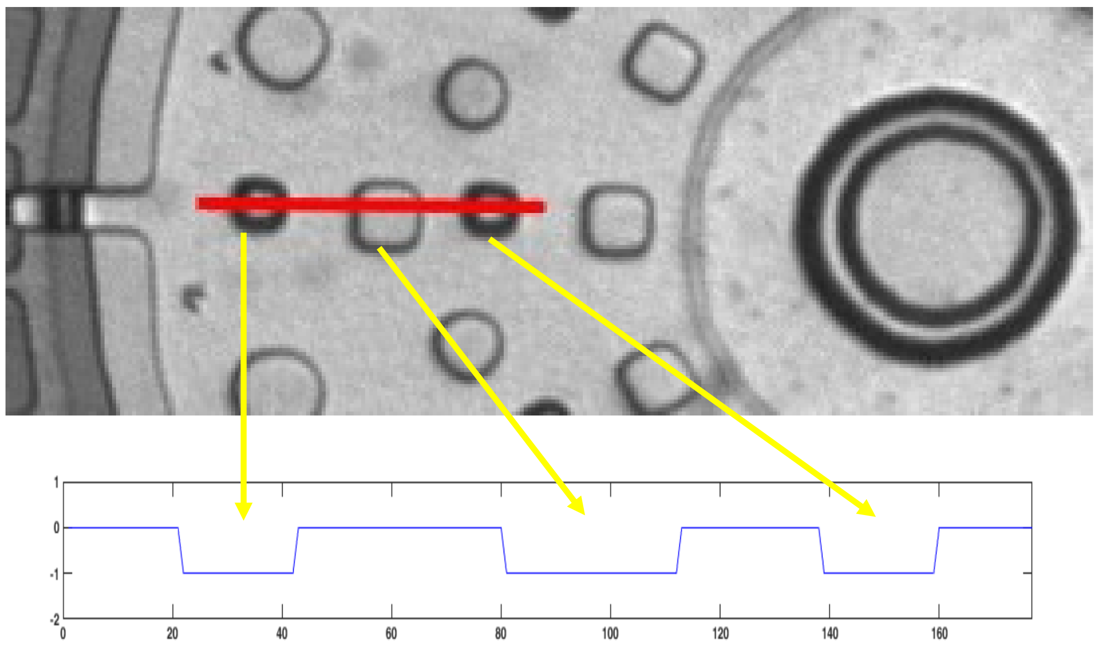


Figure 83: TOP: zoom-in of user drawn line, note that the line crosses: a dimple, a hole, then another dimple. Bottom: pseudo cross-section view.

To further examine the pseudo cross-section, two other cutting plane lines are shown in Figure 84, and the profile views are plotted and compared with the MUMPs layout of the device in Figure 85.

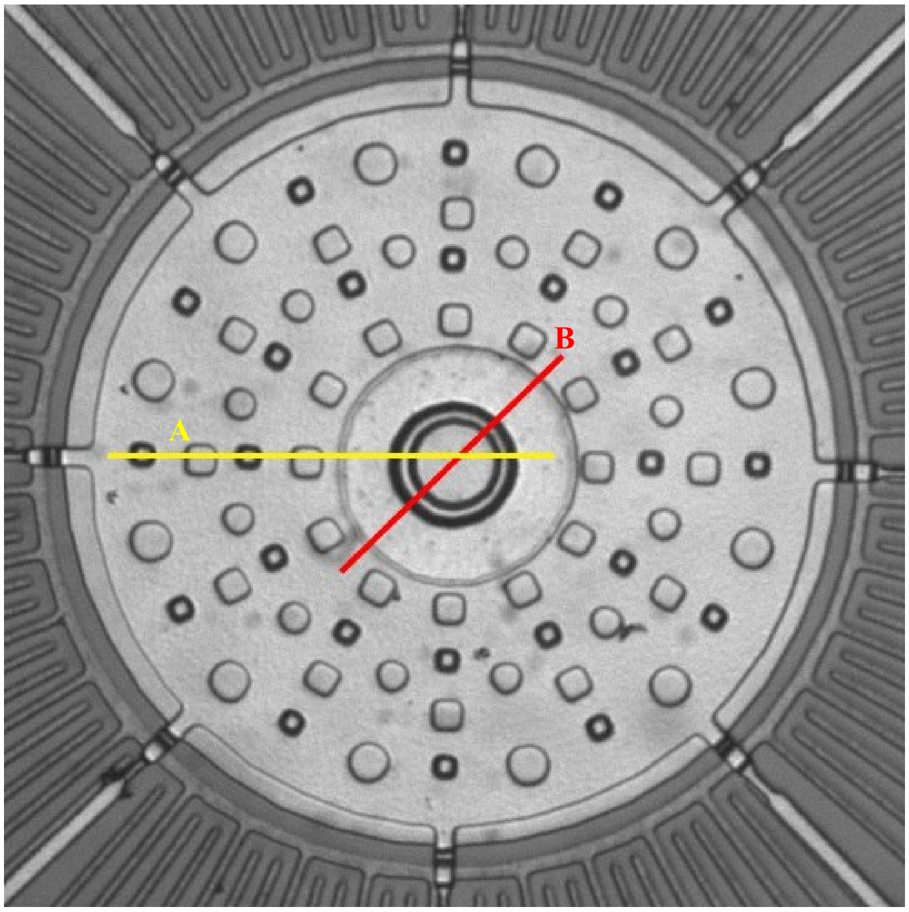


Figure 84: Two different cutting plane lines to examine the profile algorithm. Compare to Figure 87.

