ABSTRACT

Applied IMU Image Registration for Super Resolution on Mobile Devices

Tyler Hartwig, M.S.

Mentor: Keith Evan Schubert, Ph.D.

Mobile phones and cameras are incredibly popular and the hardware is becoming very impressive. The photos these devices are currently able to take are already of high quality, however it is possible to improve these cameras in software. It is possible to take burst-shot photos and utilize the phase offsets to realize a higher resolution signal. While this takes a large amount of computation, it is possible to reduce that computation by using the IMU devices on mobile phones. This research explores that idea, as well as investigating what signal resolving algorithms produce the highest quality image in combination with the IMU data. IMU sensors are shown to help in reducing the time it takes to register photos to sub-pixel accuracy. It is also shown that the results by using the sensors are comparable to not using these sensors.

Applied IMU Image Registration for Super Resolution on Mobile Devices

by

Tyler Hartwig, B.S.

A Thesis

Approved by the Department of Electrical and Computer Engineering

---

Kwang Y. Lee, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of
Master of Science

Approved by the Thesis Committee

Keith Evan Schubert, Ph.D., Chairperson

Robert Marks, Ph.D.

Byron, Newberry, Ph.D.

Accepted by the Graduate School
May 2017

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE

Introduction

Personal mobile devices have become complete replacements for various other electronic devices recently. In particular, modern smartphones have almost completely eliminated the need to own a point-and-shoot camera. The hardware supporting these cameras is constantly improving, however not much is done to dramatically improve capture ability in software.

1.1 Background

"Super resolution" refers to improving the resolution of a signal, by sampling the signal multiple times, each with a phase difference, and then extracting a higher resolution signal from these samples. This method obviously only works if you have the ability to sample the same, or similar signal multiple times, with some phase offset. For the purposes of mobile devices this occurs with some frequency. People often take still-motion pictures, meaning in these instances it is fairly trivial to take multiple samples of the same, or similar signal (Park et al. 2003).

1.1.1 Camera Phase Offset

Fortunately, phase offset is naturally achieved when capturing a photo with a phone, or digital camera. When one snaps a photo, a small amount of sway and movement exists as it is nearly impossible for a person to stand completely motionless. If this motion were not to be enough, or if the phone or camera were to be held on a tripod, it is also possible to induce some phase offset during capture. Conveniently, nearly every modern mobile device has a vibration unit built in, which can be used to create some small phase offset.

1.1.2 Measuring Phase Difference

Phase offset between samples is useless in this application unless the difference can be measured. Here in lies the challenge of accomplishing self-contained super resolution on a mobile phone. The challenge is not the need for an algorithm that can measure the difference, but rather one that is computationally efficient and feasible for a mobile device to use. These algorithms are computationally heavy as they need to estimate both rotational shifts, as well as vertical and horizontal shifts. Again modern mobile devices conveniently have other measurement devices in them as well. In particular, the accelerator and gyroscope in these devices can help to determine these shifts.

1.2 Software Selection

This project used various different phones from volunteers. Because of this design, all data was uploaded to an Azure data system. This allowed for easy development of the mobile app, without a need for server code as well. Azure also integrated very well with the tooling chosen for the mobile application, Xamarin. This tooling was used both for previous familiarity, as well as its ability to let one share code between both iOS and Android versions of the app.

CHAPTER TWO

Related Work

2.1 Device Sensors for Image Processing on Mobile Devices

Scholarly literature exists for utilizing device sensors for image processing applications on mobile devices. In particular Saragadam R V Vishwanath conducted research on using these sensors for blur-correction, image registration and depth estimation. Of specific interest to this research is the image registration, as it is the first step in the super resolution process.

2.1.1 Sensor-Based Registration

Digital image registration (alignment) is a well known problem and many solutions already exist. The existing algorithms pose a problem, as they are not computationally efficient enough for a mobile device to compute them quickly. Given this, a faster approach to image registration is needed, in particular, motion sensors on mobile phones can be used to eliminate or speed up some of these algorithms.

