Characterizing the Evolution of Contact Area in Frictional Interfaces

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Citable link</td>
<td><a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:41940967">http://nrs.harvard.edu/urn-3:HUL.InstRepos:41940967</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:dashboard.current.terms-of-use#LAA">http://nrs.harvard.edu/urn-3:HUL.InstRepos:dashboard.current.terms-of-use#LAA</a></td>
</tr>
</tbody>
</table>
Characterizing the Evolution of Contact Area in Frictional Interfaces

A senior design project submitted in partial fulfillment of the requirements for the degree of Bachelor of Science at Harvard University

Thomas Pilvelait

S.B. Degree Candidate in Mechanical Engineering
Faculty Advisor: Professor Shmuel Rubinstein
Harvard University School of Engineering and Applied Sciences
Harvard University
Cambridge, MA
April 5, 2019
This report is dedicated to my loving parents whose support in all of my academic endeavors made this project possible.
Acknowledgements

I would firstly like to thank the wonderful support and guidance that I have received from the SMR Lab; working in the SMR Lab has simply been the most intellectually fulfilling activity I have been involved with while as an undergraduate at Harvard. All of the SMR team, but especially Professor Rubinstein and PhD candidate Sam Dillavou, have contributed enormously to my growth as an engineer, as a scientist, and as a person over the past year. With them, I truly stand on the shoulders of giants. Thank you to Michelle Rosen and Mike Litchfield for providing useful insight and suggestions for project directions. I also thank the ES100 and Active Learning Labs support staff, and in particular Steve Cortesa, Maddie Hickman, and Elaine Kristant as well as James Weaver from the Wyss Institute for their support.
# Table of Contents

Acknowledgements ................................................. ii

List of Figures ................................................... vi

List of Tables ..................................................... viii

1 Introduction and Motivation .................................. 1

2 Background Research ............................................ 2
   2.1 Disadvantages of a Shear-Based System .................... 4
   2.2 An Optical-Mechanical Configuration ....................... 4
   2.3 Project Influences and Potential Future Uses ............. 6

3 Design Goals ..................................................... 7
   3.1 Apparatus Description and Schematic ....................... 7
   3.2 Design Metrics ............................................. 9
   3.3 Rationale for Design Criteria ............................... 9
      3.3.1 Design and Creation of Sample Molds.................. 9
      3.3.2 Optics ................................................ 11
      3.3.3 Load-cell ............................................. 11
      3.3.4 Motor stage and mechanical supports ................. 12
      3.3.5 Motor Control ....................................... 13
      3.3.6 Real-time Processing ................................. 13
      3.3.7 Post-processing ..................................... 14

4 Design Approach ............................................... 14
   4.1 Sample Generation ......................................... 14
   4.2 Optics ..................................................... 15
A  Bill of Materials

B  Engineering Drawings
   B.1  Photo of Apparatus .................................................. 42
   B.2  Thor Labs Long Travel Stage ...................................... 42
   B.3  Thor Labs Long Travel Stage ...................................... 44

C  Specification Sheets of Major Components
   C.1  Thor Labs Long Travel Stage ...................................... 45
   C.2  Thor Labs CMOS Sensor ............................................. 47
   C.3  Canon Macro Lens .................................................... 49
   C.4  Futek Load Cell ...................................................... 50
   C.5  Dragon Skin Silicone Rubber .................................. 52

D  MATLAB Scripts
   D.1  Feedback Protocols .................................................. 53
      D.1.1  Determine Area Function ..................................... 53
      D.1.2  GoToForce Function .......................................... 53
      D.1.3  GoToArea Function ........................................... 55
      D.1.4  MeasureForce Function ..................................... 56
   D.2  Surface Generation .................................................. 56
      D.2.1  Generate Random Surface Function ......................... 56
      D.2.2  Generate Contact Height Distribution Function .......... 61
      D.2.3  Transmute Surface into Solid Function ................... 63
      D.2.4  Transmute Solid into STL File Function .................. 67
   D.3  Miscellaneous Scripts ............................................. 70
      D.3.1  Script used to calculate image drift ....................... 70
List of Figures

2.1 Shear loading apparatus ......................................................... 3
2.2 Example of a tribometer ....................................................... 3
2.3 Diagram of microscopic interface ............................................. 5
2.4 Example plots of aging and loading data ................................... 6
3.1 Diagram of the apparatus ...................................................... 7
3.2 Diagram of total internal reflection ........................................... 9
3.3 Images of a power spectrum generated surface and its 3D printed version .................................. 10
4.1 Examples of 3D printed molds with Form2 and Stratasys printers .......... 15
5.1 A PDMS sample casted from a 3D printed mold ........................... 19
5.2 A silicone rubber sample casted from a 3D printed mold ................ 19
5.3 A plot demonstrating the optimal values of $K_I$ and $K_D$ ................. 20
5.4 A plot demonstrating the optimal value of $K_P$ ............................ 21
6.1 Examples of Raw and Binarized Images .................................... 24
6.2 Contour plots of the contact area for aging and loading .................. 25
6.3 Intensity plots for a force calibration piece ................................ 26
6.4 Plot of force versus time for motor control feedback ...................... 27
6.5 Plot of step responses for motor control ................................... 28
6.6 Plot of motor movements over a 5 minute experiment .................... 28
6.7 Plot of motor movement response times ................................... 29
6.8 Experimental data for aging and loading with a PDMS soda-lime glass interface .................................. 30
6.9 Plot of constant area using real time area feedback ....................... 31
6.10 Image depicting asperity tracking scheme .................................. 31
6.11 Plot of a decomposition of a surface into its constituent asperities ....... 32
6.12 Plot of absolute difference for subpixel shifts ............................. 33
7.1 2D Plot demonstrating differences between aging and loading ............ 35
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.2   Plot of absolute difference between aging and loading with the constant contact area protocol</td>
<td>36</td>
</tr>
<tr>
<td>B.1   Photo of apparatus</td>
<td>42</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Project Metrics</td>
<td>9</td>
</tr>
<tr>
<td>4.1</td>
<td>Camera specifications</td>
<td>16</td>
</tr>
<tr>
<td>7.1</td>
<td>Project Metrics</td>
<td>34</td>
</tr>
<tr>
<td>A.1</td>
<td>Bill of Materials</td>
<td>41</td>
</tr>
</tbody>
</table>
Abstract

Seemingly static interfaces between two solids are in fact dynamic. These multicontact interfaces (MCIs) exert enormous stresses on tiny contact areas, causing time-dependent deformation. Though these phenomena are ubiquitous (e.g. from micromachines to plate tectonics), their underlying mechanisms are not yet well understood. By constructing a novel apparatus, we may now take real-time, 2D measurements of the real contact area in an interface over a variety of normal loads and time steps. Given that the real contact area of an interface may be used as a proxy for frictional strength, this area based tribometer allows for an investigation into the underlying dynamics of frictional interfaces. Furthermore, by varying geometric and elastic properties of samples, we induce variations in the 2D structure of contact growth, which give insight into the mechanisms that produce deformations in frictional interfaces.
1 Introduction and Motivation

Though friction exists ubiquitously in mechanical systems—from micro-machines to tectonic plates—there still exist several gaps in the understanding of how frictional interfaces evolve within different environments, such as over different pressures, temperatures, and time scales. The deformation of a frictional interface over time—also referred to as “aging” or “healing”—is of particular interest as the temporal evolution of frictional interfaces remains understudied when compared to other causes of deformation, specifically shearing and loading. Moreover, letting an interface age produces a surprising result: the frictional strength of the interface increases logarithmically in time [1,2]. To illustrate how strange this result is, if a book is set down on a table, it will become logarithmically more difficult over time to push such a book. The most commonly proposed hypotheses involve adhesion forces [3] which increase the bonding energy of the surfaces logarithmically [4]. Notwithstanding the evident similarity in these logarithmic results, there does not currently exist a model which models these dynamics at a mesoscopic, (in contrast to microscopic or macroscopic level).

To obtain results at a mesoscopic level, one may extract the frictional strength of an interface from a combination of its material properties and contact area measurements due to the fact that an interface’s real area of contact correlates well with its frictional properties [1]. Furthermore, aging and loading an interface are often thought to be geometrically equivalent. Preliminary testing in the SMR Lab, however, indicates that temporal and load-based deformations may be fundamentally different phenomena. To properly study these phenomena with conventional methods, one would require an enormous of data. Tribometers, for example, merely yield one data point per experiment simply the ratio of the shear and load forces, as per the familiar Anton-Coulomb friction law:

\[ F_f = \mu \times F_N. \]  

(1)

Thus the ultimate goal of this project is to design, build, calibrate, and test an apparatus which is able to characterize the evolution of frictional interfaces in 2D and in real-time. This apparatus would chiefly help those who currently study relaxation in disordered systems. The Rubinstein Lab, for example, often studies relaxation friction-based systems, and so this project could be of great use to such a lab in understanding how and why relaxation occurs. In fact, a
recent paper by Harvard PhD candidate Sam Dillavou and Professor Rubinstein demonstrated that aging and de-aging an interface often produce memory effects [2]. These effects, however, have not been thoroughly investigated two-dimensionally as they might be in this new apparatus. Moreover, these results may fit in with a recently proposed theoretical model on relaxation [5]. Such a model incorporates a wide variety of topics including crumpled paper [6], the resistivity of conductors [7], and the thermal expansion of polymers [8] (all as cited in Y. Lahini et. al. Phys. Rev. Let. 2017 [6]). As a result, one may see that the implications of this apparatus would be quite widespread in the tribological research community.

Though this project will most immediately benefit the SMR Lab, there are numerous applications to industry. Any field that could benefit from a refined apparatus and methodology of studying contact mechanics would be a target for this product; brake pads, shoes, and tires are all products which heavily rely on the wearing of rubber (highly similar to this project’s main material, a silicone rubber). In fact, the SMR Lab has for some time been working with the chemical company BASF to understand why polyurethane shoes wear down more quickly than rubber ones. This of course is an entirely distinct project, but there may be insights to be gained from the project described in this report. Moreover, this apparatus could provide insights into the underlying mechanisms which govern all frictional systems, and could therefore be applied to systems as diverse as earthquake modeling, granular materials, and machine bearings.

2 Background Research
Figure 2.1: Shear loading apparatus (a) A simplified schematic of a shear loading apparatus. (b) Both the normal and shear forces are recorded and used to create plots of the frictional coefficient (adapted from Dieterich and Kilgore, US Geological Survey, 1994 [9]).

Figure 2.2: (a) Simple diagram of a pin-on-disk tribometer (b) Surface plot of a worn surface obtained with an AFM. [10].
2.1 Disadvantages of a Shear-Based System

In many tribological experiments, simply recording the shear (i.e. frictional) force and the normal force between the two surfaces is sufficient for characterizing frictional interfaces. However, wearing down surfaces is not always desired for two reasons. First, this obviously hinders re-using the same sample to collect similar data for multiple trials, as over time the samples are ground down. Second, the apparatus normally responsible for shearing the surface—shown in Figure 2.2—has finite stiffness. Although this may be modeled well [11], its effects still limit the resolution of the data by, for example, limiting the velocity step-response [9] (i.e. how quickly the shear force is applied). This effect also exists in the optical-mechanical setup (described in Section 3.2), but only in one dimension (vertical loading) as opposed to two dimensions in shear setups (loading and shearing). Furthermore, since the real area of contact of an interface may easily be converted to its frictional strength—something that may easily be measured with a camera as explained in Section 3.2—the optical-mechanical arrangement doubly has an advantage over a purely mechanical system.

Let us take, for example, a material which is analyzed via a pin-on-disk tribometer and then later characterized by means of an Atom Force Microscope, as depicted in Figure 2.2. Such a surface analysis is quite detailed and provides useful information about how such a material wears, but only after the surface has been worn, and not during the wearing. The only information available during the wearing are primarily displacements and forces, which then means other potentially useful data about regarding dynamics which exist during the material wearing may remain unseen.

2.2 An Optical-Mechanical Configuration

As presented in the seminal work *The Frictional Lubrication of Solids* [12], we may instead infer frictional properties by determining the real contact area. This real contact area of a surface is typically much smaller than the nominal contact area as the interface consists of many discrete contact points as shown in Figure 2.3. The real contact area, therefore, is the sum of these individual contacts’ area. In the totally plastic case—a good approximation for metals, as an example—the classic model regarding plastic deformation holds. If the deformation is plastic (a good assumption for most metals), the contact points thus have a constant pressure: the shear strength of the
Figure 2.3: A microscopic view of an interface. Though most frictional interfaces may seem macroscopically smooth, they are in fact microscopically rough due to the presence of asperities. These asperities are in contact with one another, and their pushing on one another results in the macroscopic frictional strength.

interface, $\sigma_S$. Thus the frictional force experienced by the interface is given by

$$F = A_R \ast \sigma_S,$$

where $A_R$ is the real contact area [12]. One might expect this result to be invalid for elastic or elastoplastic deformation, given that such deformations allow for contacts to return to their original state, rather than becoming linearly and irreversibly deformed; however, even in the event that the deformation is totally elastic or even elastoplastic, the above formula still holds and is thus valid for a variety of materials [13, 14].

This real contact area may be found via optical measurement by determining the intensity of light which passes through an interface [15]. When light is shone on a contact plane, only the points of true contact will allow light to pass through. This schematic will be further discussed in Section 3.2. As an example of a similar configuration, one may also place a flat, transparent medium in contact with a rough, opaque medium, and measure the intensity of light which reflects off of the contact points. This intensity of light then relates to the real contact area which, to reiterate, may be used as a proxy for the frictional strength of the interface. Even if the transparent medium is not entirely flat, if its roughness is on a smaller scale than that of the opaque medium, then its roughness is essentially negligible and therefore may be approximated as being essentially a planar surface.
2.3 Project Influences and Potential Future Uses

In a seminal paper regarding the aging of frictional interfaces [9], the researchers were able to provide some basic 2D images of similar systems under loading and aging, but overall the data points were too few to draw any geometric conclusions. The researchers used an optical mechanical configuration to scatter light off of contacts in quartz samples, providing contour plots of the contact area as both functions of stress and time. This data was insufficient insofar as it was 1) collected for different samples for aging and loading and 2) only four data points for loading and three data points for aging were obtained. A primary goal of the research component of this project—i.e. the experimental apparatus’ primary future use— is to expand upon and potentially draw conclusions from the results in this original paper by Dieterich and Kilgore. See Figure 2.4 for an example of such results. If one were able to definitively argue not only the fact that aging and loading cause distinct deformations in frictional interfaces, but also how these deformations arise, this result would have significant implications. Most importantly, many contemporary models assume that any equivalent total contact area is the same, without considering the nuances of the interface which arise to differences between aging and loading [16–18].