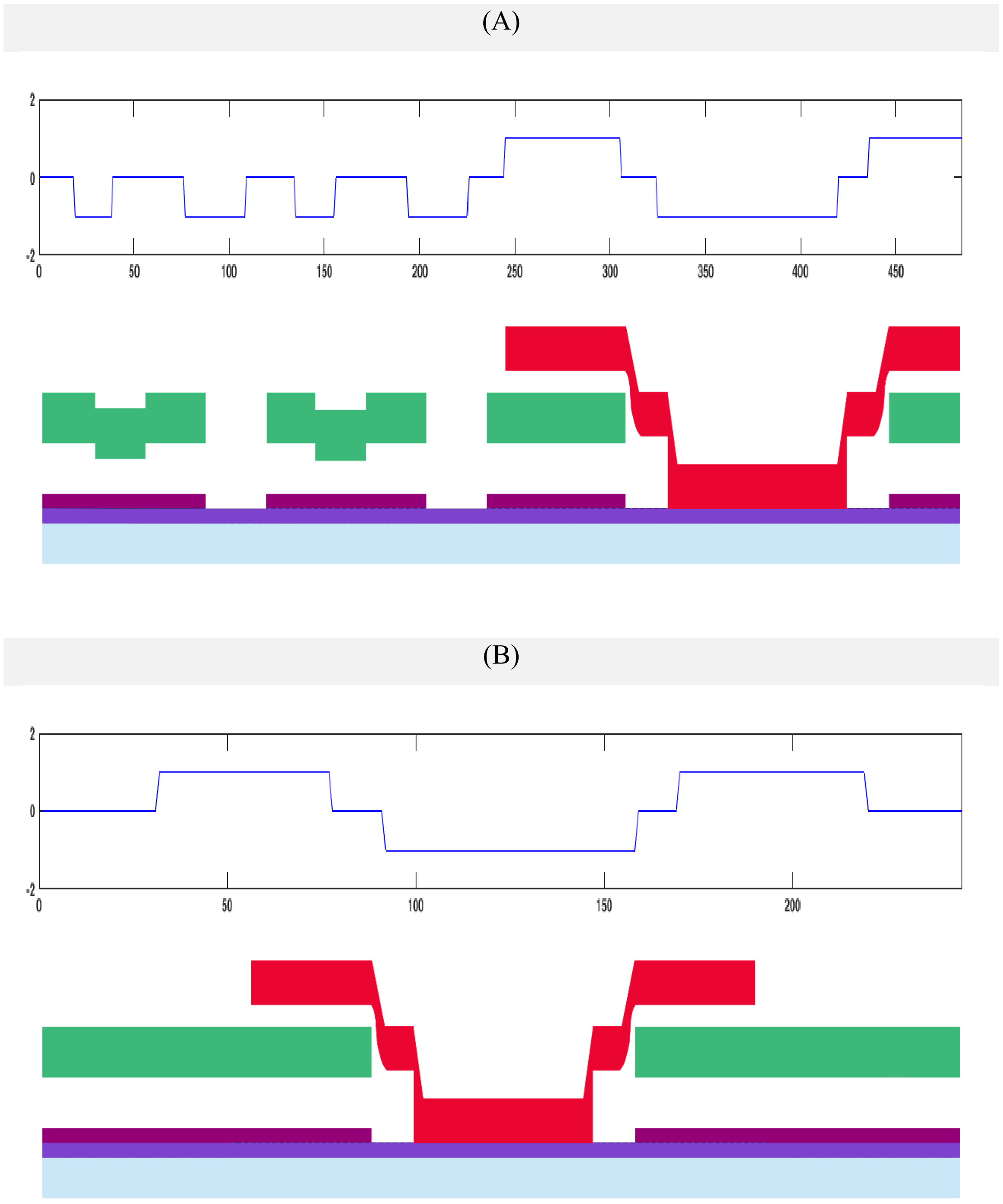


Figure 85: Profile views of the MEMS bearing with cutaway MUMPs layouts: (A) across line A, and (B) line B in Figure 86.

According to the MUMPs layout of the MEMS bearing, the yellow line (A) crosses the following features from the left: a dimple, a hole, a dimple, a hole, a POLY2 step up, two consecutive POLY2 steps down. It implies that there are 4 consecutive depressions, one elevation, and two consecutive depressions across the profile line. Therefore, the profile view in Figure 85 (A) corresponds to the features that the yellow line crosses.

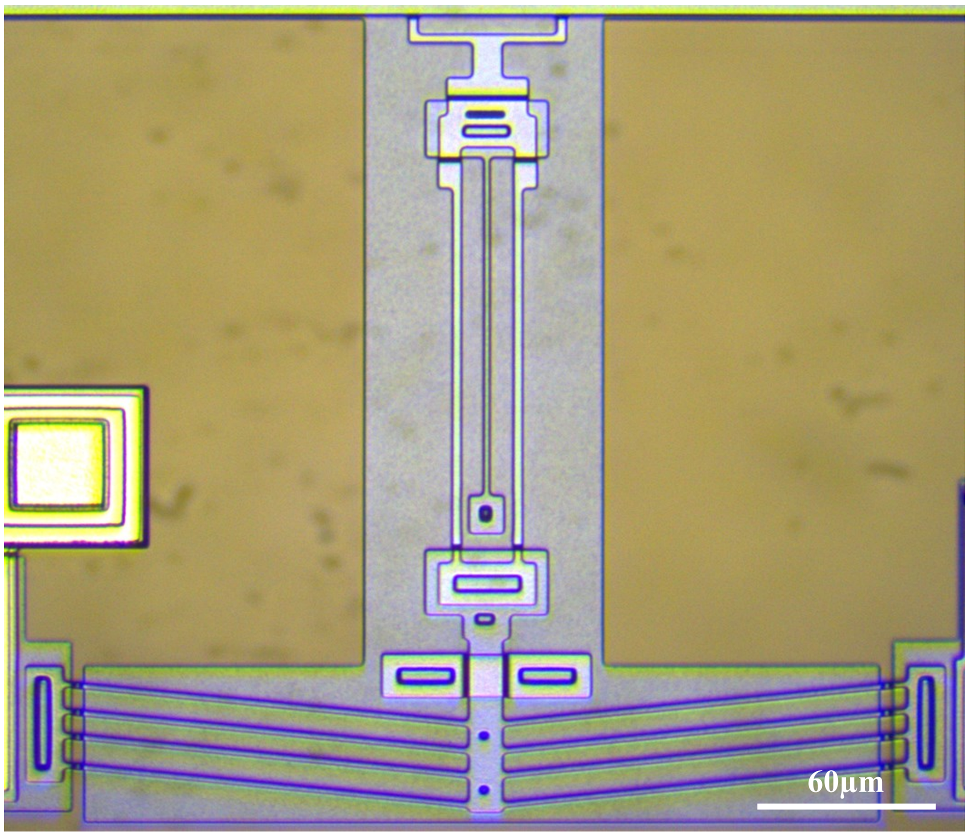
The red cutting plane line (B) crosses the hub which is made of a POLY2 layer at an elevation that slides down twice to reach the substrate followed by two rising edges since the surface is symmetrical across the red line, and then a final depression that corresponds to the side edge of POLY2 hub. Thus, firstly, the profile view should detect an elevation, and then two falling edges of the anchor should be discerned followed by two rising edges, and lastly as the POLY2 is etched away, another falling edge should appear. Looking at Figure 85 (B) the profile view corresponds to the description and the MUMPs layout.

Therefore, the proposed new feature added to the micro-RTI Viewer program displays rising and falling edges across a line drawn by the user, and thus can serve as a tool for identifying complex MEMS structures.

7.2. Chevron-based Vertical Electro-Thermal Micro-Actuator

Electro-thermal actuators are MEMS devices that can generate output motion at low operating voltages. When an electrical current is applied, the components of a micro-actuator demonstrate a thermal expansion upon heating that is proportional to the distribution of the current density. The mechanical expansion of the components can drive the device towards the X, Y, or Z axes. The previous chapter addressed a two-beam thermal actuator that generates in-plane motion parallel to the substrate.

Figure 86 (top) shows the brightfield image and (bottom) the MUMPs layout of an out-of-plane thermal actuator that drives a folded beam structure using a chevron thermal actuator. The folded beams are made of POLY1 and POLY2 layers lying at different heights and as a result out-of-plane motion can be produced. The chevron thermal actuator itself is composed of four pairs of ~90 μm length, V-shape beams (POLY1) that at the ends are anchored to the substrate (POLY0), and at the center come into contact with a suspended shuttle 2 μm above the substrate (POLY1).



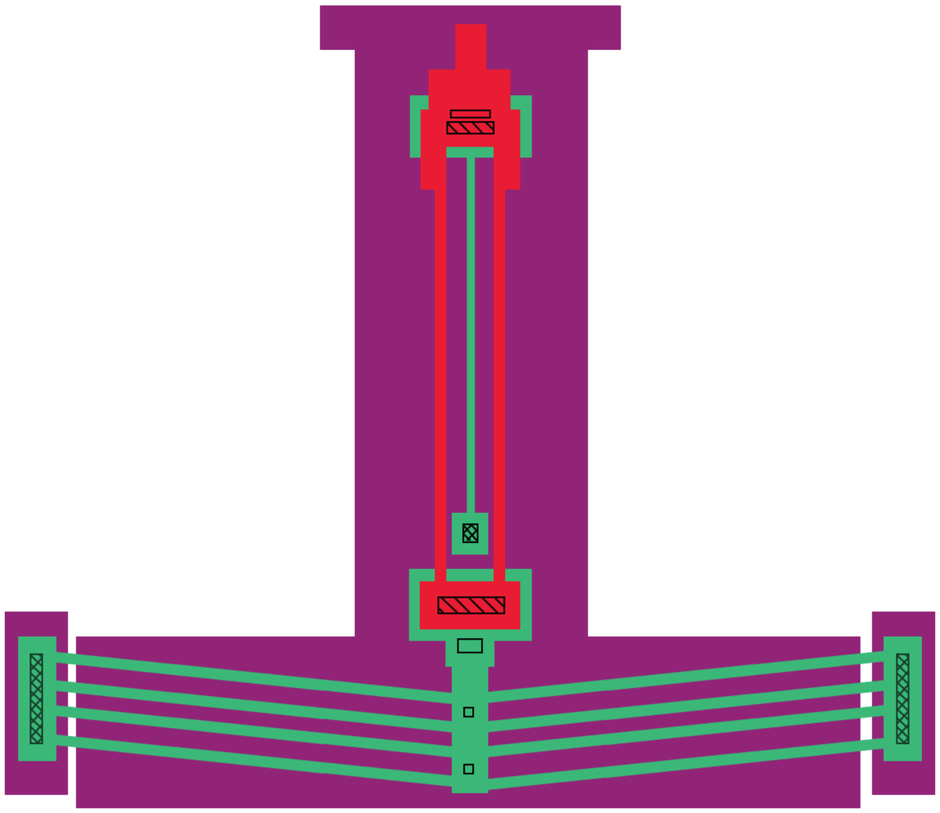


Figure 86: Top: Brightfield image of a chevron-based out-of-plane thermal actuator. Bottom: MUMPs layout.