Vishwanath's research (Patrick Vandewalle and Vetterli 2005) in relation to image registration was to use solely an accelerometer. With this limitation, only rotational shifts or directional shifts were estimated, but not both for a single image. The three-axis accelerometer cannot be used to accurately estimate both of these shifts, but rather another sensor, such as a gyroscope, would need to be combined with it to obtain extra information. This research restricted the movement of the phone to be purely horizontal and vertical shifts, or rotational shifts. Additionally this research focused on using these methods to acquire image registration results on highly blurred images.

The research was able to accurately and quickly register images using these data readings by dramatically reducing the registration search space. For pure translation estimation, the accelerometer values are summed to find the distance traveled. However, this distance cannot be directly translated to pixels, as the distance from the camera to the object is not known. Additionally, when considering a wide variety of phones, different cameras and different sensors would add in additional variables, increasing the difficulty of converting this distance to pixels. This information is shown to not be useless however, as the distances tell the direction the shift occurred in. The search space for the shift of the image is dramatically reduced to a single dimension, rather than than in 2 dimensions. This concept can be applied in mobile devices to quickly and efficiently acquire an accurate sub-pixel registration.

Similarly, the accelerometer was also used to estimate an image's rotational shift. In this case, the acceleration was used directly to determine the current angle of the device (assuming gravity is the only force acting on the phone). The angles between photos could then be determined, and thus corrected as necessary. The paper shows that these methods work fairly well for blurred images, and an approximate registration is found for each rotation. Translation shifts are also estimated decently, however no explicit proof is given to support the registration being able to achieve sub-pixel registration.

2.1.2 Sensor-Based Motion Blur Correction

This research dives into another very useful concept. Motion blur is very easy to correct if the blur angle is known. Since the research presented in this paper highly relies on at least small motion in between photo captures, it is somewhat likely that some amount of motion blur occurs. Correcting this blur will be necessary in those cases, and accurate angle prediction will be critical to implementing a blur correction well.

Image blur can be modeled as a single kernel (generally a 3x3 matrix) convoluted with the entire original image. If the kernel is known, de-convolution can be applied, and the original image can be restored. Using the sensor information, it is easy to accurately estimate the direction of motion blur, leaving only the distance being left unknown. The kernel can be determined well from this information, and de-convolution can be used to remove blur from the image.

As an example the research presents an extremely blurred image. Several known algorithms are used on this image and some of the results are presented. Out of the results presented, the kernel estimated from sensor data aids the most in resolving the original image. It is nowhere close to restoring the original image; however, many more of the original details of the image can be seen.

Motion blur will most likely commonly occur in real applications of the research presented in this model, making the ability to accurately correct this issue of importance to this research.

2.1.3 Sensor-Based Depth Estimation

One of the final things this article explores is estimating the various depths of an image using multiple captures. Due to the nature of how cameras are generally built, they can only truly focus on one depth at a time. Often the human eye does not notice the things slightly out of focus from the focal length; however, objects that are far from the focal length are visibly blurred.

Depth maps of the photos can be generated from this information if multiple images can be compared against each other. Two main methods exist, one based on the blur in the image and the other based on the focus of the image. The blur technique uses one focused picture, and one unfocused picture. These pictures are compared, and the radius of the blur is compared between both images. The different sizes of blur indicate the depth of a specific object in the photo, and a depth map

can easily be created from this comparison. This method did not work very well, and obvious error can be seen in the depth map.

The focus method of creating the same depth map is slightly more complicated. Many photos are taken in this scheme (approximately a hundred in this case) all at different focal lengths. Each part of the image can then be analyzed for sharpness and assigned a distance value, since the relative focal length is known. Compared to the blur method, this one works extremely well. Objects in the depth map can be clearly distinguished, and relative depths are established well.

No sensors were mentioned in the previous outlines, as one small detail was left out. Before doing either the focal or blur comparison, the images must be registered with each other. If the images are not well registered, then the focus and blur changes expected are useless.