Figure 2.4: Example of aging and loading data (a) Plot of contact area versus normal stress and (b) Plot of contact area versus time. These are the only data points given, and appear to be from different surfaces which bars any immediate conclusions regarding the differences between these two types of deformation (adapted from Dieterich and Kilgore, US Geological Survey, 1994 [9].
3 Design Goals

This project consists of several components, all of which in aggregate produce an apparatus capable of real-time contact area analysis, followed by post-processing method which allow for the decomposition of the surface into individual asperities. More specifically, the project consisted of the following components:

3.1 Apparatus Description and Schematic

- Sample Generation
  1. Design and creation of 3D printed molds

- Physical Instrument

Figure 3.1: A diagram of the apparatus, as described in the text.
1. Optics
2. Load-Cell
3. Motor stage and mechanical supports

- Software
  1. Motor control feedback
  2. Real time area feedback

- Post-Processing Methods
  1. Asperity Decomposition

Given that a version of this setup had been previously constructed, the load-cell as well as the motor stage and its mechanical supports had previously been installed, and are described here; however the remainder of the design specifications will focus on the components which were improved upon or constructed throughout the course of this project, rather than previously constructed components.

Together, these components produce the apparatus as depicted in Figure 3.1. The process of applying a load to a sample is as follows. From an experimental protocol, a predetermined force is converted into a displacement which is then sent to the motor. This change in displacement induces a change in load, measured by the load-cell. This load is transmitted to the sample, which is contact with a totally internally reflected glass plate. In other words, this light reflects at an angle at less than the critical angle, via Snell’s Law:

\[ n_1 \sin \theta_1 = n_2 \sin \theta_2 \implies \theta_{Cr} = \arcsin \frac{n_1}{n_2}. \]  

The close arrangement of LEDs on the glass plate along with the use of tinfoil as a secondary reflector reduces the amount of scatter to be minimal. Therefore, when the LED light comes into contact with asperities, it is reflected into the lens of the camera, which then obtains an image of the contact area of the interface. A diagram depicting the total internal reflection is shown in Figure 3.2.
Figure 3.2: LED light is internally reflected inside of a glass plate and thus light beams are scattered from the true points of contact to the camera lens. This allows for determining the total contact area.

3.2 Design Metrics

The overall design metrics as well as the achieved metrics may be found in Table 3.1, and explanations for the criteria selections may be found in Section 3.3.

<table>
<thead>
<tr>
<th>Component</th>
<th>Target Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Generation</td>
<td>Mean feature size $\geq 10$ px</td>
</tr>
<tr>
<td>Optics</td>
<td>Resolves $\leq 50\mu m$</td>
</tr>
<tr>
<td>Motor Control</td>
<td>Loading $\leq 5.0$s; Variation $\leq 10%$</td>
</tr>
<tr>
<td>Real-Time Processing</td>
<td>Variation in constant area $\leq 1%$</td>
</tr>
<tr>
<td>Post-processing</td>
<td>0 mismatches in feature tracking</td>
</tr>
</tbody>
</table>

Table 3.1: The metrics which the project is measured, along with the realized specifications.

3.3 Rationale for Design Criteria

3.3.1 Design and Creation of Sample Molds

In order for the samples to mimic natural surfaces, they must have contact heights which follow a normal distribution. Contact heights of natural surfaces are randomly generated by surface formation (e.g. magma cooling and forming rocks, smelting metal, etc.), breaking, and microscopic forces such as adhesion. Thus by the Central Limit Theorem these contact heights must follow a Gaussian, and so too must these samples. But the size and number of these contacts could be many
values, and what’s more is the optimal arrangement of these contacts is not known.

Because differences between aging and loading deformations form the crux of this project, it is desirable to construct varying sample geometries so as to provide a wide range of data points. The first step of the sample generation procedure is to generate an STL file of a mold, which is done via a power spectrum analysis script in MATLAB. Power spectrum analyses are useful for characterizing surface topography and decomposing a surface into different spatial frequencies. The surface, then, is simply a Fourier series, consisting of various sine and cosine waves. These samples are therefore created in reverse of what a power spectrum analysis normally is used for: there exists some surface geometry that we wish to study, and so the power spectrum analysis then generates such a surface. A sample surface, for example, might need a particular root-mean-square roughness (i.e. RMS contact height), or perhaps a specific correlation between features (e.g. “mountains” and “valleys”). These samples, moreover, require a Gaussian distribution of heights since they, at a mesoscopic level, are mimicking real surfaces at a microscopic level. This principle of self-similarity (i.e. that a surface should have the same geometric qualities regardless of scale), is widely accepted within the tribology community, and has on occasion been verified for several scales [19]. For an example of an STL generated via a power spectrum script, see Figure 6.8. For what this translates to after 3D printing, also see Figure 6.8.

Although it is useful to mimic real surfaces, these sample profiles really can be any design. In the near future, in fact, there are plans to characterize simpler sample profiles, such as uniformly

Figure 3.3: (a) An example of a power spectrum generated surface. (b) A 3D printed sample via the Form2 printer. This now forms the base of the PDMS sample molds.
distributed hemispheres or a single plane sine wave. This approach could yield useful information about more complicated structures seen in the power spectrum generated samples in use at the moment, and demonstrates the utility of the approach taken in this project regarding the way the samples are studied (that is, the method of total internal reflection, being able to vary the intensity of light, and so on).

In summary, the governing design metrics for these surfaces are 1) That they mimic real surfaces with a particular (roughly Gaussian) contact height distribution 2) That they are of a size resolvable with the optical setup, preferably with a standard DSLR camera lens and sensor for ease of use and 3) That they deform on a scale consistent with the force range of the motor and the optical setup.

3.3.2 Optics

**LED Lights** This consists of the aforementioned total internal reflection by means of shining blue LED light through a transparent medium (in this case, soda-lime glass). These lights are connected to a power supply, and so one can tune the intensity of the light produced by a contact point so that the detected signal is between a pixel value of 0 (no light) and 255 (saturated), ideally in the middle of this range.

**Camera and Sensor** The camera and sensor are intimately tied with the sample size as well as the size of the asperities, but more generally for the camera, it was agreed upon that the optical setup ought to be rather cheap (\(<\$1000\)) and easy to assemble should other versions need to be installed. The primary design criteria for the camera is that it may resolve below the mean asperity size in the samples.

3.3.3 Load-cell

This is another component which had previously been installed before work on this project formally began, and is therefore left out of the overall design metrics. In essence, the load-cell must be able to resolve the maximum load that can be applied by the motor stage (\(\approx100\text{N}\)), while being able to provide a precision of \(\pm 0.1 \text{ N}—0.1\%\) of the maximum applicable force.
3.3.4 Motor stage and mechanical supports

Motor Stage  The motor stage and the mechanical components had already been installed by the formal start-date of this project, and so again their design specifications are left for Appendices B and C. The motor stage was one already available in the lab—the Thor Labs 300mm Linear Translation Stage—and therefore the mechanical supports were to be installed around the stage’s specifications. As a brief note, however, this motor was originally chosen—much earlier than the inception of this project—for its precision, reliability, and ease of access via MATLAB. For the mechanical supports, these simply had to support a restoring force of up to 100 N (not including their own weight), as well as to provide a level surface for the glass plate and sample to be placed upon.

Springs and Carriage Attachment  Although in an ideal world the sample could simply be loaded with the load cell and motor feedback loop, the high stiffness of the mostly steel apparatus would require a motor resolution much finer than what is reasonable. As a quick calculation, a load of 5N (a reasonable step-size for this type of experiment) for an area of 10cm X 10cm results in a strain of

\[ \varepsilon = \frac{\sigma}{E} = \frac{F/A}{E} = 2.5 \times 10^{-9} \, m^0 \]  

assuming this steel has a modulus of elasticity of 200 GPa. The Thor Labs motor in use has a minimum resolution of 1 µm [20], which means the apparatus, which is mostly steel between the motor and the sample, would need to be on the order of \( L = \frac{\Delta L}{\varepsilon} \approx 400m \). Since this is nearly three orders of magnitude larger than the current experimental setup, comparatively weak aluminum springs must be used to reduce the effective spring constant. As a brief mechanical explanation for this choice, from a simple Newtonian analysis one may find that two springs placed in series have an effective spring constant of

\[ k_{eq} = \frac{k_1 * k_2}{k_1 + k_2}. \]  

If we consider the steel components of the apparatus to effectively be a spring, we find that by introducing weaker springs in series will reduce the overall effective spring constant.

The aluminum springs now in use are kept in place by short posts as seen in Figure 3.1, and therefore lack a proper connective mechanism to the rest of the apparatus. This results in minor
inconsistencies insofar as the springs may slightly translate or twist, resulting in an inconsistent load application and sample placement. However, after collecting data for multiple experiments for the same sample and orientation, it was determined that a though spring holder would improve the user interface of the apparatus, it was ultimately not essential for the core of the purpose of constructing the apparatus. In the future, securing the springs to the stage would allow for raising and lowering the sample by means of accessing the motor feedback via MATLAB, rather than physically manipulating and orienting the sample.

3.3.5 Motor Control

Due to the logarithmic nature of aging, the first second of data collection contains the same amount of information as the subsequent 10 seconds, that decade’s subsequent 100 seconds, and so on. Thus it is crucial for the apparatus to have as small of a response time as possible. Unfortunately, the current response time of the motor to load 50 Newtons—a typical load value— is on the order of 5 seconds.

Given the measured spring constant as 6.15 N/mm, the distance required to travel by the motor is \( x = \frac{F}{k} = 0.81 \text{mm} \). Thor Labs’ documentation has that the maximum acceleration of the motor is 20mm/s/s (the maximum possible velocity is large and thus does not factor in) [20]. Lastly, to optimize the travel time the motor should accelerate for half the distance: \( \frac{d}{2} = \frac{1}{2} \ast a \ast t^2 \rightarrow t \approx 0.4 \text{s} \). This is merely a kinematic picture of the setup and does not consider latency in the feedback loop (among other constraints with the motor), but clearly the response time of the motor may be decreased.

3.3.6 Real-time Processing

Given that the most immediate use of this apparatus will be used to distinguish between deformations caused by aging and loading, the crowning achievement of the project is to provide a real-time contact area analysis of the interface—such is the area-centric tribometer discussed in the introduction. More specifically, a design premise is to maintain a constant contact area as a function of time, regardless of the sample, motor, camera, etc. This feedback control is one which continually decreases the load to decrease the contact area, given that the contact area of an interface will always increase as a function of time (i.e., it will “age”). Therefore, the guiding design
principle for the real-time area feedback loop is that the variation in the constant area is small: on
the order of 1%.

3.3.7 Post-processing

The ability to decompose the surface into its constituent components is another hallmark of
this project, and this method allows for future analyses on how distinctions between aging and
loading arise. As a brief aside, this overall analysis is much more of an engineering sciences research
project than an engineering design project, and is therefore outside the scope of this report. But to
even provide the requisite data for this type of analysis, this asperity tracking method ought to be
able to clearly distinguish between individual asperities over the course of any type of experimental
protocol, whether it be strictly loading, strictly aging, or perhaps some combination of the two (as
is the case in the real-time area feedback control mentioned in the preceding section).

4 Design Approach

For each of the following components, there were more or fewer design considerations depending
upon the nature of the component. Software related components, for example, more so underwent
a continuous evolution whereas the physical components evolved in a much more discrete way.

4.1 Sample Generation

From the start of this project, the samples were desired to have custom geometries, which for
rapid generation requires 3D printing. The primary questions, then, were which 3D printers, resins,
and sample materials to use. The Active Learning Labs here at Harvard have resolutions between
25-100µm, and so these were chosen as a first step due to their ease of access. Unfortunately, the
quality of these molds was on the whole lackluster and were not faithful to the original STL file. As
shown in Figure 4.1, these initial molds were warped and had a rather poor resolution (the Form 2 is
advertised as having a 50 micron resolution, but these prints were not even accurate to within half a
millimeter in spots). As a result the next iteration of molds were generated via the Wyss Institute’s
Stratasys printer, which, as also shown in Figure 4.1, came out immaculately in comparison to the
Form 2 molds. For the remainder of the project, the Stratysys printer was utilized in lieu of the
FormLabs molds, and was provided cost-free for this project as well.
4.2 Optics

4.2.1 LEDs

LEDs were chosen as the source of illumination since the start of this project due to their low price point as well as their ease of use. The central question for the design choice, then, concerned the type of LEDs to be used. So as to mitigate scattering as much as possible, a lower wavelength strip of LEDs was desired. This commercially corresponds to blue LEDs, which Professor Rubinstein’s lab happened to already have.

4.2.2 Camera and Sensor

Given that the samples are limited by the resolution of the 3D printers (roughly 50 µm), a camera which could resolve less than 50µm was desired. Fortunately, this resolution aligns quite nicely with the capabilities of most name-brand DSLR camera lenses and sensors. As in the case of the LEDs, the SMR Lab already had in its possession several types of lenses and sensors, as described in Table 4.1.

Thus in the end I chose to use a Canon Macro lens, coupled with a Thor Labs CMOS sensor, which attaches to the lens by means of a c-mount. This combination allowed for imaging of the
4.3 Motor Control

Several schemes for implementing a feedback system for the motor-loadcell-MATLAB system. The first was a simple PID control which would incorporate the classical PID equation:

\[ u(t) = K_P * e(t) + K_I * \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt} \]  \hspace{1cm} (6)

The second scheme involved implementing a more advanced control scheme, such as Kalman Filtering. Such schemes were to explore if the PID controller proved insufficient, but in the end this PID controller was more than satisfactory and so no such methods were needed.