To extract information regarding the thermal actuator out-of-plane displacement, only the cylinder NeoPixel was installed on the 25× objective lens close to the MEMS chip. Two sets of 14 images were captures: the before state with the actuator OFF, and the after state with the actuator ON. The micro-RTI Viewer program shown in the previous chapter was modified slightly: the up/down arrows now toggle between before/after images rather than between ring/cylinder images. Switching consecutively between OFF state and ON state images of a device facilitates detecting in-plane displacements and enables the user to identify visual cues regarding out-of-plane motion.

By applying voltage to the device, the current passes through the V-shape beams of the chevron actuator and as a result of thermal expansion of the beams, the device generates an in-plane motion that cannot be perceived using the brightfield images. Figure 87 depicts the magnified brightfield images of the chevron actuators (top) before actuation and (bottom) after actuation. A slight difference in the brightness of the center shuttle surface may be noted, which cannot be considered as a definite sign of displacement.

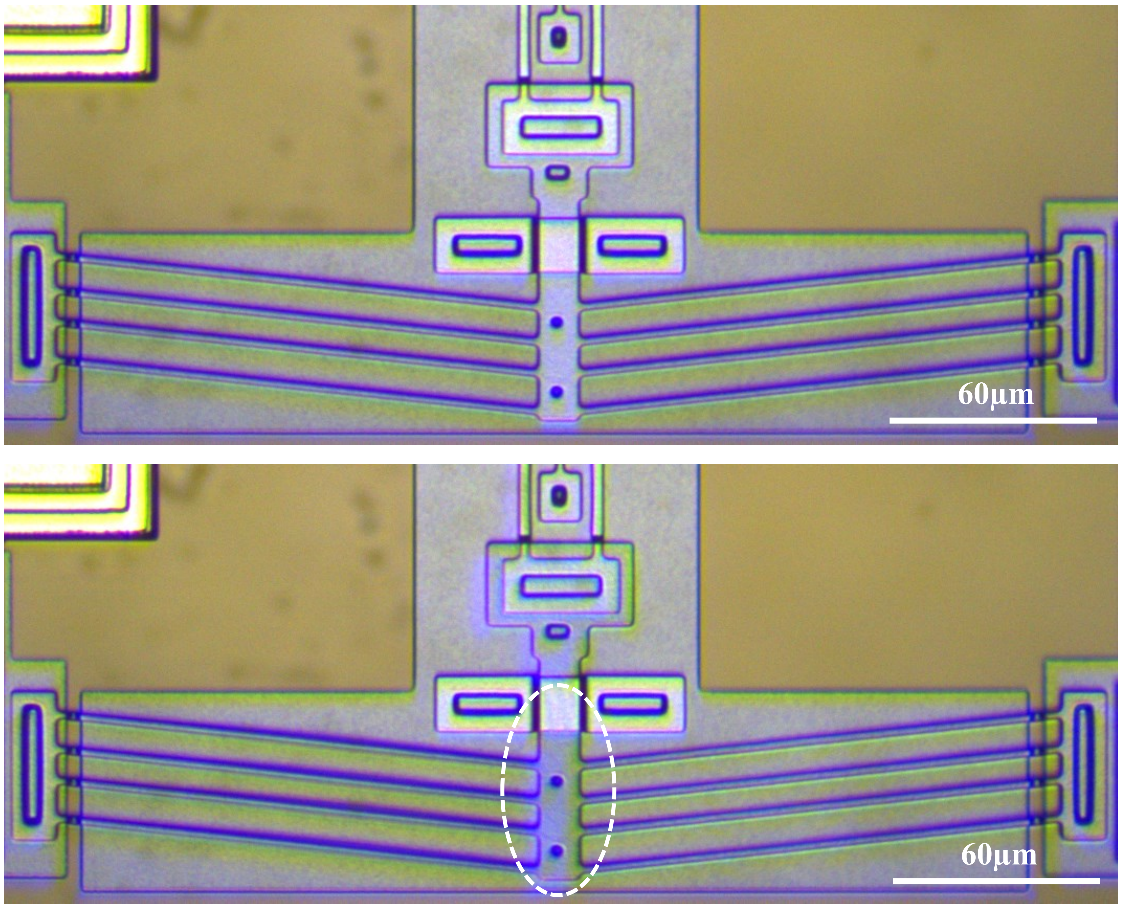


Figure 87: Brightfield image of the chevron actuator. Top: before actuation (OFF). Bottom: after actuation (ON). Note slight darkening at chevron center.

The darkfield images of the same device before and after actuation is depicted in Figure 88 and reveal additional visual representations that may correspond to the out-of-plane motion of the device. According to the figure, the main difference between the two states of the device is an increase in the surface brightness of the actuator center after applying the voltage. The reason could be an uneven thermal expansion of the beams that leads to the deflection of the shuttle towards some direction, but the direction cannot be discerned in the darkfield image. However, the darkfield image successfully revealed a visual difference that arises from an out-of-plane motion. To further investigate the motion, the HSV images are extracted to visually display the geometrical properties of the scene and provide clues regarding the displacement.

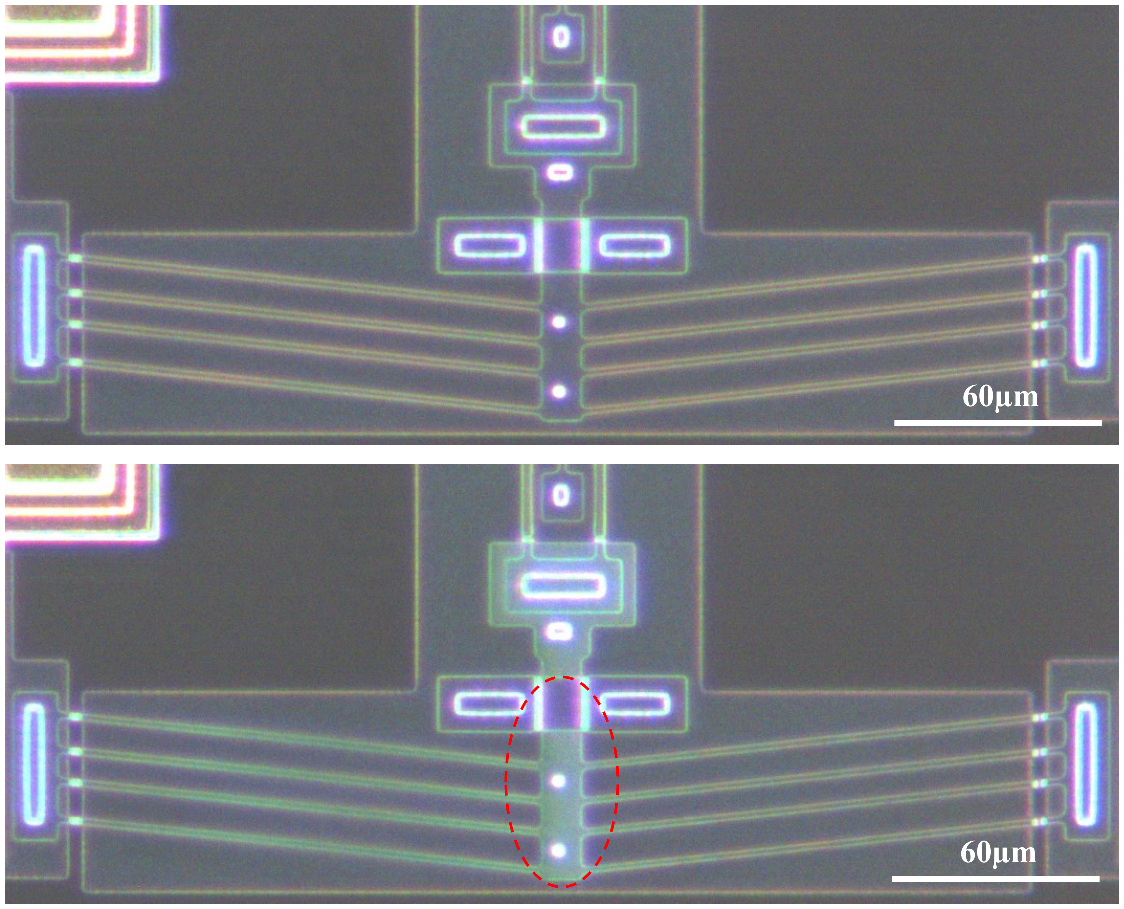


Figure 88: Darkfield image of the chevron actuator. Top: before actuation (OFF). Bottom: after actuation (ON). Note lightening of the chevron center.