Theoretically, with proper calibration these methods could be used to correlate position estimations from the accelerometer with actual pixel shift values, based on the distance of the camera from the target. These methods provide interesting ideas into creating efficient higher-level image processing algorithms on mobile devices. While the depth estimation does not currently apply directly to mobile super resolution research, it is possible that it could have significant impact in the future.

2.2 Algorithmic Registration

Patrick Vandewalle, Sabine Süsstrunk, and Martin Vetterli have also published very important literature relating to super-resolution. These three made contributions improving the speed of image registration algorithms which do not utilize sensor data. First, it is shown that the rotational element of the shifted image to a references image can be evaluated in the frequency domain. Rotations in the spatial domain are equivalent to an equal rotation in magnitude of the Fourier transform of the image. This is critical, as it allows for the rotation of the image to be determined independent

of any translation shifts. It is difficult in the spatial domain to estimate rotations accurately if translations are also affecting the signal.

This research then presents an efficient algorithm for estimating rotation given the magnitude of the Fourier transform of the image. The naive way of doing this would be to iteratively rotate one of the images, until a maximum correlation is found between the magnitudes. This clearly would be computationally heavy and not very efficient. Another approach is to transform the magnitude from a rectangular representation to polar one, Fourier transform this image yet again and finally divide these images by each other to find the translational shift. This shift will equate to some rotation in the original spatial domain.

Instead of taking one of these approaches, the paper reduces the search to one dimension. To do this, a new function is defined $h(\alpha)$ which is a double integral over a small sliver of the magnitude of the transform. Only a disk of the magnitude is used for this, the radius being the largest radius still contained in the magnitude image. Additionally a tenth of the radius is also cut out of the center, due to the low frequencies containing much energy and not being able to be sampled well. A discrete function $h(\alpha)$ is created for each image, and then shifted and the maximum correlation is found. This method is fairly efficient, and allows for precise rotation to be determined.

Once the rotations are known, the translational shift differences are easy to approximate. Similarly to how the magnitude of the Fourier transform helps with finding rotational shift, the phase of the transform helps in finding the translational shifts. This algorithm is well known, but will be explained for completeness. With the rotational shifts known, the spatial images are corrected for rotational differences, so that the shifts may be acquired accurately. After the images are corrected for rotation, the phases of the transforms are divided by each other. When this signal is brought back to the spatial domain, it's values directly indicate the translational shifts in the

original spatial images. Additionally, to find sub-pixel resolution with this method, a least squares solution is used to find the location of what should be the brightest point in the picture.

2.3 Single Image Super-Resolution

Rapid and Accurate Image Resolution (RAISR) is a very interesting method in a related area of research (Romano et al. 2016). Single Image Super-Resolution (SISR) is similar to the goal of the research found in this Thesis; however, it uses vastly different methods. There are many known was of increasing the resolution of a single image. The simplest of these methods is generally some sort of interpolation function such as nearest-neighbor, bilinear interpolation or bicubic interpolation.

RAISR in particular uses machine learning to accomplish this. The basic idea is to solve the $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}$ problem where A is a blurring filter applied to a high resolution photo $\boldsymbol{x}$ to result in a low resolution photo $\boldsymbol{b}$. Low resolution photos can be generated from higher resolution photos for the learning aspect of this application. This $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ problem is applied only to patches of these images, so as to learn various patterns, not images. RAISR additionally uses a quick and efficient hashing mechanism to match a given image patch the appropriate texture type, allowing RAISR to learn and use many different filters.

This method is clearly extremely advanced and intricate as well as powerful. The methods used are entirely different than the research presented here, which is advantageous. If one truly needed to acquire a very high resolution image from a low resolution camera, it is possible to use both methods and achieve a higher resolution photo than either one could do alone. The RAISR concept has the potential to take the image capabilities of modern mobile cameras even further beyond what this research presents.

CHAPTER THREE

Design and Implementation

The system design for this research focuses on making the data easy to collect and easy to test. Part of the experiment involves using many phones to collect data. The most cost-effective way to accomplish this is to use volunteers' phones on campus. This motivates the need for making the data relatively easy to collect. Additionally, the data being collected potentially could be used for other projects, which motivates making the collected data easy to access and test.