4.4 Real-time Processing

As previously mentioned, the ultimate objective of the real-time processing is to maintain the area as constant as a function of time. Simply put, the only possible method is for the apparatus to take an image, calculate the area, and then consequently adjust the area as a function of time. In other words, the heart of this problem lies in how the area is adjusted, given images and areas at some \( t_0 \) as well as at some later time \( t \). One possible method involves determining whether the differences are positive or negative, and then moving the motor by some displacement until the areas become equal. This was not even tested, inasmuch as clearly it would not only be slow, but would also possibly overshoot, introducing oscillations into the system. The underlying issue with this first process is that the relationship between displacement and area is not known, and really cannot be known. What may be known, however, is the relationship between contact area and

<table>
<thead>
<tr>
<th>Brand</th>
<th>In Focus for 50mm?</th>
<th>Full Field of View</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logitech C920 Webcam</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Canon 50mm Macro Lens</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Nikon DX 18-55mm</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>AF-S DX NIKKOR 18-55mm</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Canon EFS 55-250mm</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4.1: The feasibility of each of the cameras available in the SMR Lab.

entire field of view of the sample (chosen as 50 mm X 50 mm at a distance of 50 cm), while providing images in focus.
normal load. As mentioned in the Introduction, this relation is a linear function; in other words we have

\[ A(F_N) = b * F_N. \]  

(7)

The change in normal loads is then given by

\[ \Delta F = \frac{\Delta A}{b} \implies F_N(t) = \frac{A(t) - A(t = 0)}{b} + F_N(t = 0). \]  

(8)

Thus we have a transfer function which relates the change in force to the change in area and, by using the motor feedback, we may then translate this change in force to a change in displacement, which is of course something the motor stage understands as an input.

The difficulty, then, is determining \( b \). \( b \) is simply the best fit slope for the linear relationship between contact area versus normal load. So, one first calibrates the system by obtaining strictly loading data (i.e. as a separate experiment), and then the best fit slope may then be passed onto the constant area feedback protocol.

4.5 Post-processing

The objective of this component is to track each asperity over varying load or time steps. This script is rather similar conceptually to particle tracking, but it has the added difficulty of the fact that these asperities not only change in their total area and perimeter, but they also appear, disappear, merge, and split. Thus this problem is not just a problem of tracking \( N \) objects over the course of some experiment, but it rather concerns the process of identifying and matching objects that may exist in different states throughout the course of an experiment.

Fortunate that MATLAB is the software of choice for this project, this script utilized MATLAB’s excellent Image Processing Toolbox. In this toolbox, there exists a fairly simple yet effective algorithm for object detection. Essentially, the image is grouped into consecutive matrices of up to nine elements (this number is limited by the size of the image). If any one element is marked as “1” (for this works only in binarized images), then the two pixels are labeled as being from the same object. This process, however, is not consistent from image to image, as objects—asperities in this case—are constantly changing (splitting, merging, disappearing, etc.). Thus this asperity tracking script was designed to track asperities in a variety of ways, and therefore applicable to any
5 Design Evolution

5.1 Sample Generation

Although the first batch of molds came out nicely, there seemed to be a chemical affinity between the VeroBlue (the plastic used by the Stratasys) and Sylgard 184 PDMS. In the future this will be mitigated by curing the VeroBlue molds in a ventilated oven in the Wyss Institute at 60°C for 24 hours.

One last important aspect of sample preparation relates to the sample’s color. When cured, the Sylgard 184 PDMS is translucent, and therefore its contact points will not be detected by the camera. As shown in Figure 3.2, the sample must be an opaque surface in order for the internally reflected light to reflect off the contact points and into the camera’s sensor. The first samples, as shown in Figure 5.2, were made white by use of a white dye while mixing the PDMS cross-linker and base elastomer. White samples were too reflective and caused images to be over-saturated at points of contact. This was effectively mitigated by using black samples—by mixing a run-of-the-mill black dye into the sample mixture during the curing process—in lieu of white samples. Another goal of these samples would be that they can easily be tuned in terms of their various parameters, primarily geometry and stiffness. To avoid issues with not having samples cure properly from using an inadequate ratio of polymer to crosslinker, the latest version of the samples are made from the Dragon Skin brand, specifically Dragon Skin Medium. Dragon Skin is a silicone rubber, and there exist several types of stiffness for this brand, as shown in Appendix C. These samples, in the end, were sufficient for the function of the rest of the apparatus. An example of these final Dragon Skin samples as well as a 3D printed mold are shown in Figure 5.2.

One final issue that was observed relates to the quality of degassing the samples. If the samples were not properly degassed (that is, placed in a vacuum chamber and connected to an air pump), then asperities would oftentimes form incorrectly or perhaps have bubbles trapped on the surface. This would result in an increased amount of scattered light, which would then render the contact area measurement as rather inaccurate. Once the samples were thoroughly degassed, the inaccuracies were effectively mitigated, as shown in Figure 6.1.
Figure 5.1: An initial white PDMS sample.

Figure 5.2: A silicone rubber sample casted from a 3D printed mold using the Dragon Skin material, whose spec sheet is given in Appendix C.

5.2 Optics

As already mentioned, the optical arrangement was finalized after one iteration and so there did not exist any evolution of this component over the lifetime of the project.
5.3 Motor Control

Algorithm 1 GoToForce.m Pseudo Code

1: \textbf{procedure} GoToForce(Stage, loadCell, desiredForce) \hspace{1em} \triangleright \text{A function which moves the motor stage to a desired force}

2: currentForce = MeasureForce(loadCell)

3: \hspace{1em} \textbf{if} currentForce \leq \text{forceTol} \hspace{1em} \textbf{then}

4: \hspace{2em} WithinLimit = true

5: \hspace{1em} \textbf{else}

6: \hspace{2em} \textbf{while} WithinLimit \hspace{1em} \textbf{do} \hspace{1em} \triangleright \text{We are done when the measured normal load is within the prescribed force tolerance}

7: \hspace{3em} error = currentForce - desiredForce

8: \hspace{3em} newPosition = TransferFunction(error) \hspace{1em} \triangleright \text{Apply the transfer function which incorporates the PID controller}

9: \hspace{3em} MoveMotorStage(newPosition)

10: \hspace{2em} \textbf{return} currentForce \hspace{1em} \triangleright \text{The desired force is achieved}

Figure 5.3: A plot demonstrating the optimal value of $K_D$ and $K_I$, which are both zero. This demonstrates that this system is optimized for a strictly proportional controller.

To fully implement the PID controller, the typical PID tuning methodology was used, which
consists of first underdamping the system to introduce oscillations (for this system, ensuring that the motor oscillates between displacements and therefore about some desired normal force) and to then tuning $K_I$ and $K_D$. The essential MATLAB logic for the feedback loop is shown in Algorithm 1. As shown in Figure 5.3, the ideal values of $K_I$ and $K_D$ were in fact 0. In other words, the system was optimized as simply a proportional controller. Retrospectively this makes sense as this concludes that the system may be modeled by Hooke’s Law: $F = kx$.

![Graph](image)

Figure 5.4: A plot demonstrating the optimal value of $K_P$, which is around 1.1 in normalized units.

After discovering the values of $K_I$ and $K_D$, a parametric sweep over $K_P$ was then performed to find its optimal value. As shown in Figure 5.4, this value is roughly 1.1. One should also note the units of the proportionality constants here. They have all been scaled to the effective spring constant of the system, which was obtained by running a linear regression on measured force versus displacement data (that is, measuring the normal load for increasing motor displacements).

### 5.4 Real-time processing

This feedback loop, on the whole, worked quite well, except for one consideration. While the images are being obtained—which then are used for determining the contact area—the sample continues to age as this process takes some finite amount of time. To account for this, a small
The amount of area increase is included in the transfer function:

\[
\Delta F_N(F_N) = \frac{\Delta A_R(F_N)}{\beta} + b(F_N) \ast \log(t_{\text{sampling rate}})
\]  

(9)

Note here that \( b \) here is a function of the normal load, meaning that the aging rate is dependent on the normal load. This coefficient itself has to be found through performing a linear regression on various aging experiments, and is similar to the loading transfer function calibration mentioned earlier.

### 5.5 Post-processing

For a first iteration, the properties available in MATLAB’s object detection function \textit{regionprops} were used. Firstly, MATLAB provides the overall area and perimeter of an object, which in this case corresponds to exactly the parameters the SMR Lab would look to characterize. Secondly, MATLAB provides information on the centroid—really just the geometric center of mass of a clump of pixels, which would work excellently if the centroidal movement was minimal from image to image—with little individual asperity movement. If this was the case, then the asperities could be matched from image to image by determining if the magnitude of the distance between two asperities’ centroids from two adjacent images (in time or load) was beneath some length threshold. As has already been mentioned, though, asperities are merging, splitting, appearing, and disappearing throughout the experiments. Imagine two asperities which are shaped in such a way that their centroids are quite far apart, but when they merge, their centroid moves quite close together. Clearly this new centroid could be farther from the two merged asperities’ original centroids than two other non-merged asperities that are rather close to one another.

This issue was fixed in two ways. First, replaced the centroidal approach was replaced with the \textit{PixelIdxList} property in MATLAB. This property lists all of the pixels in an object as a vector, and so instead of comparing centroids, this algorithm now involves cross checking pixel lists between asperities in two adjacent images and setting some threshold for the overlap (30% works fairly well). That is to say, if two asperities share more than 30% of the same pixels, then they are labeled as the same asperity.

Then to resolve the issue of mislabeling, a matrix of asperity tags was created. The previous
system involved trying to sort through the asperities as their statistics (centroid, perimeter, area) were being placed in their respective arrays. This is approach on the order of $O(n^4)$, and requires, for example, roughly 5 minutes of computing time for 100 images with approximately 100 asperities each. The best system seems to be marking all asperities that combine or split as the same object, and to label disappearing and appearing asperities as if they were any other asperity in the images they appear.

6 Evaluation of Design

6.1 Image Acquisition and Apparatus Optics

To quickly reiterate the data collection process: one may infer the frictional properties of an interface from its area, and in fact in most of the literature these two quantities are seen as interchangeable. One may observe the real contact area (not the nominal or apparent contact area) by means of detecting the emitted light from a totally internally glass plate. See Figure 3.2 for a depiction of this setup. A camera may then collect images at different time intervals, and the intensity values from these images show how the real contact area of the surface evolves. To transform a standard RGB image to a binarized image, the raw images are first reduced to a 2D array of the blue pixel values since blue LEDs ($\lambda=470\text{nm}$) shine on the glass plate-material interface. A threshold is then established by calibrating area vs. load plot to a linear fit. Because the images are grayscale, each pixel may be binarized if it is below or above a certain intensity value between 0-255. This may be done due to the background theory mentioned in Section 2 which indicates that the real contact area is directly proportional to the normal force; see Figure 6.8 for an example of this trend observed with the apparatus defined in this report. For an example of the logarithmic trend in aging data, also see Figure 6.8. By choosing the intensity threshold value which has the lowest Coefficient of Correlation, one may then apply this value to the aging data. This threshold is determined for a set of parameters: the individual sample, its orientation, and the intensity of light (determined by using a power supply which can apply variable voltages to the set of LEDs attached the glass fixture). See Figure 6.1 for an example of a gray scale and a binarized image taken with a loading protocol.

As seen in Figure 6.1, the mean asperity size is clearly resolvable for this setup. Moreover, as seen in Figure 6.2, the size of the deformations (both due to load and time) for the available forces
Figure 6.1: Examples of Raw and Binarized Images. (a) Raw, grayscale image taken with the Thor Labs CMOS sensor Canon Macro Lens of a sample at 50N (b) Thresholded image are also resolvable, which implies that the samples and the optics are within the specifications.

This all of course is to highlight and explore a discrepancy between the deformations due to aging and loading. If it can be determined that aging is highly dependent on the geometry and that loading, which the data has shown, is not, then this will become an invaluable part of and perhaps the central part of the model developed in this project. Though no conclusions may be drawn about Figure 6.2, it is interesting to note the apparent differences that already exist between aging and loading in 2D. As expected, loading seems to consist more of new asperities coming into contact, whereas aging looks to comprise the same asperities being stretched out. Whether or not this is actually the case has yet to be determined, but this preliminary data is promising. Once the sample holder has been machined, not only will the data be highly reproducible, but these contour plots will be able to be compared directly together as opposed to evaluating them independently.
Figure 6.2: Contour plots of (a) Loading and (b) Aging for various load and time steps, respectively. This is an example of the potential analyses that could be done with data generated on the experimental apparatus built throughout this project.

6.2 Checking Uniformity of Applied Force

To check that the distribution of forces are evenly shared across the samples, a calibration piece was inserted into the apparatus. This calibration piece was created by means of the typical casting process with the Dragon Skin silicone rubber material, but the mold here took the form of a petri dish. A load of 100 N (the maximum applied across all experiments) was then applied to this calibration piece. Given that the low amount of scattering here, as well as the fact that the intensity correlates with the normal load, one can use intensity variance as a proxy for force variance. As shown in Figure 6.3, the intensity variation across the image is quite minimal, and therefore the force is as well. One should also note that the image here has been normalized for bubbles (due to imperfect degassing), and so all bright spots with an intensity above a certain value have been removed, so as to facilitate one’s ability to discern trends in the normal force. If the intensity profiles, for example, decreased in the $y$-direction, then one could summarize that in fact there was a tilt in the apparatus in the $y$-direction. However intensity variance does not trend in any discernible fashion and therefore we may conclude that the force does not either.
(a) A picture of the calibration piece, marked with lines for the intensity profiles in red

(b) A plot of vertical and horizontal intensity profiles along the line segments indicated in 6.3a

(c) A surface plot of the intensity of the image

Figure 6.3: A series of plots demonstrating the uniformity of the intensity throughout the image.
6.3 Motor Control

Figure 6.4: A plot of force vs. time for the motor-loadcell-computer feedback system. Here the simple command of maneuvering to a force and waiting is given while force measurements are taken. Clearly the system demonstrates 1) a response time (5.38s, in blue) and 2) a non-negligible relaxation (0.5 N over 115 seconds, in green).