Figure 89 shows The HSV images of the thermal actuator before and after actuation. The colored pixels indicate the direction towards which they are tilted with respect to the HSV color wheel. Figure 89 (top) shows that the shuttle and the cantilever attached to it are flat and as a result, no colored pixels can be detected on their surface unless it belongs to a dimple, anchor, or sloping sidewall. However, after being actuated, according to Figure 89 (bottom), the shuttle center and cantilever are markedly tilted towards the LED on the right side of the image.

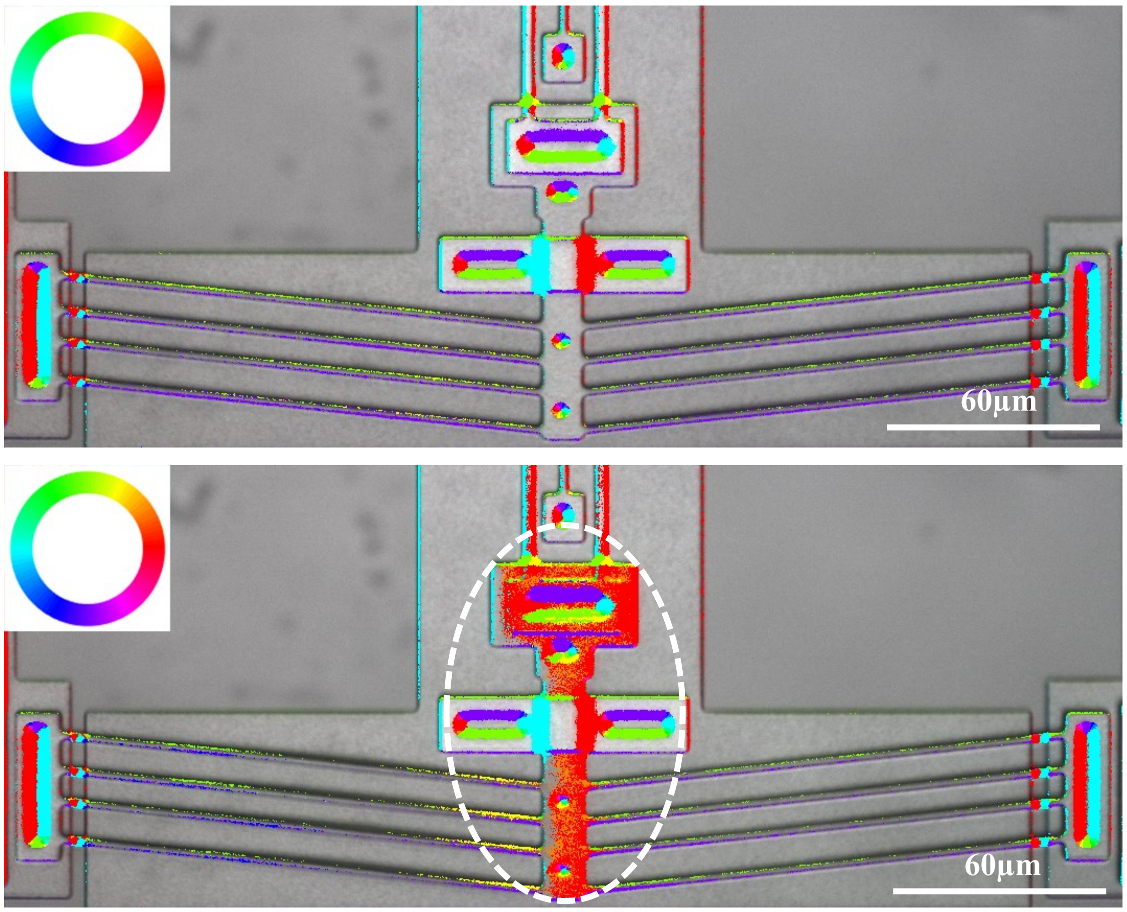


Figure 89: HSV image of the chevron actuator. Top: before actuation (OFF). Bottom: after actuation (ON). Note the rightward tilt.

Browsing the RTI single images and the composite images (50/50 contribution of a single image and the brightfield image) can also reveal the fact that the shuttle is leaned towards the right. Among the 12 RTI composite images, it can be discerned that the surface brightness of the shuttle remains unchanged except for the image that illuminates the surface from the right side of the device. Figure 90 (top) shows the left/ right illuminated composite images in the OFF state, and Figure 90 (bottom) shows the left/ right illuminated after the device is ON. It can be discerned that in the left images both the ON and OFF images show the surface with the same brightness. However, the right images, despite the constant lighting direction, exhibit an alteration in the brightness of the surface between OFF and ON.

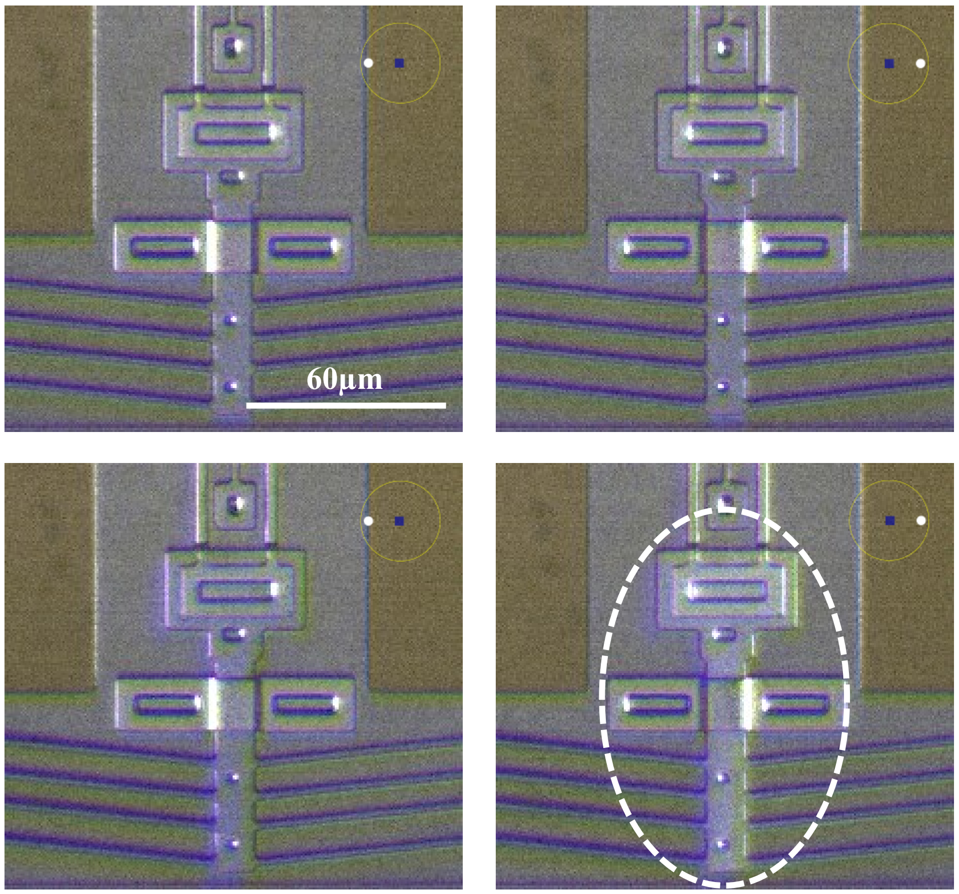


Figure 90: The composite images of the chevron actuator. Top: before actuation (OFF). Bottom: after actuation (ON).