Experiment trials also need to be consistent and quick, in addition to considering and respecting the privacy of the volunteer's phone. These concerns have a significant impact on the design of the experiment and data collection as a whole.

3.1 Data Storage

3.1.1 Data Storage Design

Data storage design focuses mostly on easy access to the data. A vast number of storage formats and strategies exist, however they can be narrowed down to a few categories. Generally data are either stored locally on storage media, or can be stored on a remote server to allow more universal access. Another consideration is how the data are stored, whether to use a simple standard data format, or to utilize a database for more structured storage. Finally, storing images can also be a challenge, due to the large number of images being captured, and the need to be able to easily correlate the associated sensor data for each image.

Due to the nature of how the mobile applications are generally designed, and the nature of the experiment, keeping the data online is the obvious choice. Storing the data on a server allows for the data to never even be written to permanent storage

on the phone. No data ever needs to be written to the phone's storage media and the data collector does not need to search through a stranger's phone's data in order to collect the data. Additionally, it is very natural and common for mobile applications to store and transport data over the internet.

The way in which the data is stored is slightly more complicated. Since a large amount of data is collected from each phone, and each phone trial consists of various types of tests, a database is used to store all sensor data and phone information. This allows the data to be easily searched and queried for specific sensor readings. A simple schema was chosen for the database, making it easy to create and still easily queried.

A simple approach was taken in designing the database for this data. To begin with, two tables are used for general information about the device. One table is used to hold all data concerning the make, model, and version of the phone, along with a unique identifier for the device. Additionally, each phone is weighed during testing and its data are also recorded in this table. While iPhones tend to have a fixed set of sensors, Android phones often have a variety of different types of sensors and many times different brands for each phone. Because of the variety that may be encountered, a sensor table is used to store any relevant information about each phone. In this table the sensor's name, the device it belongs to, and a unique identifier are simply stored.

The experiment conducts five different types of tests. It is critical to be able to correlate specific data readings to specific tests. Each test is considered a "session" and a table is dedicated to keeping track of all sessions. This table contains a unique identifier for the session, the device that ran the session, and the type of session that it ran. To simplify the database design, sensors were generalized to one standard data model. For the purposes of this experiment, only positional sensors are used, which means all sensor readings consist of an x, y, and z value in addition to a timestamp.

Therefore all sensor readings are stored in one table. Each entry consists of the sensor identifier for the reading (and it's name, for convenience), the session associated with the reading, the x, y, and z data values, and a timestamp.

Unfortunately, storing images in a database does not make much sense. Instead, a blob storage container is used for all images. Blob storage is able to organize the files uploaded in a file hierarchy. Using blob storage and an appropriate file structure allows all images to be quickly found and downloaded. The structure used in this application starts by creating a folder for each type of session that captures an image. Directly under these folders, a folder is created for each session, with the name of the session identifier from the database. Within these folders, all the images from that session are stored, and each of these images is named with its capture timestamp.

The data storage for this application is fairly straightforward, and aims at making the data easy to upload, and easy to access. This scheme also allows for the data to possibly be used in other studies beyond the scope of this research.

3.1.2 Data Storage Implementation

The back-end and server side of many mobile applications can often be quite complicated, even more complicated than the client app. This research focuses on mobile computing, and has no need for a complex, custom made back-end. All of the previously mentioned design is quite standard, which means many "out-of-the-box" solutions exist. One of these solutions was chosen for this project so that more time could be dedicated to the design and implementation of the mobile application.

In particular, Microsoft's Azure services were used for storage. Azure is a fitting choice considering the architecture of the mobile project. Rather than coding each of the platform apps (iOS and Android) in their native languages, a tool from Microsoft was used called Xamarin. This allows the test application created to be written completely in .NET and is very naturally oriented to the Microsoft ecosystem.

Azure is also very easy and intuitive to use, and its cost is very minimal. The Azure portal has a convenient user interface, which meant no time needed to be spent visualizing the data during development time. Additionally, Azure has all the services needed for this project, which included a mobile service, database storage, and blob storage. This in total allowed for quick and easy development of the mobile application.