Shown in Figure 6.4 is the relaxation of the system over the course of two minutes (as well as the 5 second response time mentioned in Section 4). In other words, over a few minutes the motor relaxes thereby decreasing the applied load. Some sort of periodic correcting factor must be implemented as this relaxation currently renders the data virtually unusable for 2D analyses (although the 1D contact area data is fine). As a simple fix, one could periodically calculate the current error between the expected load and the actual load, and make the corresponding adjustment. This adjustment would be problematic, especially in the beginning of the experiment, in that it adds some latency to the entire system, potentially offsetting the time intervals between successive images.

This effect has now been effectively removed via the PID controller as seen in Figure 6.5. Now the mean response time has been reduced to roughly 2.0 s for loading to 50 N—a typical load in these experiments—and is moreover quite consistent, as in Figure 6.7. A second issue, as in Figure 6.4, is that a decay in the normal force was observed when uncorrected for. An addition to the experimental script now detects every 1 ms whether the force is within some tolerance—± 0.1N. This solution works rather well, as shown in Figure 6.6 given that no force measurement exceeds the given force tolerances.
Figure 6.5: Upward and downward step responses for various forces. Here one may see that the upward step response here (2.15s, 1.81s) is much faster than the previous upward step response as seen in Figure 6.4 (5.8s).

Figure 6.6: Motor movements over the course of one 5 minute experiment.
Figure 6.7: Plot of motor movement response times for several iterations. Note the consistency of the control system.

6.4 Final Contact Area Measurements

After improving the feedback loops to improve the volume of data for both loading and aging experiments, the apparatus provides, in general, highly repeatable data for both types of experiments. As expected, the contact area growth is highly linear as a function of normal load and moreover continuously increases as a function of time for a constant load, as seen in Figure 6.8.
6.5 Area Feedback

To briefly summarize the reasoning for having the area feedback in the first place, as shown in Figure 6.8, this system produces highly repeatable data with little variances between experiments. In general, it is difficult to isolate out the effects of aging and loading, perhaps the reason their differences have not been extensively studied. When a load is applied to an interface, it of course takes the stage some finite amount of time to reach the right displacement. While the motor is moving, the interface continues to age while these data points are being taken. For the loading data shown in this report, in general the contact area growth of aging is small compared to the total growth due to loading. This may be deduced from performing measurements of the contact area growth at various loads, and then estimating these contributions over the course of a loading experiment. However, in the limit of large time steps between load steps (the motor is moving very slowly as it moves to another load), aging begins to dominate. This then reduces the number of data points that can be collected over the course of a loading experiment, as the motor must maintain some average speed to outrun the effects of aging. The feedback loop which maintains the contact area at a constant value works fairly well—as seen in Figure 6.9.
6.6 Asperity Tracking

Figure 6.10: Here one may see how the labeling scheme seems to do quite well. Notice how the asperities are labeled with the same tags over the course of a 10,000s experiment.

In Figure 6.10, one may see the labeling scheme in action, which clearly shows that individual asperities are consistently tracked between time-steps (frames). Moreover, this method has been tested on various experiment types, where the histories of individual asperities are quite chaotic.
In Figure 6.11, one may see how this sort of analysis can be used. These individual asperities grow at different rates (in fact some are even non-linear) which sum up to a macroscopic linear relation between the contact area and the normal load.

6.6.1 Checking for the Existence of Image Shift

As described above, one of the types of experiments to be run on this apparatus is one where the contact area is kept constant. The initial data from this experiment, however, was quite concerning as it looked to be the case that there may have been some mechanical shift in the samples, either over time or over some loads. The concern was that there was some sort of shear force due to a misalignment in the apparatus that had gone undetected. Another explanation could be that there was some sort of Poisson stretching of the sample’s bulk, due to forces and torques at the interface. To investigate this, whole pixel shifts were first investigated. This was done by writing an autocorrelation script (see Appendix D) which would take a subsection of the first image in an experiment, and shift it by some whole pixel value in the x direction, y direction, or both. This autocorrelation function takes the absolute value of the difference between the binarized images (the first, shifted image and the nth, unshifted image), adds up the values of 1 (i.e. the points of contact that are not in common between the two), and looks for an image shift which minimizes this number. This absolute value of the difference between two binarized images has been named
the (Se)cond (M)oment (o)f (A)rea, as it seems to resemble the second moment in that it roughly corresponds to the variance of image differences.

If there was a shift being introduced in the x direction, for instance, then the autocorrelation function would be minimized at some non-zero point in the x-direction, and the autocorrelation which produce a shift of some $x$ for the $n$th image. This, however, was not the case. Across multiple experiments (different samples in different orientations at different loads), the absolute difference between shifted images was in fact minimized for a shift of zero, other than some small noise corresponding to 1 px distributed randomly across the images.

$$\mu = \text{img}_1 \ast A + (A - 1) \ast \text{img}_2,$$

where $A$ varies from 0 to 1. We may then threshold these images, take the absolute difference of these binarized (i.e. count how many pixels are different), and see how this difference grows with the shift. This may be found in Figure 6.12. We may see that the magnitude and shape of this sub pixel shift do not align with the absolute difference found in Figure 7.2, which indicates that there is no linear image shift. However, it may also be the case that there are some dynamics in the material—such as a Poissonian stretching of some sort—that is responsible for this apparent “shift.” In any case, this “shift” does not seem to be an effect of the experimental apparatus and therefore is outside the scope of this project.

![Figure 6.12: A plot showing the absolute difference generated from subpixel shifts. The magnitude of the values here are much greater than that of 7.2 and therefore may not explain the apparent asperity shift.](image)
7 Conclusions and Future Work

7.1 Summary of Results

<table>
<thead>
<tr>
<th>Component</th>
<th>Target Metric</th>
<th>Achieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Generation</td>
<td>Mean feature size ≥10 px</td>
<td>100 px</td>
</tr>
<tr>
<td>Optics</td>
<td>Resolves ≤50µm</td>
<td>46 µm</td>
</tr>
<tr>
<td>Motor Control</td>
<td>Loading ≤ 5.0s; Variation ≤ 10%</td>
<td>Loading time 2.5s; Max. Variation 15%</td>
</tr>
<tr>
<td>Real-Time Processing</td>
<td>Variation in constant area ≤ 1%</td>
<td>1% on long time scales (≥10s)</td>
</tr>
<tr>
<td>Post-processing</td>
<td>0 mismatches in feature tracking</td>
<td>0 mismatches</td>
</tr>
</tbody>
</table>

Table 7.1: The metrics which the project is measured, along with the realized specifications.

As shown in Table 7.1, each component here is within the initial design specification. These working components together produce a device capable of characterizing the real time contact area evolution in frictional interfaces. One should note, however, that several of these design parameters are heavily interdependent. If, for example, the system were to be scaled down to resolve asperity deformations in a metal, for example, the instrumentation would change, but the overall control systems, post-processing, etc. would not. Such a sample would require a much finer resolution, perhaps with a microscope capable of resolving scales at less than a micron. Nevertheless, we are confident that the methodology outlined in this report is highly versatile in its ability to process different sample sizes, resolutions, and other such parameters.

7.2 Implications of Results

7.2.1 Constant Contact Area Protocol Implications

As a result of the constant contact area protocol, this apparatus has already provided some insight into what the differences between aging and loading may be. Again, this result is not really the focus of this project, but is rather a byproduct which ought to be rather useful for future uses of the apparatus by members of the SMR Lab. In Figure 7.1, one may see how differences in deformations arise due to loading and aging in an interface, and moreover that most importantly these differences do seem to exist. The explanation of how and why these differences arise could be investigated with this apparatus, but, at the risk of being redundant, such a task is outside the scope of a design engineering project and will be undertaken by researchers in the SMR Lab.
These differences, when compared to the initial image, are summed to produce what is called the absolute difference between images. This absolute difference shows us whether the dynamics of the interface really matter: in other words, if aging and loading cause distinct deformations in frictional interfaces. If these deformations were *not* distinct, then this absolute difference would simply resemble noise dependent on the accuracy of the constant area feedback loop. In Figure 7.2, however, one may see that this absolute difference is in fact increasing as a function of time, which is in essence a more quantitative conclusion reached with the discussion regarding Figure 7.1.

Drawing conclusions on the nature of aging and loading—if the reader will excuse my enthusiasm—a thrilling investigation. As has already been mentioned, (to my knowledge) no current rate-and-state models of friction differentiate between histories in the contact area; arriving to one specific contact area level for an interface—and therefore one specific frictional strength—primarily due to aging or primarily due to loading has been factored in thus far. This apparatus as well as its complementary software methods demonstrate that in fact the details in the dynamics of an interface are in fact significant.

![Image of deformations](image)

Figure 7.1: Over time, one may see that loading and aging in fact seem to cause distinct deformations in frictional interfaces.
7.2.2 General Implications of Silicone Rubber

PDMS had not previously been tested for detectable aging effects in the SMR Lab. With the fact that PDMS does age quite well, PDMS can definitively be used for generating data. And as has already been mentioned, PDMS will function well for two primary reasons. The first is that by adding a cross linking polymer during the baking process, one can vary the stiffness of the PDMS sample. In effect, this provides an incredibly useful dial to turn in varying the material properties of the samples in a consistent manner. Secondly, PDMS may be cleaned from its mold after being baked, which allows for recycling molds for different experiments. This could previously have been done by 3D printing the same STL files, but one could argue that there are far fewer inconsistencies from casting to the same mold than 3D printing the same STL file many times. Though PDMS currently was the material of choice for this project—both for this capstone and for the SMR Lab more generally—this polymer could very well be any sort of material that can be molded.

7.2.3 Asperity Tracking

Because there are roughly several hundred asperities for these samples, this tool furthermore could potentially allow for an analysis with machine learning that could predict what the evolution
of a frictional interface could be, based on the contact area at a certain load. Such a result would be highly useful in real-world applications. One could, for example, place a shoe into the apparatus and because real contact area may be used as a proxy for frictional strength, the evolution of the shoes frictional strength could be predicted as both a function of load and time. The only caveat here, however, is that this interface is between soda-lime glass and some other type of sample—in our case, a shoe. However, the evolution of the real contact area between glass and our shoe would offer a good model for predicting the evolution, and theoretically could be mapped onto some other combination of materials, such as our model shoe with brick or asphalt.
8 References


Appendix A  Bill of Materials
Table A.1: Bill of Materials

<table>
<thead>
<tr>
<th>Item</th>
<th>Vendor</th>
<th>Source</th>
<th>Price</th>
<th>Quantity</th>
<th>Item Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>80W Power Supply</td>
<td>ExTech</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>2</td>
<td>0.0</td>
</tr>
<tr>
<td>USB Data Acquisition Card</td>
<td>National Instruments</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>LSB200 Load Cell</td>
<td>Futek</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>6&quot; X 8&quot; Mirror</td>
<td>Amazon</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>Monochrome CMOS Sensor</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>EF-S 35mm f/2.8 Macro IS STM Lens</td>
<td>Canon</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>3/4-Inch-by-6-Inch-by-.091-Inch Spring</td>
<td>Farney</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>4</td>
<td>0.0</td>
</tr>
<tr>
<td>300 mm Translation Stage</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>Blue LEDs</td>
<td>Lighting Ever</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>Duct Tape</td>
<td>Gorilla Tape</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>PF175 - Clamping Fork</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>Nexus Optical Table</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>TR2 0.5&quot; OD Post</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>9</td>
<td>0.0</td>
</tr>
<tr>
<td>P12 1.5&quot; OD Post</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>4</td>
<td>0.0</td>
</tr>
<tr>
<td>80/20 40mm X 36mm L bracket</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>8</td>
<td>0.0</td>
</tr>
<tr>
<td>80/20 T-Nut</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>4</td>
<td>0.0</td>
</tr>
<tr>
<td>1/4&quot;-20 Bolt</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>29</td>
<td>0.0</td>
</tr>
<tr>
<td>Headless 1/4&quot;-20 Bolt</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>5</td>
<td>0.0</td>
</tr>
<tr>
<td>UB2 Universal Base Plate</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>3</td>
<td>0.0</td>
</tr>
<tr>
<td>AP90 Right Angle Plate</td>
<td>Thor Labs</td>
<td>Faculty Lab</td>
<td>0.0</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>VeroBlue Resin</td>
<td>Stratasys</td>
<td>Wyss Institute</td>
<td>0.0</td>
<td>300 mL</td>
<td>50.00</td>
</tr>
<tr>
<td>Form2 Resin</td>
<td>Form Labs</td>
<td>Active Learning Labs</td>
<td>175/L</td>
<td>300 mL</td>
<td>50.00</td>
</tr>
</tbody>
</table>

41
Figure B.1: A photo of the apparatus in the SMR Lab space, equipped with the motor stage, loadcell, camera, LEDs, and support structure.
Motor Stage

PARTS

(1) 300 mm Thor Labs Long Travel Stage
(8) 0.5" OD x 2" TR2 Optical Posts
(1) AP90 Perpendicular Plate
(1) AP90RL Perpendicular Plate
(19) 1/4"-20 Bolts
(2) UBP2 Base Plate

This stage is responsible for applying a load to the samples.
Support Structure

PARTS

<table>
<thead>
<tr>
<th>PARTS</th>
<th>NAME</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2) 10' 80/20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4) T-Nut</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4) 1/4-20X3/4&quot; Stud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(8) 1/4-20 Hex Bolts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(8) 1/4-20 Hex Nuts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(8) 1/4-20 Bolts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(8) 80/20 1.5&quot; L-bracket</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4) Thor Labs 1.5&quot; ODx10&quot; Post</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DO NOT SCALE DRAWING

TITLE:

Support Structure

SIZE

A

SCALE: 1:12
WEIGHT:

SHEET 1 OF 1
300 mm Linear Translation Stage with Integrated Controller, Stepper Motor