According to the schematic on the upper right corners, the actuator has been tilted towards the LED on the right side after the V-shape beams of the chevron actuator have been thermally expanded. Therefore, browsing the images provided by the micro-RTI imaging technique provided enough evidence to draw conclusions regarding the out of plane displacement of the vertical thermal actuator.

The present chapter involved interpreting two-dimensional images in terms of three-dimensional subject structure and out-of-plane displacement for two different MEMS devices. A sequence of images captured using the micro-RTI Capture program along with the post-processed images achieved using the micro-RTI Viewer program can be used to enhance viewer's understanding of MEMS devices when there are no visual clues in the conventional brightfield image.

8. Conclusion & Future Work

In this thesis, an imaging technique based on the Reflectance Transformation Imaging (RTI) was presented to study and characterize the three-dimensional structure of MEMS devices. The main idea behind the presented imaging technique is to illuminate a micro-scale subject from lateral directions rather than conventional in-line illumination, and as a result expose non-planar surfaces to light and make them visible by the camera. The technique was required to go beyond the conventional brightfield images by facilitating the optical inspection of geometric complexities and out-of-plan displacements of MEMS devices and avoiding the necessity of using high-cost imaging techniques such as SEM.

Given the significant difference between the scale and reflectance characteristics of MEMS devices and typical RTI subjects, the proposed RTI setup was designed for micro-scale subjects to be viewed through an optical microscopy equipped with a digital camera and two 12 LEDs NeoPixels (Ring & Cylinder) that are mounted on an objective lens.

The concept was implemented by an algorithm that communicated with both the camera and individual lights to capture a sequence of images that incorporates: 1) a conventional brightfield image, 2) the darkfield image illuminated by the 12 LEDs of the lighting frame, and 3) RTI individual images illuminated by a single LED of the lighting frame. Each lighting frame delivered a set one 14 images.

Upon comparing the RTI images with the brightfield images, it was demonstrated that the brightfield image contains an ambiguous visual representation of a MEMS device. To detect the sidewalls and non-flat surfaces, the darkfield image should be observed that draws the attention to all the inclined facets that are leaned towards the LEDs. To discern the orientation of a specific sloped surface, RTI individual images can be browsed to identify the LED that brings the maximum intensity to the surface. It was demonstrated that the surface slope of the sidewalls can be distinguished by comparing the RTI images delivered using the ring lighting frame versus the cylinder lighting frame.

This observation argued that the dimples, anchors and VIAs have more inclined sidewalls at depth than at the edges.

To facilitate the process of interpreting a MEMS surface, an image post processing algorithm provided a color-coded image (HSV image) that successfully identified and visually represented spatial orientation of lateral facets and created an illusion of a 3D image. Additionally, the algorithm was developed to be able to study the surface anomalies along a line drawn by the user and draw a profile view of the device. The profile view corresponds to the topography of the scene across the line in terms of identifying depressions and elevations, but not the depth of features.

The algorithm provided an interactive visual representation of the scene by enabling the user to browse through the captured images and post-processed illustrations to compare the two states of the ring and cylinder illumination for a single scene, and extract the profile views across multiple lines.

It was demonstrated that the boundaries of the proposed algorithm can be extended to include the visual recognition of out-of-plan motions. This fact was represented by comparing a sequence of captured images before and after the out-of-plane displacement of a device. The darkfield image demonstrated the out-of-plane motion through an alteration in surface brightness, and the HSV image exhibited a color variation at the points where the displacement occurred.

A range of MEMS devices that include three-dimensional topographical features and out-of-plane displacements have been studied to re-enforce the fact that the proposed technique has the ability to distinguish lateral facets from flat surfaces, elevated features from depressions, vertical sidewalls from sloped sidewalls, and out-of-plane motion of a device from its equilibrium state. The main advantage of the proposed algorithm in comparison to the well-known imaging techniques is the interactive visual analysis principle that facilitates and accelerates the process of interpretating the pseudo-three-dimensional images without incurring cost, surface preparation or time limitation.

Future research could further to enhance the image quality by employing a light source with collimated light rays that focus at a certain point on a surface. This could minimize the dispersion of light with distance and thus enhance the intensity of the images. A key limitation is the confined space under the microscope and the need for very compact lighting frames. A useful feature would be to configure a more compact lighting frame that is able to illuminate the scene at varying polar angles rather that only two angles.

The proposed profile view could be improved to take into account the depth of the features by differentiating two comparable features (such as dimples and holes). This might be done by comparing the thickness of the sidewalls and the separation distance between two opposite walls within a feature.

The breadth of the study could also be extended to include investigating the MEMS surface physical failures due to environmental factors such as contamination, and to incorporate other non-MEMS micro-scale subjects.

Appendix A-1: Micro-RTI Capture MATLAB Program

```
close all
clearvars
clc
%----------------------------------------------------------------
% Display a dialog box for the user to specify a directory.
disp('Select directory to save images...')
path = uigetdir;
% Set the selected directory to be current working directory.
cd(path)
% image file extension = png, pgm or jpg % set to .xxx to omit
file_extension = 'png';

image_name =  'Img_#';

disp(' ')
display([num2str(num_leds),' x ',image_name,'.',file_extension]);
disp(' ')

%----------------------------------------------------------------
disp('-------------------');
disp('Arduino neopixel')
disp('-------------------');

disp(' ')
disp('Checking for Arduino...')

if exist('ard','var')
    disp(['Already Connected to:  ',ard.Board,'  on ', ard.Port]);

else
    com_ports = seriallist;
    num_ports=size(com_ports,2);
    fprintf('PORTS: ')

    for i = 2:num_ports
      fprintf(com_ports(i));fprintf(', ')
    end
    disp(' ')

    ard = arduino ('COM3', 'Uno', 'Libraries', 'Adafruit\NeoPixel');
    disp(['Connected to:  ',ard.Board,'  on ', ard.Port]);
    disp('Checking for Adafruit Neopixels...')

end
 num_leds=12;
neostrip = addon(ard, 'Adafruit/NeoPixel', 'D6',num_leds);
%----------------------------------------------------------------
closepreview

disp(['FLIR USB camera image capture'])
vid_mode = 'BGR8';
```

```
disp(['Video mode = ',vid_mode])
disp(' ')

vid = videoinput('mwspinnakerimaq', 1, vid_mode);
src = getselectedsource(vid);
% exposure command to camera should be in microseconds
src.AutoExposureExposureTimeUpperLimit = 15*1000;
pause(0.25)

% in case you want to avg multiple frames
num_avgs=1;
vid.FramesPerTrigger = num_avgs;
%--------------------------------------------------------------
% set up preview figure window
fig = figure('NumberTitle','off','MenuBar','none',...
             'units','inches','Position',[7.5 2.75 12 8]);
fig.Name = 'Camera Preview';

ax = axes(fig,'Units','Normalize','Position',[0 0 1 1]);
start(vid)
frame = getdata(vid);

im = image(ax,zeros(size(frame),'uint8'));
axis(ax,'image');

disp('Previewing camera...')
preview(vid,im)
%--------------------------------------------------------------
disp(' ')
disp('1. Turn ON Brightfield light.')
disp('2. Press any key to save Brightfield image ...')
pause

start(vid)
frame = getdata(vid);

my_image = frame;
my_image= im2double(my_image);
imwrite(my_image,[image_name,'00','.',file_extension])
pause(0.5)

disp(' ')
disp('1. Turn OFF Brightfield light.')
disp('2. Press any key to adjust LED 1...')
pause

writeColor(neostrip, 1, [1,1,1]);

%default exposure time in milliseconds
exposure = 15;
%exposure command to camera should be in microseconds
src.AutoExposureExposureTimeUpperLimit = exposure*1000;
pause(0.25)
%bring current figure to front
figure(gcf)
```

```
disp(' ')
disp('Press enter to accept exposure time')
loop = 1;
while loop
   exposure_new =  input(['Exposure time in msec: (', num2str(exposure),')
']);
    if isempty(exposure_new)
        loop = 0;
    else
        exposure= exposure_new;
        src.AutoExposureExposureTimeUpperLimit = exposure*1000;
        pause(0.25)
        % bring current figure to front
        figure(gcf)
    end
end

closepreview
stop(vid);
close all

wait_time = 1;
neostrip.Brightness = 1;

disp(' ')
disp(['3. Taking 1 -> ',num2str(num_leds),' images...'])

fprintf('Image # ')

for i = 1:num_leds +1
    start(vid);
    if i <= num_leds
        fprintf([num2str(i),', ']);
        writeColor(neostrip, i, [1,1,1]);
    else
        fprintf(['darkfield']);
        writeColor(neostrip, 1:num_leds, [1,1,1]);
    end
    pause(wait_time);

    frames=getdata(vid,num_avgs,'double');
    if num_avgs == 1
        my_image = frames;
    else
        my_image = squeeze(mean(frames,4));
    end

    my_image= im2double(my_image);
    imshow(my_image)
    pause(wait_time)
    writeColor(neostrip, 'off');

    if i <10
        imwrite(my_image,[image_name,'0',num2str(i),'.',file_extension])
    else
```

```
        imwrite(my_image,[image_name,num2str(i),'.',file_extension])
    end
end
clear neostrip
stop(vid);
close all
display('Done.');
```