Only one struggle was found with Azure during the testing process. Downloading data from the Azure database was not intuitive or flexible. Only two methods exist for easily retrieving the data from Azure, one of which only runs on the .NET framework (does not run on the Mono Framework). This binds the project to one JavaScript (Node.js) framework for grabbing the data from Azure. While this is not much of an inhibitor, it is an area in which Azure proved to not be very flexible.

3.2 Mobile Application

3.2.1 Mobile Application Design

The mobile application created for this project is very simple; it simply collects and organizes the relevant data, and uploads them to the online database. One of the more challenging parts of the mobile application was designing for both Android and iOS. Each of these devices have different capabilities and use different hardware. The biggest difference between the two relevant to this research is Android's native support for burst capturing images and iPhone's lack of support.

Burst capture is still possible on iOS however, as iPhones can support full resolution samples from a video stream at 10 fps. This is the method used by the research presented here. Further differences exist between these platforms as well; however, both have the capability to accomplish the task at hand. In particular the platforms both support burst image capture in some way, and both generally have an on-board accelerometer and gyroscope. One additional device used in the tests is

the vibrator on the phone as well. The phone vibration is another place where the iPhone gives us difficulty. On Android, the specific length of the vibration request can be set; however, on iPhone only one set length of vibration can be used, which is actually quite short. While multiple requests can be issued, it is difficult to control them to be in sync with the photos.

The user interface for data collection was also kept as simple as possible. This application consists of a simple preview window, along with 5 colored and numbered buttons below this window. Each button corresponds to a different type of data collection. The first of these buttons simply vibrates the phone and records the sensor information (with the forethought of using this for calibration if it is later necessary). The next two tests involve taking a burst shot of pictures, while recording the sensor data from the accelerometer and gyroscope. The first of these two tests does not attempt to vibrate the phone while capturing images, while the second does. The final two tests are a repeat of the ones just described; however, the experiment is designed for one set of these to be taken on a tripod, while the other is taken in the user's hand.

As mentioned previously, the application on each test simply records all relevant data, uploads the images and data to Azure storage, and proceeds to the next test. Between these operations however, the application responds with multiple notification pop-ups indicating capturing has finished, and data upload has completed.

3.2.2 Mobile Application Implementation

For this project Xamarin was used for the implementation of these applications. Xamarin allows developers to create native iOS and Android apps in C#; it additionally allows these applications to use the .NET framework, which makes data upload to Azure fairly painless. This framework was also chosen for familiarity and previous experience, cutting down the development time needed to write the application.

Xamarin's framework also allows for code to be shared between the iOS and Android projects. For this project, all code relating to Azure and the data service only needed to be coded once, then was used in each platform. Additionally a few cross-platform plugins were used for easy device management and vibration control.

The architecture of this application is straightforward and consists of four main components. As previously mentioned, a single data service exists for both platforms. A burst image camera is also created for each device, the Android API is used for its platform, and a video burst camera is implemented for iOS. Finally a sensor recorder is also made for each platform which abstracts the available readings for each device. On Android, it is almost completely unknown what sensors will be present, while iOS provides some higher-level sensor functionality. The sensor recorders unify the readings into one, easy to store type. The fourth and last component is the control layer, which hooks together the previously mentioned components, along with the user interface.

These platforms both have very simple designs, however both are easily able to provide all the needed data and images for testing which methods will be most effective with limited computing resources. Xamarin was also chosen for its ease of use and friendliness with the chosen ecosystem.

3.3 Experiment

3.3.1 Experiment Design

Since not much research has been done in this field, the experiment was designed to take into account anything that could affect the ability to register an image with a phone's sensors. Whether or not the vibration was needed to induce a phase offset was also unknown, which added to the variety of data desired. The main variables which were subject to change were the mounting of the phone (a tripod or a user's hand) and the vibration during capture.

The process of the experiment proceeded as follows. After the data collection application was installed on the volunteer's phone, the phone's mass was measured in grams and recorded. A calibration reading was then taken, in case any calibration is needed during data analysis. To take the calibration reading, the phone was set flat on a table, and sensor readings were taken while vibrating the phone.