**Overview**

### LTS300

- **Integrated Controller with Keypad and Remote USB Control**
- **Stackable in XY, XZ, and XYZ Configurations**
- **Minimum Calibrated On-Axis Accuracy of 5.0 µm**
- **Horizontal Load Capacity of 15 kg (33.1 lbs)**

### Electrical Specifications

<table>
<thead>
<tr>
<th>Motor Specifications</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Step Angle</td>
<td>1.8° (50 Poles ±2 Phases for 360° Divided by 200)</td>
</tr>
<tr>
<td>Step Accuracy</td>
<td>5%</td>
</tr>
<tr>
<td>Rated Phase Current</td>
<td>0.85 A</td>
</tr>
<tr>
<td>Phase Resistance</td>
<td>5.4 Ω</td>
</tr>
<tr>
<td>Phase Inductance</td>
<td>5.6 mH</td>
</tr>
<tr>
<td>Holding Torque</td>
<td>20 N·cm</td>
</tr>
<tr>
<td>Detent Torque</td>
<td>2.0 N·cm</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>-20 to 40 °C (Motor Specification Only)</td>
</tr>
</tbody>
</table>

### Controller Specifications

| Microsteps per Full Step | 2048 |
| Microsteps per Revolution of Motor | 409,600 (for 200 Step Motor) |
| Motor Drive Voltage      | 24 V |
| Motor Drive Power        | 12.5 W (Avg) Up to 25 W (Peak) |
| Motor Speeds             | Up to 3000 RPM (200 Full Step Motor) |

### Input Power Requirements

- Voltage: 24 VDC
- Power: 25 W (Peak)

### Power Supply Requirements

- 90 - 264 VAC (47 - 63 Hz)

### Notes

- **Max Velocity, Max Acceleration, and Max Load**
  - The max velocity and acceleration values quoted above are achievable with lighter loads. As the load is increased, the velocity and acceleration should be decreased accordingly. For the maximum 15 kg load, the velocity should be reduced to either 15 mm/s with 3 mm/s² acceleration or 12 mm/sec with 5 mm/s² acceleration, depending upon whether speed or acceleration is more important for the intended application.
- **Min Achievable Incremental Movement**
  - The measured minimum incremental motion that the stage can achieve, also referred to as the minimum step size.
- **Min Repeatable Incremental Movement**
  - The minimum incremental motion that the stage can repeatedly achieve within its standard error.
- **On-Axis Accuracy**
  - This is the absolute accuracy of the commanded position. It is defined as the maximum discrepancy between the commanded position and the absolute position over the full travel of the stage. For example, if a stage is specified with an on-axis accuracy of 20 µm, then a command to travel to 10 mm will result in an absolute position of within 20 µm of 10 mm. This value will tell you the maximum possible inaccuracy at any point in your travel.
  - However, sometimes a more useful specification can be maximum percentage accuracy (see Note C for details) as the discrepancy between commanded position and absolute position generally increases linearly with the amount of travel. This gives rise to an on-axis accuracy near the end of the travel range of the stage which is generally lower than the accuracy at the beginning of the travel range. A percentage accuracy can give you a good idea of what to typically expect along the stage's travel. Accuracy should not be confused with repeatability, which refers to the ability of the stage to travel to the commanded position over many attempts.
- **On-Axis Maximum Percentage Accuracy**
  - This is the maximum discrepancy between the commanded position and the absolute position expressed as (|P_{absolute} - P_{commanded}| / P_{commanded}) and gives a better idea of what to typically expect along a stage's travel. The advantage of expressing this as a percentage is that the higher accuracy at the lower travel range of a stage can now be represented. As the maximum percentage error will not necessarily be at full travel, this means the value given will be an excellent representation of the maximum possible error that can be expected.

---

CMOS Cameras: USB 2.0 and USB 3.0

Overview

Software

Features

- Easy to Use in a Wide Range of Applications from Microscopy to Monitoring
- Two Sensor Resolutions Available:
  - 1.3 Megapixel (1280 x 1024 Pixels) Monochrome, Color, and NIR CMOS Sensors
  - 2.3 Megapixel (1936 x 1216 Pixels) Monochrome and Color CMOS Sensors
- Available with Global Shutter and External Trigger
- Software Support:
  - ThorCam GUI with 32- and 64-Bit Windows® 7 or 10 Support
  - Included SDK Supports C++, C#, VB, and LabVIEW
  - Compatible with µManager / ImageJ

These compact, lightweight CMOS cameras are available with either a monochrome (M models), color (C models), or NIR (N model) sensor. They can be used in a wide range of applications from microscopy to monitoring. Our CMOS cameras offer a full-frame resolution of 1280 x 1024 pixels or 1936 x 1216 pixels. All camera series are controlled and powered via a standard 5 V USB 2.0 or 3.0 port.

The DCC1545M and DCC1645C compact CMOS cameras have an electronic rolling shutter and their small footprints make them ideal for applications where space is a premium. The DCC1240 and DCC3240 high-sensitivity CMOS cameras include CMOS sensors that allow for switching between rolling and global shutter mode, offer a considerably higher dynamic range, and include an input for an external trigger. Finally, the DCC3260 CMOS cameras utilize the Sony IMX249 sensor with 2.3 megapixel resolution and very low read noise of <7 e-.

A brief comparison of the features available in each model is presented in the table below. For a detailed list of specifications, see the Specs tab.

For quantitative applications requiring low noise, high quantum efficiency cameras, consider our Quantalux® sCMOS and Kiralux™ CMOS Cameras.

USB and Trigger Cables

For the DCC1240 cameras, optional CAB-DCU-T1 and CAB-DCU-T2 USB and trigger cables allow one to use the additional trigger input and output ports (T1 and T2) of these cameras together with the USB 2.0 connection. The exposure and readout/transfer events of the camera can be initiated via the input trigger, and external events like strobe lights can be triggered by the camera using the output trigger. The CAB-DCU-T3 GPIB cable can be used with the USB 3.0 cameras as an additional means of connecting and triggering peripheral devices. The trigger configuration (i.e., the source of the input trigger and the timing for the output trigger) can be set via the provided software or the LabVIEW driver.

Software

Each camera also comes with ThorCam, our Windows-compatible GUI software package. Standard drivers like Direct Show (WDM) and .NET are provided and offer support for LabVIEW. An extensive SDK is available. The C/C++ drivers can additionally be imported to Matlab using MEX files.

<table>
<thead>
<tr>
<th>Item #</th>
<th>DCC1545M</th>
<th>DCC1645C</th>
<th>DCC1240M</th>
<th>DCC1240C</th>
<th>DCC3240M</th>
<th>DCC3240C</th>
<th>DCC3240N</th>
<th>DCC3260M</th>
<th>DCC3260C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>1.3 Megapixels (1280 x 1024)</td>
<td>2.3 Megapixels (1936 x 1216)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensor</td>
<td>Monochrome</td>
<td>Color</td>
<td>Monochrome</td>
<td>Color</td>
<td>Monochrome</td>
<td>Color</td>
<td>NIR</td>
<td>Monochrome</td>
<td>Color</td>
</tr>
<tr>
<td>Exposure Mode</td>
<td>Rolling Shutter</td>
<td>Global and Rolling Shutter</td>
<td>Global Shutter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interface and Included Cable</td>
<td>USB 2.0</td>
<td>USB 3.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input/Output Trigger</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Compact USB 2.0 CMOS Cameras

- Color and Monochrome Versions Available
- Electronic Rolling Shutter
- USB 2.0 Connection in an Ultra-Compact Housing
- 25 fps in Freewrun Mode and over 200 fps with Limited Area of Interest
- Ships with USB 2.0 Cable

Related Items

CMOS &sCMOS Scientific Cameras

C-Mount Camera Lenses

USB CCD Cameras

High-Magnification Zoom Lenses

The DCC1545M and DCC1645C CMOS cameras operate with only a rolling shutter and feature an ultra-compact dustproof housing. Frame rates up to 250 fps are possible with a limited area of interest and sufficient light conditions. The small footprint and mini USB 2.0 connector at the side of the housing allow usage in setups where space is at a premium.

These cameras feature a CS-mount lens mounting thread. To equip any of our C-mount camera lenses, the included CML05 CS-mount to C-Mount extension adapter is required to ensure that the sensor is in the focal plane of the camera lens.

In addition, the DCC1545M and DCC1645C cameras are also shipped with CS-mount to SM1 internal and CS-mount to SM1 external thread adapters. Additional compatible adapters are available at the bottom of the page. Two 1/4"-20 screw adapters are also included to allow to color camera housing to be post mounted using 8-32 or M4 standard screws. Please note that larger lenses may need to be supported independently of the camera.

Our color CMOS cameras have an IR shortpass filter that cuts off transmission above 650 nm. Removing the filter will expose the CMOS sensor to the environment, which could result in dust entering the camera and causing the performance to deteriorate. For those who are very familiar with cameras and sensors, it is possible to change the filter yourself in a cleanroom environment. If you are not comfortable performing this procedure, please send the camera to Thorlabs where our skilled technicians have the tools to safely remove the filter without damaging the camera. Contact technical support for assistance.

These cameras are compatible with our C-Mount Camera Lenses and High-Magnification Zoom Lenses using the included CS to C-Mount adapter. Our standard lenses include fixed focal lengths from 3.5 mm to 75 mm with maximum apertures of up to f/0.56, as well as an 18 - 108 mm focal length, f/2.5 zoom lens. Our high-magnification zoom lenses are a modular system that features magnifications from 0.07 to 28.

### High-Sensitivity CMOS USB 2.0 Cameras with Global Shutter

<table>
<thead>
<tr>
<th>Item #</th>
<th>DCC1240M</th>
<th>DCC1240C</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOS Sensor Type</td>
<td>Monochrome</td>
<td>Color</td>
</tr>
<tr>
<td>Sensitivity Graph</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exposure Mode</td>
<td>Global and Rolling Shutter</td>
<td></td>
</tr>
<tr>
<td>Read Out Mode</td>
<td>Progressive Scan</td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>1280 x 1024 Pixels</td>
<td></td>
</tr>
<tr>
<td>Optical Sensor Format</td>
<td>1/1.8&quot;</td>
<td></td>
</tr>
<tr>
<td>Pixel Clock Range</td>
<td>7 - 35 MHz</td>
<td></td>
</tr>
<tr>
<td>Frame Rate, Freerun Mode</td>
<td>25.8 fps</td>
<td></td>
</tr>
<tr>
<td>Trigger Input</td>
<td>9-Pin, D-Sub Connector</td>
<td></td>
</tr>
<tr>
<td>Lens Mounting Thread</td>
<td>C-Mount (1.00&quot;-32)</td>
<td></td>
</tr>
<tr>
<td>Post Mounting Threads</td>
<td>8-32 and M4 Taps, 5 mm Deep</td>
<td></td>
</tr>
<tr>
<td>Dimensions (H x W x D)</td>
<td>40.4 mm x 32.0 mm x 41.5 mm (1.59&quot; x 1.26&quot; x 1.63&quot;)</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>0.16 lbs (74 g)</td>
<td></td>
</tr>
<tr>
<td>Included Adapters</td>
<td>C-Mount to External SM1 and C-Mount to Internal SM1</td>
<td></td>
</tr>
</tbody>
</table>

a. Depends on the PC hardware used.
b. Requires maximum pixel clock frequency.
c. Please note that CS-Mount and C-Mount lens mounts both use 1.00"-32 threads but perform different flange-to-sensor distances.
d. Be careful not to thread a screw longer than the depth of the tap into the camera housing, as this could lead to damage.
e. The included CS to C-Mount adapter is not anodized. The black anodized CML05 adapter is available as a replacement or substitute.

Based on your currency / country selection, your order will ship from Newton, New Jersey

### High-Sensitivity USB 3.0 Cameras with Global Shutter

<table>
<thead>
<tr>
<th>Item #</th>
<th>DCC3240M</th>
<th>DCC3240C</th>
<th>DCC3240N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOS Sensor Type</td>
<td>Monochrome</td>
<td>Color</td>
<td>NIR Monochrome</td>
</tr>
</tbody>
</table>

Based on your currency / country selection, your order will ship from Newton, New Jersey

<table>
<thead>
<tr>
<th>Lens</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Focal Length &amp;</td>
<td>35mm f/2.8</td>
</tr>
<tr>
<td>Maximum Aperture</td>
<td></td>
</tr>
<tr>
<td>Lens Construction</td>
<td>10 elements in 6 groups</td>
</tr>
<tr>
<td>Diagonal Angle of View</td>
<td>42°35'</td>
</tr>
<tr>
<td>Focus Adjustment</td>
<td>AF with full-time manual</td>
</tr>
<tr>
<td>Closest Focusing</td>
<td>0.43 ft / 0.13m</td>
</tr>
<tr>
<td>Distance</td>
<td></td>
</tr>
<tr>
<td>Filter Size</td>
<td>1.9 in / 49mm diameter</td>
</tr>
<tr>
<td>Max. Diameter x Length, Weight</td>
<td>Ø2.7 x 2.2 in, approx. 6.7 oz / Ø69.2 x 55.8mm, approx. 190g</td>
</tr>
</tbody>
</table>
FEATURES
• Up to 10 times the overload protection
• Overload is available in Tension and Compression
• Light weight
• Notable nonlinearity
• Loads up to 100 lb (445 N)
• Miniature size

SPECIFICATIONS

PERFORMANCE
Nonlinearity ±0.1% of RO
Hysteresis ±0.1% of RO
Nonrepeatability ±0.05% of RO

ELECTRICAL
Rated Output (RO) See chart on third page
Excitation (VDC or VAC) 10 max
Bridge Resistance See chart on third page
Insulation Resistance ≥500 MΩ @ 50 VDC
Connection #29 AWG, 4 conductor, spiral shielded silicone cable, 5 ft [1.5 m] long
Wiring Code WC1

MECHANICAL
Weight (approximate) 0.3 oz [9 g]
Safe Overload 1000% of RO
200% tension only (50–100 lb)
Material Aluminum (10 g–10 lb), stainless-steel (25–100 lb)
IP Rating IP40