Appendix A-2: Micro-RTI Viewer MATLAB Program

```
close all
clearvars
clc
%--------------------------------------------------------------
% Display a dialog box for the user to specify a directory.
disp('Select a directory...')
path = uigetdir;
% Set directory from above to be current working directory.
cd(path)

% image file extension = png, pgm or jpg % set to .xxx to omit
file_extension = 'png';

% Create list of images with defined extension
list_of_images = dir(strcat(path,'\*.',file_extension));
num_pics = size(list_of_images,1);

% Create an array of images names
file_names(1:num_pics,:) = char(list_of_images(1:num_pics).name);
disp(['# of .',file_extension,' images = ',num2str(num_pics)]);
disp(' ')

num_leds=12;
% Disable the warning that appears when expected number of images is not
found
if num_pics ~= num_leds+2
   disp(['ERROR: did not find expected ',num2str(num_leds),' + 2 pics!'])
   disp(' ')
   disp('Pausing... press ^C')
   pause
end
%--------------------------------------------------------------
%Read in first image
 my_image = imread(file_names(1,:));
%Optional Crop images
roi_val = 0;
if roi_val == 1
    % Disable the warning that appears when the imcrop is too big to fit the
screen.
    %MATLAB automatically resizes the image so the warning is not important.
    warning('off','images:initSize:adjustingMag');

    figure('Color',[.8 .8 .8],'Name','Choose ROI')
    title('Choose ROI (dbl. click or Enter to finalize)'),hold on
    disp('Choose an ROI (dbl. click or Enter to finalize) ...')
    [roi, rect]=imcrop(my_image);
    x_size=size(roi,2);
    y_size=size(roi,1);
else
    x_size=size(my_image,2);
    y_size=size(my_image,1);
end
close all
```

```
%----------------------------------------------------------------
disp('Reading images')
%Prelocating the arrays that store images in color and grayscale
image_stack =zeros(y_size,x_size,num_leds+2,3);
gray_stack =zeros(y_size,x_size,num_leds+2);

theta_val = 1;
for i=1:num_pics

    if mod(i,2)==0
        fprintf('.');
    end
    ith_image = imread(file_names(i,:));
    if roi_val == 1
        ith_roi = imcrop(ith_image,rect);
    else
        ith_roi = ith_image;
    end
    %imshow(ith_roi)
    % Convert uint8 to double, avoid 0-255 rounding errors
    ith_roi = im2double(ith_roi);

    image_stack(:,:,theta_val,:)=ith_roi;
    gray_stack(:,:,theta_val)=rgb2gray(ith_roi);

    theta_val = theta_val+1;
end

%Change the Order of images if required
%The theta order is: Brightfield - Darkfield - num_led images
% THETA OFFSET: need to know where LED #1 is located
% each led is 30 degs, e.g. 3 = 90 deg
theta_offset = 3;  %
temp = image_stack(:,:,3:num_leds+2,:);
    image_stack(:,:,3:num_leds+2,:) = circshift(temp,theta_offset,3);
temp = gray_stack(:,:,3:num_leds+2,:);
    gray_stack(:,:,3:num_leds+2,:) = circshift(temp,theta_offset,3);

bright_image = squeeze(image_stack(:,:,1,:));
dark_image = squeeze(image_stack(:,:,2,:));
darkgray_image = rgb2gray(dark_image);

% correct for nonuniform brightness
% remove dark image glare
top = 0;
if top
    radius=40;
    se = strel('disk',radius);
    dark_image = imtophat(dark_image,se);
end

fprintf('\n');
disp(' ');
```

```
%---------------------------------------------------------------
% section to find spectrum plots
ones_image =ones(y_size,x_size);
zeros_image =zeros(y_size,x_size);

% Extract the grayscale version of RTI single images
gray_stack_filt = squeeze(gray_stack(:,:,3:num_leds+2));
% Determine the maximum value each pixel carries and its location
[max_image, theta_indices] = max(gray_stack_filt,[],[3 4],'linear');
[~,~, max_theta] =ind2sub([y_size x_size num_leds],theta_indices);
[min_image, ~] = min(gray_stack_filt,[],[3 4],'linear');

% convert theta (1 -> 12) to 0 -> 1
max_theta_hsv = (max_theta-1)/12;
%---------------------------------------------------------------
theta = 1;
weight = 0.5;
brighten = 3;
opposite= 0;
paired= 0;
display_mode = 1;
threshold = round(graythresh(darkgray_image),2);
on_off = 1;

fig_RTI = figure('Name','IMAGE','units','inches','Position',[6 2 16 16]);
imshow(bright_image)
RTI_view = gcf;

title('BRIGHT FIELD','Color',[.5 .5 .5],'Fontsize',14)


% WAIT FOR KEYPRESS
disp(' ')
disp('Make sure cursor is over RTI image')
disp('Waiting for keypress...')

key_value = 0;
while key_value ~=113
    k = waitforbuttonpress;
    % 28, 29 leftarrow, rightarrow
    % 30, 31 uparrow, downarrow
    % 32 space
    % 113 q = quit

 key_value = double(get(gcf,'CurrentCharacter'));
 if size(key_value,1)==0
     key_value = 1;
 end

    switch key_value
        case 63
        % pressed ?
            disp(' ')
            disp('COMMANDS:')
            disp('q = quit')
            disp('1 = Brightfield ')
            disp('2 = LEDs')
            disp('3 = Mixed')
            disp('4 = Darkfield')
            disp('5 = Greyscale w highlights')
            disp('6 = HSV Normal image')
            disp('arrows  = +/- theta or brighten')
            disp('a / z = +/- weight')

        case 113
             close all
             break
        case 28  % pressed left arrow
            if theta > 1
                theta = theta-1;
            else
                theta = num_leds; % wrap around
            end
        case 29   % pressed right arrow
            if theta < num_leds
                theta = theta+1;
            else
                theta = 1;        % wrap around
            end
        case 115   % pressed s key (increase brightness)
            fprintf('b+ ');
            if brighten < 5
                brighten = brighten+.5;
            end
        case 120   % pressed x key (reduce brightness)
            fprintf('b+ ');
            if brighten >.5
                brighten = brighten-.5;
            end
        case 97    % pressed a key (increase weight)
             fprintf('w+ ');
            if weight < .99
                weight = weight+.1;
            end
        case 122   % pressed z key (increase weight)
             fprintf('w- ');
            if weight > .1
                weight = weight-.1;
            end
        case 111   % pressed o key
            % display image of opposite side LED
            % remember theta = 2 -> num_led+1 (2 ->13)
            theta = mod(num_leds/2+theta,num_leds);
            if theta==0
               theta=num_leds;
            end
        case 112   % pressed p key (display sum of two opposite image)
            if paired == 0
                paired = 1;
                fprintf('Paired on ');
            else
                paired = 0;
                fprintf('Paired off ');
            end
        case 49   % pressed 1 key (display Brightfield image)
            disp(' ')
            disp('1: Brightfield')
            display_mode = 1;
        case 50   % pressed 2 key (display RTI single images)
            disp(' ')
            disp('2: LEDs')
            display_mode = 2;
        case 51   % pressed 3 key (display combined images)
             disp(' ')
             disp('3: Mixed')
             display_mode = 3;
        case 52   % pressed 4 key (display Darkfield image)
             disp(' ')
             disp('4: Darkfield')
             display_mode = 4;
        case 53   % pressed 5 key (display gray_combined images)
            disp(' ')
            disp('5: Greyscale w highlights')
            display_mode = 5;
        case 54   % pressed 6 key (display HSV image)
            disp(' ')
            disp('6: HSV Normal images')
            display_mode = 6;
        case 55   % pressed 7 key (User draw a profile plane line)
            disp(' ')
            disp('7: HSV IMPROFILE')
            display_mode = 7;
        case 116
            % pressed t key (increase threshold)
            if threshold < 1
                threshold = threshold+0.02;
            end
            disp(['threshold = ', num2str(threshold)]);
        case 104
            % pressed h key (increase threshold)
            if threshold >0.001
                threshold = round(threshold-0.02,2);
            end
            disp(['threshold = ', num2str(threshold)]);

    end

 %-------------------------------------------------------------------------
      add_image = squeeze(image_stack(:,:,theta+2,:));

      theta_text = ['\theta = ',num2str(theta)];

      if paired
        theta_pair = mod(num_leds/2+theta,num_leds);
```

```
        if theta_pair==0
           theta_pair=num_leds;
        end
        pair_image = squeeze(image_stack(:,:,theta_pair+2,:));
        add_image = add_image + pair_image;
        theta_text = ['\theta = ',num2str(theta),' & ',num2str(theta_pair)];
      end
%-----------------------------------------------------------------------------
switch display_mode
        case 1
            % just brightfield
            display_image = bright_image;
        case 2
            % just LED image
            display_image = add_image*brighten;
        case 3
            % mixed
            display_image =  weight * add_image*brighten + (1-
weight)*bright_image;
        case 4
            % Darkfield
            display_image =  dark_image;
        case 5
            % Grayscale bright
            grey_image = rgb2gray(bright_image);
            display_image =  weight * add_image*brighten + (1-
weight)*grey_image;
        case 6
            sat_image = max_image;
            val_image = rgb2gray(bright_image);
            % make flat areas grayscale
            %mask = del_image > threshold;
            mask = imbinarize(darkgray_image,threshold);
            sat_image(mask) = ones_image(mask);
            val_image(mask) = ones_image(mask);

            HSV_norm_image =cat(3, max_theta_hsv, sat_image, val_image);
            RGB_norm_image= hsv2rgb(HSV_norm_image);
            limits=stretchlim(RGB_norm_image);
            RGB_norm_image = imadjust(RGB_norm_image,limits);
            imshow(RGB_norm_image)

            hold on
            %display the HSV color wheel
            HSV_image=imread('HSV wheel.png');
            imshow(HSV_image)
            hold off
            colormap hsv
      case 7
            sat_image = max_image;
            val_image = rgb2gray(bright_image);
            % make flat areas grayscale
                mask = imbinarize(darkgray_image,threshold);
                sat_image(mask) = ones_image(mask);
                val_image(mask) = ones_image(mask);
        HSV_norm_image =cat(3, max_theta_hsv, sat_image, val_image);
        RGB_norm_image= hsv2rgb(HSV_norm_image);
        limits=stretchlim(RGB_norm_image);
        RGB_norm_image = imadjust(RGB_norm_image,limits);

        imshow(RGB_norm_image)
        hold on
        imshow(HSV_image)
        hold off
        colormap hsv
        title(['HSV PROFILE: Select a Line'],'Color',[1 0
                0],'Fontsize',14)

        max_theta_degs = max_theta_hsv*360+0.1;
        mask_inv = imcomplement(mask);
        max_theta_degs(mask_inv) = zeros_image(mask_inv);

        [~,~,~,x_vals,y_vals]=improfile;
        x_del = x_vals(2)-x_vals(1);
        y_del = y_vals(2)-y_vals(1);
        profile_ang = -180/pi()* atan2(y_del,x_del);

        profile_ang_image = ones_image*profile_ang;
        height=-cosd(max_theta_degs-profile_ang_image);
        height(mask_inv) = zeros_image(mask_inv);

        % smooth
        height= imgaussfilt(height,2);

        slope_xy = improfile(height,x_vals,y_vals);
        profile_xy= slope_xy;
        profile_len = size(profile_xy,1);

        for i=2:profile_len
            profile_xy(i)=profile_xy(i)+ profile_xy(i-1);
        end

        fig_profile =
        figure('Name','Profile','units','inches','Position',[9 1 10 2]);

        xx =linspace(1,profile_len,profile_len);
        plot(xx,profile_xy,'k','Linewidth',1)
        %ylim([0,255])
        xlim([0,profile_len])
        title(['Height Profile,Press any key to continue...'],'Color',[1
               0 0],'Fontsize',14)
        pause
        close(fig_profile)
        title(['HSV PROFILE: Single Click select another
               line...'],'Color',[0 .7 0],'Fontsize',14)
end
```

```
%--------------------------------------------------------------------------
% lighting indicator section

     switch display_mode
        case {2,3,5}
            L_origin = 80;
            L_fov = 70;
            L_radius = 60;
            L_size = 7;

            theta_l = -1*(theta-1)*360/num_leds;
            phi_l = 90;

            L_x = L_origin+L_radius*cos(theta_l/57.3)*sin(phi_l/57.3);
            L_y = L_origin+L_radius*sin(theta_l/57.3)*sin(phi_l/57.3);

            display_image= insertShape(display_image,'FilledRectangle',...
                [L_origin-L_size*1.3 L_origin-L_size L_size*2*1.3
L_size*2],...
                'Linewidth',2,'Color','blue','Opacity',1);

     end
%--------------------------------------------------------------------------
% title section

    switch display_mode
     case 1
        title(['BRIGHT FIELD'],'Color',[.5 .5 .5],'Fontsize',14)
     case 2
        title(['LED Images:  ',theta_text,...
               ',  brighten = ',num2str(round(brighten,1))]...
               ,'Color',[0 .5 0],'Fontsize',14)
     case 3
        title(['MIXED:  ',theta_text,...
              ',  brighten = ',num2str(round(brighten,1)),...
              ',  weight = ',num2str(round(weight,1))],...
              'Color',[.5 0 0],'Fontsize',14)
     case 4
        title(['DARK FIELD'],'Color',[0 0 0],'Fontsize',14)
     case 5
        title(['Grayscale w Highlights: ',theta_text,...
              ',  brighten = ',num2str(round(brighten,1)),...
              ',  weight = ',num2str(round(weight,1))],...
              'Color',[.5 0 0],'Fontsize',14)
     case 6
        title(['HSV NORMALS: color \rightarrow \theta'],...
        'Color',[.5 0 0],'Fontsize',14)
    end

end
 close all
 disp(' ')
 disp('done')
 disp(' ')
```