TEMPERATURE
Operating Temperature -60 to 200°F [-50 to 93°C]
Compensated Temperature 60 to 160°F [15 to 72°C]
Temperature Shift Zero ±0.01% of RO/°F [0.018% of RO/°C]
Temperature Shift Span ±0.02% of Load/°F [0.036% of Load/°C]

CALIBRATION
Calibration Test Excitation 5 VDC
Calibration (standard) 5-pt Tension
Calibration (available) Compression

MODEL LSB200
Miniature S-Beam Jr. Load Cell
Active end
+ Output (tension) - Output (compression)
FUTEK Label

www.futek.com
Model LSB200

**DIMENSIONS** inches [mm]

<table>
<thead>
<tr>
<th>Item #</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø 0.08 [Ø 2.1] nom</td>
<td>CABLE</td>
</tr>
<tr>
<td>Ø 0.13 [Ø 3.2] nom</td>
<td>SPRING</td>
</tr>
<tr>
<td>0.26 [6.7]</td>
<td></td>
</tr>
<tr>
<td>0.32 [8.0]</td>
<td></td>
</tr>
<tr>
<td>0.185 [4.7]</td>
<td></td>
</tr>
<tr>
<td>2X THREAD 0.110 [2.8] DEEP</td>
<td></td>
</tr>
</tbody>
</table>

**WIRING CODE** (WC1)

<table>
<thead>
<tr>
<th>Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RED</td>
<td>+ EXCITATION</td>
</tr>
<tr>
<td>BLACK</td>
<td>– EXCITATION</td>
</tr>
<tr>
<td>GREEN</td>
<td>+ SIGNAL</td>
</tr>
<tr>
<td>WHITE</td>
<td>– SIGNAL</td>
</tr>
<tr>
<td>SHIELD</td>
<td>FLOATING</td>
</tr>
</tbody>
</table>

**CAPACITIES**

<table>
<thead>
<tr>
<th>Item #</th>
<th>lb</th>
<th>N</th>
<th>RO (nom)</th>
<th>Bridge Resistance</th>
<th>Shunt Calibration Value</th>
<th>Deflection (in.)</th>
<th>Natural Frequency (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSH03867</td>
<td>10g</td>
<td>0.1</td>
<td>0.5 mV/V</td>
<td></td>
<td>301 kOhm</td>
<td>0.004</td>
<td>140</td>
</tr>
<tr>
<td>FSH03868</td>
<td>20g</td>
<td>0.2</td>
<td>1 mV/V</td>
<td></td>
<td>1000 Ohm nom</td>
<td>0.008</td>
<td>140</td>
</tr>
<tr>
<td>FSH03869</td>
<td>50g</td>
<td>0.5</td>
<td></td>
<td></td>
<td>150 kOhm</td>
<td>0.010</td>
<td>200</td>
</tr>
<tr>
<td>FSH03870</td>
<td>100g</td>
<td>1.0</td>
<td></td>
<td></td>
<td>1000 Ohm nom</td>
<td>0.008</td>
<td>300</td>
</tr>
<tr>
<td>FSH03871</td>
<td>250g</td>
<td>2.5</td>
<td></td>
<td></td>
<td>350 Ohm nom</td>
<td>0.007</td>
<td>530</td>
</tr>
<tr>
<td>FSH03872</td>
<td>1</td>
<td>4.5</td>
<td></td>
<td></td>
<td>60.4 kOhm</td>
<td>0.004</td>
<td>930</td>
</tr>
<tr>
<td>FSH03873</td>
<td>2</td>
<td>8.9</td>
<td></td>
<td></td>
<td></td>
<td>0.004</td>
<td>1340</td>
</tr>
<tr>
<td>FSH03874</td>
<td>5</td>
<td>22.2</td>
<td></td>
<td></td>
<td></td>
<td>0.005</td>
<td>1900</td>
</tr>
<tr>
<td>FSH03875</td>
<td>10</td>
<td>44.5</td>
<td></td>
<td></td>
<td></td>
<td>0.004</td>
<td>3000</td>
</tr>
<tr>
<td>FSH03876</td>
<td>25</td>
<td>111</td>
<td></td>
<td></td>
<td></td>
<td>0.004</td>
<td>2770</td>
</tr>
<tr>
<td>FSH03877</td>
<td>50</td>
<td>222</td>
<td></td>
<td></td>
<td></td>
<td>0.004</td>
<td>3600</td>
</tr>
<tr>
<td>FSH03878</td>
<td>100</td>
<td>445</td>
<td></td>
<td></td>
<td></td>
<td>0.005</td>
<td>4600</td>
</tr>
</tbody>
</table>

Drawing Number: FI1455

FUTEK reserves the right to modify its design and specifications without notice.
Please visit www.futek.com/salesterms for complete terms and conditions.

10 Thomas, Irvine, CA 92618 USA
Tel: (949) 465-0900
Fax: (949) 465-0905
www.futek.com
Product Overview

Dragon Skin™ silicones are high performance platinum cure liquid silicone compounds that are used for a variety of applications ranging from creating skin effects and other movie special effects to making production molds for casting a variety of materials. Because of the superior physical properties and flexibility of Dragon Skin™ rubbers, they are also used for medical prosthetics and cushioning applications. Dragon Skin™ rubbers are also used for a variety of industrial applications and have a service temperature range of a constant -65°F to +450°F (-53°C to +232°C).

Great for Making Molds for a Variety of Applications - Available in Shore 10A, 20A and 30A, Dragon Skin™ silicones can be used to make exceptionally strong and tear resistant molds for casting plaster, wax, concrete (limited production run), resins and other materials. Dragon Skin™ 10 AF is an anti-fungal silicone suitable for making a variety of skin-safe cushioning device configurations that resist fungi for orthopedic and orthotic applications.

Time Tested, Versatile Special Effects Material – Soft, super-strong and stretchy, Dragon Skin™ 10 (Very Fast, Fast, Medium and Slow speeds) is used around the world to make spectacular skin and creature effects. An infinite number of color effects can be achieved by adding Silc Pig™ silicone pigments or Cast Magic™ effects powders. Cured rubber can also be painted with the Psycho Paint™ system. Cured material is skin safe and certified by an independent laboratory.

Easy To Use – Dragon Skin™ silicones are mixed 1A:1B by weight or volume. Liquid rubber can be thinned with Silicone Thinner™ or Vacuum degassing is recommended to minimize air bubbles in cured rubber.

Technical Overview

<table>
<thead>
<tr>
<th>Material</th>
<th>Mixed Viscosity (cP)</th>
<th>Specific Gravity (g/cc)</th>
<th>Specific Volume (cu.in./lb)</th>
<th>Cure Time (min.)</th>
<th>Shore A Hardness (ASTM D-2240)</th>
<th>Tensile Strength (ASTM D-412)</th>
<th>100% Modulus (ASTM D-412)</th>
<th>Die B Tear Strength (ASTM D-524)</th>
<th>Shrinkage (in./in.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dragon Skin™ 10 Very Fast</td>
<td>23,000</td>
<td>1.07</td>
<td>25.8</td>
<td>4</td>
<td>30</td>
<td>10A</td>
<td>475</td>
<td>22</td>
<td>100%</td>
</tr>
<tr>
<td>Dragon Skin™ 10 Fast</td>
<td>23,000</td>
<td>1.07</td>
<td>25.8</td>
<td>8</td>
<td>75</td>
<td>10A</td>
<td>475</td>
<td>22</td>
<td>100%</td>
</tr>
<tr>
<td>Dragon Skin™ 10 Medium</td>
<td>23,000</td>
<td>1.07</td>
<td>25.8</td>
<td>20</td>
<td>5</td>
<td>10A</td>
<td>475</td>
<td>22</td>
<td>100%</td>
</tr>
<tr>
<td>Dragon Skin™ 10 Slow</td>
<td>23,000</td>
<td>1.07</td>
<td>25.8</td>
<td>45</td>
<td>7</td>
<td>10A</td>
<td>475</td>
<td>22</td>
<td>100%</td>
</tr>
<tr>
<td>Dragon Skin™ 10 AF</td>
<td>23,000</td>
<td>1.07</td>
<td>25.8</td>
<td>20</td>
<td>5</td>
<td>10A</td>
<td>475</td>
<td>22</td>
<td>100%</td>
</tr>
<tr>
<td>Dragon Skin™ 20</td>
<td>20,000</td>
<td>1.08</td>
<td>25.6</td>
<td>25</td>
<td>4</td>
<td>20A</td>
<td>550</td>
<td>49</td>
<td>60%</td>
</tr>
<tr>
<td>Dragon Skin™ 30</td>
<td>20,000</td>
<td>1.08</td>
<td>25.7</td>
<td>45</td>
<td>16</td>
<td>30A</td>
<td>500</td>
<td>86</td>
<td>364%</td>
</tr>
</tbody>
</table>

Mix Ratio: 1A:1B by volume or weight
Color: Translucent
Useful Temperature Range: -65°F to +450°F (-53°C to +232°C)
Dielectric Strength (ASTM D-147-97a): >350 volts/mil

Processing Recommendations

Preparation... Safety – Use in a properly ventilated area (“room size” ventilation). Wear safety glasses, long sleeves and rubber gloves to minimize contamination risk. Wear vinyl gloves only. Latex gloves will inhibit the cure of the rubber.

Store and use material at room temperature (73°F/23°C). Warmer temperatures will drastically reduce working time and cure time. Storing material at warmer temperatures will also reduce the usable shelf life of unused material. These products have a limited shelf life and should be used as soon as possible. Mixing containers should have straight sides and a flat bottom. Mixing sticks should be flat and stiff with defined edges for scraping the sides and bottom of your mixing container.

Cure Inhibition – Addition-cure silicone rubber may be inhibited by certain contaminants in or on the pattern to be molded resulting in tackiness at the pattern interface or a total lack of cure throughout the mold. Latex, tin-cure silicone, sulfur clays, certain wood surfaces, newly cast polyester, epoxy, tin cure silicone rubber or urethane rubber may cause inhibition. If compatibility between the rubber and the surface is a concern, a small-scale test is recommended. Apply a small amount of rubber onto a non-critical area of the pattern. Inhibition has occurred if the rubber is gummy or uncured after the recommended cure time has passed.

Because no two applications are quite the same, a small test application to determine suitability for your project is recommended if performance of this material is in question.
Appendix D  MATLAB Scripts

D.1  Feedback Protocols

D.1.1  Determine Area Function

```matlab
function [currentArea, img] = DetermineArea(cam, MemId, Bits, dimX2, dimX1,
    dimY1, dimY2, threshold)
    cam.Aquisition.Freeze(uc480.Defines.DeviceParameter.Wait);
    Height = 1024;
    Width = 1280;
    [~, tmp] = cam.Memory.CopyToArray(MemId);
    img = reshape(uint8(tmp), [Bits/8, Width, Height]);
    img = img(3, dimY2:dimY1, dimX2:dimX1);
    img = permute(img, [3, 2, 1]);

    binarized = img > threshold;
    intensity = 0;
    for k = 1:dimX1 - dimX2
        for l = 1:dimY1 - dimY2
            intensity = intensity + binarized(k, l);
        end
    end
    currentArea = intensity / ((dimX1 - dimX2) * (dimY1 - dimY2)) * 100;
end
```

D.1.2  GoToForce Function

```matlab
% Moves the motor to a location where the load cell experiences a given
% force
% 6-14-18
% Tom Pilvelait
function GoToForce(Force, Stage, loadCell, conversionFactor, zero)
    if (Force > 105)
        error('Force too high!');
    elseif (Force < 0)
        error('Force too low!');
    end

    springConstant = 6.1529;
    forceErr = 0.1;
    WithinLimit = false;
    kp = 0.8;
```
% kd=0;
% ki=0;
forces=zeros(1,1000);
times=zeros(1,1000);
curPos=zeros(1,1000);
nextPos=zeros(1,1000);

%initialize other variables for PID
% errSum=0;
measuredForce=MeasureForce(loadCell, conversionFactor, zero);
forceDifference=abs(Force-measuredForce);
%lastErr=forceDifference/springConstant;

%for troubleshooting and plotting forces
% i=1;
% j=1;
% figure;
t1=tic;
% lastTime=toc(t1);
while(~WithinLimit)
  while(IsMoving(Stage))
  end
  measuredForce=MeasureForce(loadCell, conversionFactor, zero);
  forces(i)=measuredForce;
  times(i)=toc(t1);
  curPos(i)=Stage.GetPosition_Position(0);
  i=i+1;
  scatter(times, forces, 'b');
  hold on
  %check to see if measured force is within error
  if (forceErr>abs(measuredForce−Force))
    %pause for 1ms to verify measurement
    twait=tic;
    while(toc(twait)<.001)
      end
    measuredForce=MeasureForce(loadCell, conversionFactor, zero);
    if (forceErr>abs(measuredForce−Force))
      WithinLimit=true;
      break
    end
  end
currentPosition=StageGetPosition_Position(0);

forceDifference=Force−measuredForce;
err=(forceDifference)/springConstant;

timeNow=toc(t1);
timeChange = timeNow − lastTime;
errSum = errSum+ (err * timeChange);
dErr = (err − lastErr) / timeChange;

posChange = kp*err;
% + ki * errSum + kd * dErr;
% lastErr = err;
% lastTime = toc(t1);

nextPosition=currentPosition+(posChange);

nextPos(j)=nextPosition;
% j=j+1;
%make sure the motor isn’t going somewhere crazy
if (nextPosition <20)
    break

elseif (nextPosition >130)
    break
end
MoveStage(Stage, nextPosition);
end

D.1.3 GoToArea Function

%Tom Pilvelait
%11-19-18
%Moves the motor to a certain load to obtain a corresponding area change.
%This area change must be determined by fitting a function to the area vs.
function GoToArea(newArea, currentArea, loadingCoeffs, coeffsOfAgingA, coeffsOfAgingB, Stage, loadCell, mult, zero, startTime)
%inputs the area change into the corresponding F(A) function
currentLoad=MeasureForce(loadCell, mult, zero);
[nextForce]=DetermineLoadChange(currentLoad, loadingCoeffs, coeffsOfAgingA, coeffsOfAgingB, currentArea, newArea, startTime);
GoToForce(nextForce, Stage, loadCell, mult, zero);
end

D.1.4 MeasureForce Function

function [measuredForce] = MeasureForce(lc, conversionFactor, zero)
measuredForce=conversionFactor*(mean(lc.startForeground)-zero);
end

D.2 Surface Generation

D.2.1 Generate Random Surface Function

clear all;
%close all;

map=load('hslcolormaphotcold.dat');
Lx=50;
Ly=50;
step=0.1;
X=[0:step:Lx];  %milimeters
Y=[0:step:Ly];  %milimeters
[wx,wy]=meshgrid(X,Y);
Z=0*wx.*wy;

[Nx,Ny]=size(Z);
%Z=ones(Nx,Ny);
N=Nx*Ny;
A=8;%height of bumps (mm)
offset=10;%height of sample (mm)
\[ Z = 0.125 \times \sin(\pi \times w_x \times 4) \times \sin(\pi \times w_y \times 4); \] % wavy surface, perpendicular (lattice) sines
\[ Z_1 = A \times \text{randn}(N_x, N_y); \] % gaussian $N(0,1)$ random surface
\[ \beta = 3; \]
\[ \alpha = 2; \]
\[ Z_1 = A. \times \text{surf\_loverq\_v6fun}(Z, \beta, \alpha); \]
\[ Z_1 = Z_1 + \text{offset}; \]
\[ Z_\text{mean} = \text{mean}(Z_1(:)); \]
\[ Z_\text{rms} = \text{rms}(Z_1(:)); \]
\[ [i_{\text{max}}, j_{\text{max}}] = \text{find}(Z_1 == \text{max}(Z_1(:))); \]
\%i_{\text{max}}
\%j_{\text{max}}
\%Z_1(i_{\text{max}}, j_{\text{max}})
\[ \text{pause} \]
\[ Z_{\text{negs}} = \text{sum}(Z(:) < 0); \]
\[ Z_{\text{pos}} = \text{sum}(Z(:) > 0); \]
\[ Z_{\text{zeros}} = \text{sum}(Z(:) == 0); \]
\[ l_x \_\text{buffer} = 0; \]
\[ l_y \_\text{buffer} = l_x \_\text{buffer}; \]
\[ L_{Lx} = L_x / 2 - l_x \_\text{buffer}; \]
\[ L_{Ly} = L_{Lx}; \]
\[ x_c = L_x / 2; \]
\[ y_c = x_c; \]
\[ Z = Z_1; \]
\[ n_n = \text{size}(Z); \]
\% for i = 1:n_n(1)
\% for j = 1:n_n(2)
\%
\% if ( (wx(i,j) - x_c)^2 + (wy(i,j) - y_c)^2 > L_{Lx}^2 )
\% if ( abs(wx(i,j) - x_c) > L_{Lx} || abs(wy(i,j) - y_c) > L_{Ly} )
\% Z(i,j) = 0;
\%
\% end
\%
\% end
\%
\% end
\%
\[ X_{\text{out}}, Y_{\text{out}}, Z_{\text{out}} = \text{surf2solid}(wx, wy, Z, Lx, 0.0); \]

%filename = 'solid_square_randn.stl';
filename=strcat('flat_piece_1overq',num2str(beta),'.stl');
mode = 'binary';

%smooth surface
%Zsmooth=smoothdata(Z_out);

%for i=1:2
%Zsmooth=smoothdata(Zsmooth);
%end

%add edge buffer to z; assumes square sample
buff=6/step; %10 mm total added; 5 to each side
Zbuff=zeros(size(Z_out)+buff);
Ybuff=zeros(size(Y_out)+buff);
Xbuff=zeros(size(X_out)+buff);

%offset and center x_out in xbuff
for i=2:length(Xbuff)-1
  for j=2:length(Xbuff)-1
    if (i<5)
      Xbuff(i,j)=X_out(2,2)-buff/2*step;
    elseif (i>length(Xbuff)-4)
      Xbuff(i,j)=X_out(2,length(X_out)-1)+buff/2*step;
    else
      Xbuff(i,j)=Xbuff(i-1,j)+step;
    end
  end
end
Xbuff=Xbuff';

%offset and center y_out in ybuff
for i=2:length(Ybuff)-1
  for j=2:length(Ybuff)-1
    if (i<5)
      Ybuff(i,j)=Y_out(2,2)-buff/2*step;
    elseif (i>length(Ybuff)-4)
      Ybuff(i,j)=Y_out(length(Y_out)-1,2)+buff/2*step;
    else
      Ybuff(i,j)=Ybuff(i-1,j)+step;
    end
  end
end
Ybuff=Ybuff';
%create outer buffer
for i = 3:length(Zbuff) − 2
    for j = 3:length(Zbuff) − 2
        Zbuff(i, j) = 10;
    end
end

%add 5mm offset to z_out heights
for i = 3:length(Z_out) − 2
    for j = 3:length(Z_out) − 2
        Zbuff(buff/2+i, buff/2+j) = 5;
    end
end

%place z_out heights centered in zbuff
for i = 3:length(Z_out) − 2−10/step
    for j = 3:length(Z_out) − 2−10/step
        Zbuff(buff/2+5/step+i, buff/2+5/step+j) = Z_out(i, j) + 2;
        if (Zbuff(buff/2+5/step+i, buff/2+5/step+j) > 5)
            Zbuff(buff/2+5/step+i, buff/2+5/step+j) = 5;
        end
    end
end

Zbuff = abs(Zbuff);

%add a defect to differentiate parts
for i = 1:5
    for j = 1:5
        Zbuff(20+i, 20+j) = 12;
        Zbuff(547−i, 547−i) = 12;
        Zbuff(547−i, 20+j) = 12;
    end
end

surf2stl(filename, Xbuff, Ybuff, Zbuff, mode);

fdim1 = 1000;
fdim2 = 500;
fig2 = figure('position',[0 0 fdim1 fdim2]);

subplot(1,2,1)
%hb = pcolor(Z1);
hb = mesh(X,Y,Z1);
%set(hb,'EdgeColor','none');
title('random surface');
colorbar('location','EastOutside'); %colormap(map);
axis equal; %('square');
set(gca,'XTick',[0:2:Lx]);
set(gca,'YTick',[0:2:Ly]);
%Z1_max = ceil(max(abs(Z1)));
%Z1_min = -Z1_max;
%zlim([-Z1_max Z1_max]);
%set(gca,'ZTick',[-Z1_max:0.5:Z1_max]);

subplot(1,2,2)
histnorm(Z1(:),100,1);
title('Histogram of Z1'); xlabel('Z1'); ylabel('pdf');
title(strcat('random surface with 1/q^ ',num2str(beta), ' power spectrum'));
%savefig(fig2,strcat(strcat('surface_1overq',num2str(beta),'.fig')));

fd1 = 500;
fd2 = fd1;

fig1 = figure('position',[0 0 fd1 fd2]);
%ha = pcolor(Z1);
ha = mesh(Xbuff,Ybuff,Zbuff);
%set(ha,'EdgeColor','none');
title('stl mold');
colorbar('location','EastOutside'); %colormap(map);
axis equal; %('square');
set(gca,'XTick',[min(Xbuff(:)):2:max(Xbuff(:))]);
set(gca,'YTick',[min(Ybuff(:)):2:max(Ybuff(:))]);
title(strcat('stl mold, random surface with 1/q^ ',num2str(beta), ' power spectrum'));
%savefig(fig1,strcat(strcat('stlmold_1overq',num2str(beta),'.fig')));

fig3 = figure('position',[0 0 fd1 fd2]);
%hc = pcolor(Z);
hc = mesh(X,Y,Z);
%set(hc,'EdgeColor','none');
title('mold of surface');
colorbar('location','EastOutside'); %colormap(map);
axis equal; 
set(gca,'XTick',[0:2:Lx]);
set(gca,'YTick',[0:2:Ly]);
zlim([0 10]);
set(gca,'ZTick',[0:2:10]);
title(strcat('solid premold, random surface with 1/q^ ',num2str(beta),'
power spectrum'));
%savefig(fig3,strcat(strcat('premold_1overq',num2str(beta),'.fig')));

% Count the number of bumps
regMax=imregionalmax(Z_out);
NumberOfbumps=nnz(regMax==1)

D.2.2 Generate Contact Height Distribution Function

function V=surf_1overqb_v6fun(M, beta , alpha)

%map=load('hslcolorphotcold.dat');

% Nx=512;
% Ny=Nx;
% beta=0.0;
% alpha=1000;
[Nx Ny]=size(M);

M=ones(Nx,Ny);
sizeM=size(M)

for i=1:Nx
    for j=1:Ny
        M(i,j)=sqrt((i-Nx/2)*(i-Nx/2) + (j-Ny/2)*(j-Ny/2))+1;
    end
end
sizeM=size(M)

%pause

R=randn(Nx,Ny);
I=randn(Nx,Ny);
sizeR=size(R)
sizeI=size(I)
R=alpha*R.*M.^(beta/2);
%R
I=alpha*I.*M.^(beta/2);
%I
C=complex(I,R);
sizeC=size(C)
C = i fft2 (C) ;
V = abs (C) ;
Vmax = max (V(:,)) ;
V = V ./ Vmax ;

\[ \text{meanV} = \text{mean} (V(:,)) ; \]
\[ \text{V} = \text{V} - \text{meanV} ; \]
S = fft2 (V) ;
S = fftshift (S) ;
S = (abs (S)) .^ 2 ;
sizeV = size (V) ;
[S_rad, qr] = raPsd2d_fun (V, 1) ;
Lr = ((alpha / 2) ^ 2) * (qr) .^ ( - beta ) ;
Lr = max (S_rad (:)) / Lr (1) .* Lr ;

% pause
% Lx = ((alpha / 2) ^ 2) * fftshift (A(256,:)) .^ ( - beta ) ;
qx = 1:1:Nx ;
Lx = ((alpha / 2) ^ 2) * (qx) .^ ( - beta ) ;

fd1 = 1000 ;
fd2 = fd1 ;
fig1 = figure (' position ', [0 0 fd1 fd2]) ;

subplot (2, 3, 1) ;
mesh (V) ;
title ('Signal V(x,y)') ;
xlabel ('x') ; ylabel ('y') ; zlabel ('V(x,y)') ;
axis ('square') ;

subplot (2, 3, 2) ;
histnorm (V(:,), 100, 1) ;
title ('Histogram of V') ;
xlabel ('V') ; ylabel ('pdf') ;

subplot (2, 3, 3) ;
NNx = floor (Nx / 2) ;
plot (V(NNx, :) , 'r') ;
title (strcat ('Signal V(x=' , num2str (NNx) , ',y)')) ;
xlabel ('x') ; ylabel (strcat ('V(x=' , num2str (NNx) , ',y)')) ;
axis ('square') ;

subplot (2, 3, 4) ;
mesh (log (S)) ;
title ('PowerSpectrum LOG[ S(qx,qy) ]') ;
xlabel ('qx') ; ylabel ('qy') ; zlabel ('LOG[ S(qx,qy) ]') ;
axis ('square') ;

62
D.2.3 Transmute Surface into Solid Function

function [x_out, y_out, z_out] = surf2solid(x,y,z,S,delta)
%
% file: surf2solid.m, (c) Matthew Roughan, Mon Feb 14 2011
% directory: /home/mroughan/src/matlab/STL/
% created: Mon Feb 14 2011
% author: Matthew Roughan
% email: matthew.roughan@adelaide.edu.au
%
% Convert a surface into a solid object for printing on a 3D printer
% This routine only works for surfaces defined on a rectangular
% grid. The matrices X and Y define the axis limits only.
%
% The assumption is that
%   the surface is a function z(x,y)
%   the input are given in triples (x,y, z(x,y)) similar to the
%   matrix input option
% i.e., x, and y specify regular, rectangular lattice points

% subplot(2,3,5);
loglog(qr, S_rad, '-o');
hold on;
loglog(qr, Lr, '-k', 'LineWidth', 1);
hold off;
legend('S_rad', strcat('slope=', num2str(beta)));
title('PowerSpectrum LOGLOG[S_rad]');
xlabel('qr'); ylabel('S_rad'); axis('square'); %colormap(map);

subplot(2,3,6);
loglog((fftshift(S(NNx,:))));
hold on;
loglog(Lx, '-k', 'LineWidth', 2);
hold off;
legend(strcat('S(qx=', num2str(NNx), ',qy)'), strcat('slope=', num2str(beta)));
title(strcat('PowerSpectrum LOGLOG[S(qx=', num2str(NNx), ',qy)]'));
xlabel('qx'); ylabel(strcat('S(qx=', num2str(NNx), ',qy)'));
axis('square'); %colormap(map);

title(strcat('random surface with 1/q^', num2str(beta), ' power spectrum'));
savefig(fig1, strcat('1_over_q', num2str(beta), '.fig'));
 fout11 = sprintf(strcat('profile_1overq', num2str(beta), '.dat'));
dlmwrite(fout11, V, 'delimiter', ' ')

% D.2.3 Transmute Surface into Solid Function

% Convert a surface into a solid object for printing on a 3D printer
% This routine only works for surfaces defined on a rectangular
% grid. The matrices X and Y define the axis limits only.
% The assumption is that
%   the surface is a function z(x,y)
%   the input are given in triples (x,y, z(x,y)) similar to the
%   matrix input option
% i.e., x, and y specify regular, rectangular lattice points

% Function [x_out, y_out, z_out] = surf2solid(x,y,z,S,delta)
% % file: surf2solid.m, (c) Matthew Roughan, Mon Feb 14 2011
% % directory: /home/mroughan/src/matlab/STL/
% % created: Mon Feb 14 2011
% % author: Matthew Roughan
% % email: matthew.roughan@adelaide.edu.au
% % Convert a surface into a solid object for printing on a 3D printer
% % This routine only works for surfaces defined on a rectangular
% % grid. The matrices X and Y define the axis limits only.
% % The assumption is that
% %   the surface is a function z(x,y)
% %   the input are given in triples (x,y, z(x,y)) similar to the
% %   matrix input option
% % i.e., x, and y specify regular, rectangular lattice points