References

[1] R. Wayne, Light and Video Microscopy, Third ed., Elsevier, 2019. Available: 10.1016/C2017-0-01719-6.

[2] M. Paúr, B. Stoklasa, Z. Hradil, L. L. Sánchez-Soto and J. Rehacek, "Achieving the Ultimate Optical Resolution," *Optica,* vol. 3, no. 10, pp. 1144-1147, 2016. Available: 10.1364/OPTICA.3.001144.

[3] S. Ram, P. Prabhat, J. Chao, R. J. Ober and E. S. Ward, "Resolution beyond Rayleigh's criterion: A modern resolution measure with applications to single molecule imaging," *2007 IEEE Dallas Engineering in Medicine and Biology Workshop,* pp. 110-113, 2007. Available: 10.1109/EMBSW.2007.4454186

[4] Y. Leng, Materials Characterization: Introduction to Microscopic and Spectroscopic Methods, Second ed., Wiley-VCH, 2013. Available: 10.1002/9783527670772.

[5] B. F. S. Altshuler and T. Mannack, "Shedding New Light on Ancient Objects," *Arion: A Journal of Humanities and the Classics,* vol. 22, pp. 53-74. Available: 10.2307/arion.22.1.0053.

[6] J. Lee, *Analysis and Optimization of MEMS Out-of-plane Thermal Actuators,* Dalhousie University, 2003. Available: 10.1088/0960-1317/16/2/003.