% subplot(2,3,5);
loglog(qr, S_rad, '-o');
hold on;
loglog(qr, Lr, '-k', 'LineWidth', 1);
hold off;
legend('S_rad', strcat('slope=', num2str(beta)));
title('PowerSpectrum LOGLOG[S_rad]');
xlabel('qr'); ylabel('S_rad'); axis('square'); %colormap(map);

subplot(2,3,6);
loglog((fftshift(S(NNx,:))));
hold on;
loglog(Lx, '-k', 'LineWidth', 2);
hold off;
legend(strcat('S(qx=', num2str(NNx), ',qy)'), strcat('slope=', num2str(beta)));
title(strcat('PowerSpectrum LOGLOG[S(qx=', num2str(NNx), ',qy)]'));
xlabel('qx'); ylabel(strcat('S(qx=', num2str(NNx), ',qy)'));
axis('square'); %colormap(map);

title(strcat('random surface with 1/q^', num2str(beta), ' power spectrum'));
savefig(fig1, strcat('1_over_q', num2str(beta), '.fig'));
 fout11 = sprintf(strcat('profile_1overq', num2str(beta), '.dat'));
dlmwrite(fout11, V, 'delimiter', ' ')

function [x_out, y_out, z_out] = surf2solid(x,y,z,S,delta)
% % Convert a surface into a solid object for printing on a 3D printer
% % This routine only works for surfaces defined on a rectangular
% % grid. The matrices X and Y define the axis limits only.
% % The assumption is that
% %   the surface is a function z(x,y)
% %   the input are given in triples (x,y, z(x,y)) similar to the
% %   matrix input option
% % i.e., x, and y specify regular, rectangular lattice points

% subplot(2,3,5);
loglog(qr, S_rad, '-o');
hold on;
loglog(qr, Lr, '-k', 'LineWidth', 1);
hold off;
legend('S_rad', strcat('slope=', num2str(beta)));
title('PowerSpectrum LOGLOG[S_rad]');
xlabel('qr'); ylabel('S_rad'); axis('square'); %colormap(map);

subplot(2,3,6);
loglog((fftshift(S(NNx,:))));
hold on;
loglog(Lx, '-k', 'LineWidth', 2);
hold off;
legend(strcat('S(qx=', num2str(NNx), ',qy)'), strcat('slope=', num2str(beta)));
title(strcat('PowerSpectrum LOGLOG[S(qx=', num2str(NNx), ',qy)]'));
xlabel('qx'); ylabel(strcat('S(qx=', num2str(NNx), ',qy)'));
axis('square'); %colormap(map);

title(strcat('random surface with 1/q^', num2str(beta), ' power spectrum'));
savefig(fig1, strcat('1_over_q', num2str(beta), '.fig'));
 fout11 = sprintf(strcat('profile_1overq', num2str(beta), '.dat'));
dlmwrite(fout11, V, 'delimiter', ' ')
but note that this doesn’t support the 1, or 2 argument versions of these commands
and assumes that x and y are matrices, not vectors
— x values increase along rows
— y values along columns, e.g., as if they are the outputs of
\[ [x,y] = \text{meshgrid}(1:3, 1:4) \]

The output will be a solid with height given by z at each point \((x, y)\), and
— the surface \(z\) will be shifted by \(z_0\), such that \(z+z_0 \geq \delta\), where \(\delta\) is the minimum \(z\) height
— the \((x,y)\) co-ordinates will be shifted so that \((x+x_0, y+y_0)\) is "centered"
— the \(x, y,\) and \(z\) values will be scaled so that the largest value of
\[
\max(x) - \min(x), \max(y) - \min(y), \text{ and } 2\times(\max(z) - \min(z))
\]
is such that the largest of these dimensions is size \(S\) where \(S\) in measured in mm
(NB: matlab co-ordinates come out roughly as mm in the makerbot)
— the surface \(z(x,y)+z_0\) will have vertical walls going down to the \((x,y)\) plane,
and the bottom surface of the object will be the part of the \((x,y)\) plane including
the \((x,y)\) points of the input (suitable rescaled)
(e.g. see what meshz does)

NB: fair bit of code is inspired by meshz

err_id = 'surf2solid:InvalidInput';

% check inputs
if (nargin < 3)
    error(err_id , 'Need to input at least 3 arguments . ')
end
if ischar(x) || ischar(y) || ischar(z)
    error(err_id , 'Input should not be characters . ')
end
[m,n] = size(z);
[mx,nx] = size(x);
[my,ny] = size(y);
if (m == 1 || n == 1)
    error(err_id , 'Arrays must have size >1 in each direction.' )
end
if (~isequal(size(z),size(x)) || ~isequal(size(z),size(y)))
    error(err_id , 'Input arrays \((x,y,z)\) should all be the same size.')
end
if (~all(isfinite(z)))
error (err_id, 'z must have finite values.')
end
if (~ all(isfinite(y)))
  error (err_id, 'y must have finite values.')
end
if (~ all(isfinite(x)))
  error (err_id, 'x must have finite values.')
end
if (nargin < 4)
  S = 40; % make the default max length along an axis 40mm
end
if (S>100)
  error (err_id, 'S must be <= 100 due to size of build platform')
end
if (nargin < 5)
  delta = 3; % make the default minimum height 3 mm
end
if (delta < 0)
  error (err_id, 'delta must be >=0 to build')
end

% scale co-ordinates
x_diff = max(max(x)) - min(min(x));
y_diff = max(max(y)) - min(min(y));
z_diff = 2*(max(max(z)) - min(min(z)));
max_diff = max([x_diff; y_diff; z_diff]);
scale_factor = S/max_diff
x_s = scale_factor * x;
y_s = scale_factor * y;
z_s = scale_factor * z;
mx_x = max(max(x_s));
mx_y = max(max(y_s));
mx_z = max(max(z_s));

% shift the min z co-ordinate
z = z_s + delta - mn_z;

% center the (x,y) co-ordinates
x = x_s - (mx_x + mn_x)/2;
y = y_s - (mx_y + mn_y)/2;
xmin = min(min(x));
ymin = min(min(y));
xmax = max(max(x));
ymax = max(max(y));
% create an extended Z matrix
z_out = [
[ 0 0 zeros(1,n) 0 0 ];
[ 0 0 z(1,:) 0 0 ];
[ zeros(m,1) z(:,1) z(:,n) zeros(m,1) ];
[ 0 0 z(m,:) 0 0 ];
[ 0 0 zeros(1,n) 0 0 ];
];

% extend the x and y arrays
d = [1 1];
mm = [m m];
nn = [n n];
x_out = [
[ x(d,d) x(d,:) x(d,nn) ];
[ x(:,d) x x(:,nn) ];
[ x(mm,d) x(mm,:) x(mm,nn) ];
];
x_out = zero_pad(x_out, 2);

y_out = zero_pad(y_out, 1);

% fd1=800;
% fd2=fd1;
% figg=figure ( 'position' , [0 0 fd1 fd2] );
% map=load ( 'hslcolormaphotcold.dat' );
% mesh(x_out, y_out, z_out);
% title ( 'stl mold' );
% colorbar ( 'location', 'EastOutside' ); colormap(map);
% axis equal; %('square') ;
% set(gca, 'XTick', [min(x_out(:)) : 2 : max(x_out(:))] );
% set(gca, 'YTick', [min(y_out(:)) : 2 : max(y_out(:))] );

function B = zero_pad(A, k)

% pad zeros around entire outside of array to width k

[mA,nA] = size(A);
B = [ zeros(k,k) zeros(k,nA) zeros(k,k) ];
D.2.4 Transmute Solid into STL File Function

```matlab
function surf2stl(filename, x, y, z, mode)
%SURF2STL    Write STL file from surface data.
% SURF2STL( 'filename' ,X,Y,Z) writes a stereolithography (STL) file
% for a surface with geometry defined by three matrix arguments , X, Y
% and Z. X, Y and Z must be two–dimensional arrays with the same
% size.
%
% SURF2STL( 'filename' ,x,y,Z) , uses two vector arguments replacing
% the first two matrix arguments , which must have length(x) = n and
% length(y) = m where [m,n] = size(Z). Note that x corresponds to
% the columns of Z and y corresponds to the rows.
%
% SURF2STL( 'filename' ,dx,dy,Z) uses scalar values of dx and dy to
% specify the x and y spacing between grid points.
%
% SURF2STL(... , 'mode') may be used to specify the output format.
% 'binary' – writes in STL binary format (default)
% 'ascii' – writes in STL ASCII format
%
% Example:
% surf2stl( 'test.stl' ,1,1,peaks) ;
% See also SURF.
%
% Author: Bill McDonald, 02–20–04

error(nargchk(4,5,nargin));
if (ischar(filename)==0)
    error( 'Invalid filename' );
end
if (nargin < 5)
    mode = 'binary' ;
elseif (strcmp(mode,'ascii')==0)
    mode = 'binary' ;
end
if (ndims(z) ~= 2)
    error( 'Variable z must be a 2–dimensional array' ) ;
end
```
if any( (size(x)==size(z)) | (size(y)==size(z)) )

% size of x or y does not match size of z

if ( (length(x)==1) & (length(y)==1) )
    % Must be specifying dx and dy, so make vectors
    dx = x;
    dy = y;
    x = ((1:size(z,2))−1)*dx;
    y = ((1:size(z,1))−1)*dy;
end

if ( (length(x)==size(z,2)) & (length(y)==size(z,1)) )
    % Must be specifying vectors
    xvec=x;
    yvec=y;
    [x,y]=meshgrid(xvec,yvec);
else
    error('Unable to resolve x and y variables');
end

end

if strcmp(mode, 'ascii')
    % Open for writing in ascii mode
    fid = fopen(filename,'w');
else
    % Open for writing in binary mode
    fid = fopen(filename,'wb+');
end

if (fid == −1)
    error( sprintf('Unable to write to %s',filename) );
end

title_str = sprintf('Created by surf2stl.m %s', datestr(now));

if strcmp(mode, 'ascii')
    fprintf(fid, 'solid %s\n', title_str);
else
    str = sprintf('%80s',title_str);
    fwrite(fid, str, 'uchar'); % Title
    fwrite(fid,0,'int32'); % Number of facets, zero for now
end

nfacets = 0;

for i=1:(size(z,1)−1)
    for j=1:(size(z,2)−1)
p1 = [x(i,j)  y(i,j)  z(i,j)];
p2 = [x(i,j+1) y(i,j+1) z(i,j+1)];
p3 = [x(i+1,j+1) y(i+1,j+1) z(i+1,j+1)];
val = local_write_facet(fid,p1,p2,p3,mode);
nfacets = nfacets + val;

p1 = [x(i+1,j+1) y(i+1,j+1) z(i+1,j+1)];
p2 = [x(i+1,j)  y(i+1,j)  z(i+1,j)];
p3 = [x(i,j)   y(i,j)   z(i,j)];
val = local_write_facet(fid,p1,p2,p3,mode);
nfacets = nfacets + val;

end
end

if strcmp(mode, 'ascii')
  fprintf(fid,'endsolid %s\r\n',title_str);
else
  fseek(fid,0,'bof');
  fseek(fid,80,'bof');
  fwrite(fid,nfacets,'int32');
end
fclose(fid);
disp(sprintf('Wrote %d facets'),nfacets));

% Local subfunctions

function num = local_write_facet(fid,p1,p2,p3,mode)
if any(isnan(p1) | isnan(p2) | isnan(p3))
  num = 0;
  return;
else
  num = 1;
  n = local_find_normal(p1,p2,p3);
  if strcmp(mode, 'ascii')
    fprintf(fid,'facet normal %.7E %.7E %.7E\r\n', n(1),n(2),n(3));
    fprintf(fid,'outer loop\r\n');
    fprintf(fid,'vertex %.7E %.7E %.7E\r\n', p1);
    fprintf(fid,'vertex %.7E %.7E %.7E\r\n', p2);
    fprintf(fid,'vertex %.7E %.7E %.7E\r\n', p3);
    fprintf(fid,'endloop\r\n');
    fprintf(fid,'endfacet\r\n');
  end
end
else
    fwrite ( fid , n , ’float32’ ) ;
    fwrite ( fid , p1 , ’float32’ ) ;
    fwrite ( fid , p2 , ’float32’ ) ;
    fwrite ( fid , p3 , ’float32’ ) ;
    fwrite ( fid , 0 , ’int16’ ) ; % unused
end
end

function n = local_find_normal(p1,p2,p3)
    v1 = p2−p1;
    v2 = p3−p1;
    v3 = cross (v1 ,v2) ;
    n = v3 ./ sqrt (sum(v3.*v3)) ;

D.3 Miscellaneous Scripts

D.3.1 Script used to calculate image drift

function [ xDrift , yDrift , semoas]=CalcImageDrift(image1 , image2)
    xDrift=0;
    yDrift=0;
    border=10;
    dimX=length(image1 (: ,1) ) ;
    dimY=length(image1 (1 ,:) ) ;
    semoas=zeros(2*border ,2*border) ;
    trimmedImage1=image1(border+1:dimX−border , border+1:dimY−border) ;
    %make the minimum some arbitrarily large numberi——useful for trouble
    %shooting
    minSEMOA=10^9 ;
    %cycle the image through the x and y directions , see where SEMOA is
    %minimized
    for a=1:2*border
        for b=1:2*border
            %SEMOA for trimmed and shifted 1st image and corresponding
            section
\texttt{SEMOA}=\texttt{CalcSEMOA(trimmedImage1, image2(a:dimX-2*border+a-1,b:dimY-2*border+b-1))};
\texttt{semoas(a,b)=SEMOA;}
\texttt{if (SEMOA<minSEMOA)}
\hspace{1em} \texttt{%records dift (needs to be recentered)}
\hspace{1em} \texttt{xDrift=a-1-border;}
\hspace{1em} \texttt{yDrift=b-1-border;}
\hspace{1em} \texttt{minSEMOA=SEMOA;}
\texttt{end}
\texttt{end}
\texttt{end}