[7] D. McMullan, "Scanning Electron Microscopy," *51st Annual Meeting of the Microscopy Society of America,* vol. 17, no. 3, p. 175–185, 1995. Available: 10.1002/sca.4950170309.

[8] E. M. Payne, "Imaging Techniques in Conservation," *Journal of Conservation and Museum Studies,* vol. 10, no. 2, pp. 17-29, 2013. Available: 10.5334/jcms.1021201.

[9] C.-L. HUYNH, "How to improve the laser scanning of shiny surfaces which can give unwanted reflections," Aalborg University: Department of Electronic System, 2010.

[10] T. Schenk, "Introduction to photogrammetry," 2005.

[11] M. Pavlovskis, M. Chizhova, J. Hindmarch and M. Hess, "Application of multi-criteria decision making for the selection of sensing tools for historical gravestones," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* vol. 43, pp. 1435-1442, 2020. Available: 10.5194/isprs-archives-XLIII-B2-2020-1435-2020.

[12] R. Epstein, A. L. Yuille and P. N. Belhumeur, "Learning object representations from lighting variations," in *Object Representation in Computer Vision*, Berlin, 1996. Available: 10.1007/3-540-61750-7_29.

[13] T. Malzbender, D. Gelb and H. J. Wolters, "Polynomial Texture Maps," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001. Available: 10.1145/383259.383320.

[14] H. Mytum and J. R. Peterson, "The Application of Reflectance Transformation Imaging (RTI) in Historical Archaeology," *Historical Archaeology,* vol. 52, pp. 489-503, 2018. Available: 10.1007/s41636-018-0107-x.

[15] V. Kitanovski and J. Y. Hardeberg, "Objective evaluation of relighting models on translucent materials from multispectral RTI images," *IS&T International Symposium on Electronic Imaging Science and Technology,* pp. 133-1-133-8, 2021. Available: 10.2352/ISSN.2470-1173.2021.5.MAAP-133.

[16] S. M. Duffy, "Multi-light Imaging for Heritage Applications," 2013.

[17] V. Corregidor, R. Dias, N. Catarino, C. Cruz, L. C. Alves and J. Cruz, "Arduino-controlled Reflectance Transformation Imaging to the study of cultural heritage objects," *SN Applied Sciences,* vol. 2, no. 9, pp. 1-10, 2020. Available: 10.1007/s42452-020-03343-4.

[18] G. Earl, P. Basford, A. Bischoff, A. Bowman, C. Crowther, J. Dahl, M. Hodgson, L. Isaksen, E. Kotoula, K. Martinez, H. Pagi and K. E. Piquette, "Reflectance transformation imaging systems for ancient documentary artefacts," *Electronic Visualisation and the Arts (EVA 2011),* pp. 147-154, 2011. Avaiable: 10.14236/ewic/EVA2011.27.

[19] Y. Goldman, R. Linn, O. Shamir and M. Weinstein-Evron, "Micro-RTI as a novel technology for the investigation and documentation of archaeological textiles," *Journal of Archaeological Science: Reports,* vol. 19, pp. 1-10, 2018. Available: 10.1016/j.jasrep.2018.02.013.

[20] D.-Ø. E. Solem and E. Nau, *Two New Ways of Documenting Miniature Incisions Using a Combination of Image-Based Modelling and Reflectance Transformation Imaging,* vol. 12, Multidisciplinary Digital Publishing Institute, 2020, p. 1626. Available: 10.3390/rs12101626.

[21] G. Pitard, G. Le Goïc, A. Mansouri, H. Favrelière, S.-F. Desage, S. Samper and M. Pillet, "Discrete Modal Decomposition: a new approach for the reflectance modeling and rendering of real surfaces," *Machine Vision and Applications,* vol. 28, no. 5, p. 607–621, 2017. Available: 10.1007/s00138-017-0856-0.

[22] H. Coules, P. Orrock and C. Er Seow, "Reflectance Transformation Imaging as a tool for engineering failure analysis," *Engineering Failure Analysis,* vol. 105, pp. 1006-1017, 2019. Available: 10.1016/j.engfailanal.2019.07.037.

[23] J. Miles, M. Pitts, H. Pagi and G. Earl, "New applications of photogrammetry and reflectance transformation imaging to an Easter Island statue," *Antiquity,* vol. 88, no. 340, pp. 596-605, 2014. Available: 10.1017/S0003598X00101206.

[24] J. Min, E. Yoo, H. Choi, S. Ahn, J. Ahn and S. Ahn, "Interpretation through Digital Imaging: Reflectance Transformation Imaging (RTI) as a Tool for Understanding Paintings," *International Journal of Contents,* vol. 12, no. 6, pp. 41-50, 2020. Available: 10.5392/IJoC.2020.16.2.041.

[25] J. Lemesle, F. Robache, G. Le Goic, A. Mansouri, C. A. Brown and M. Bigerelle, "Surface reflectance: An optical method for multiscale curvature characterization of wear on ceramic-metal composites," *Materials,* vol. 13, no. 5, p. 1024, 2020. Available: 10.3390/ma13051024.

[26] A. K. R. Choudhury, "Object appearance and colour," in *Principles of Colour and Appearance Measurement*, Elsevier, 2014, pp. 53-102. Available: 10.1533/9780857099242.53.

[27] B. v. Ginneken, M. Stavridi and J. J. Koenderink, "Diffuse and Specular Reflectance from Rough Surfaces," *Applied Optics,* vol. 37, no. 1, pp. 130-139, 1998. Available: 10.1364/AO.37.000130.

[28] T. Kurihara and Y. Takaki, "Shading of a computer-generated hologram by zone plate modulation," *Optics express,* vol. 20, no. 4, pp. 3529-3540, 2012. Available: 10.1364/OE.20.003529.

[29] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM,* vol. 18, no. 6, pp. 311-317, 1975. Available: 10.1145/360825.360839.

[30] T. G. Dulecha, F. A. Fanni, F. Ponchio, F. Pellacini and A. Giachetti, "Neural reflectance transformation imaging," *The Visual Computer,* vol. 36, no. 10, p. 2161–2174, 2020. Available: 10.1007/s00371-020-01910-9.

[31] M. A. Volkova, S. V. Zlatina, S. N. Natarovskii, O. N. Nemkova, T. F. Selezneva, N. B. Skobeleva, D. N. Frolov, L. M. Kogan and B. P. Papchenko, "Prospects of using LEDs in the illuminating systems of microscopes," *Journal of optical technology,* vol. 72, no. 2, pp. 186-190. Available: 10.1364/JOT.72.000186.

[32] K. Jack, "Video and Image processing," in *Communications Engineering Desk Reference*, Elsevier, 2009, p. 568.

[33] P. Siffert and E. Krimmel, Silicon: Evolution and Future of a Technology, Heidelberg: Springer, 2004. Available: 10.1007/978-3-662-09897-4.

[34] R. C. Anderson, R. S. Muller and C. W. Tobias, "Investigations of the electrical properties of porous silicon," *Journal of the Electrochemical Society,* vol. 138, no. 11, p. 3406, 1991. Available: 10.1149/1.2085423.

[35] J. G. Korvink and O. Paul, MEMS: A practical guide of design, analysis, and applications, Freiburg: Springer Science & Business Media, 2006. Available: 10.1007/978-3-540-33655-6.

[36] H. Nathanson, W. Newell, R. Wickstrom and J. Davis, "The resonant gate transistor," *IEEE Transactions on Electron Devices,* vol. 14, no. 3, pp. 117-133, 1967. Available: 10.1109/T-ED.1967.15912.

[37] A. Cowen, B. Hardy, R. Mahadevan and S. Wilcenski, "PolyMUMPs design handbook," *MEMSCAP Inc. 13,* 2011.

[38] B. Barazani, S. Warnat, A. J. MacIntosh and T. Hubbard, "MEMS measurements of single cell stiffness decay due to cyclic mechanical loading," *Biomedical Microdevices,* vol. 19, pp. 1-11, 2017. Available: 10.1007/s10544-017-0219-7.

[39] Y. Laghla and E. Scheid, "Optical study of undoped, B or P-doped polysilicon," *Thin Solid Films,* vol. 306, pp. 67-73, 1997. Available: 10.1016/S0040-6090(97)00247-2.