Once the calibration process was done, the phone was mounted to a tripod for image capturing. The image used for the capturing is simply a printout of various items that will aid in determining the resolving power of the resulting images. Two capture sessions are done on this tripod, one with and one without vibration. This process is repeated without the tripod, and with the phone in a user's hand instead. The goal here is to determine which of these setups tends to yield the best phase offsets for conducting super resolution, if any does.

The phone uploads all data in between the steps mentioned, so that afterward, the app can simply be uninstalled, and the volunteer's phone remains unaffected. This process collects the necessary data as quickly as possible, and is designed to take as little time from the volunteer as possible.

3.3.2 Experiment Implementation

Much of this experiment has already been explained; however, a few details remain. The software's design and implementation have been covered; however, none of the hardware has been discussed.

To begin with, a simple kitchen scale is used to measure the mass of the phone in grams. The kitchen scale is of fitting size and accuracy for the experiment conducted. It is unlikely the mass of the phone will be needed to a fine precision, if it is needed at all. The scale used was able to measure the mass of the phone to a gram.

A pocket tripod was used for this experiment, as it can easily be mounted to a table and the angle can be set for the shot desired. In order to keep the tripod

from moving during the process, the feet were taped down using masking tape. It is not critical to take the exact same photo every time (nor is it possible due to the wide variety of cameras in cellphones) but rather this tape is used to remove human-interaction with the tripod's positioning.

An extension was also added to the tripod, in order to better position the phone above the printout. The extension was essentially a "selfie" stick with standard camera mounting hardware on the stick. On the end of the stick a phone mount was used for securing the phone. The mount simply holds the phone by applying pressure to the sides of the phone through a large clamp-like device.

During the experiment no volunteer was asked to remove their case either, as the cases are yet another variable that may affect how well photos can be registered. While many of the decisions mentioned here are not likely to affect the accuracy to which burst photos may be registered, they were intentionally considered and kept in order to better simulate what might occur if the application was widely distributed. This experiment has been designed to show a standard algorithm process can be implemented to quickly register images for the purpose of super resolution on practically any phone.

3.4 Data Analysis

3.4.1 Data Analysis Design

The core of the data analysis for this project consists of combining the various sensor readings, burst images, and registration and signal restoration algorithms to discover what produces the best image, and what will work the best on a mobile device. Specifically, four different tests were used at capture time, which will be evaluated against each other, three different registration algorithms were used, and four different signal resolving algorithms were used.

For each test case, every combination of registration and restoration algorithms is run. The result of this is a great number of processed images to evaluate. Unfortunately no great mathematical or algorithm process exists to objectively rank these images, and this part must be done by the human eye.

3.4.1.1 Image registration algorithms. Each registration algorithm can have two parts. The restoration algorithms expect either the rotational and translational shifts between the images, or just the translational shifts between the images. The algorithms produce two things however, as they estimate both the rotational and translational shifts between images. Each of the algorithms use a different level of sensor information. An algorithm either uses no sensor information at all, relies on just the rotational sensor information, or relies both on the rotational and translational sensor information.

The first kind of algorithm, which uses no sensor data, is the one found in the research of Vandewalle et al. This particular algorithm was used as it was shown to have an efficient balance between speed and accuracy. This algorithm, without any speedups could not be realistically usable on a mobile platform. Two versions of this algorithm also exist, one which just estimates translational shifts, and the other that measures rotational and translational shifts. The one which estimates both rotational and translational shifts begins by estimating the rotational shifts, and uses this to obtain a better estimate of the translational part.

Through observation, it was found that the algorithm just mentioned performs the translational estimation much quicker than the rotational estimation. This leads to the first sensor based algorithm: a hybrid between sensor data and algorithm estimation. The hybrid that is created replaces the entire rotation estimation portion, with the rotation obtained from the gyroscope and accelerometer. The rotation of the device (phone) is assumed to be the same as the rotation of the image the camera captures.

Finally, one last algorithm is also used, one almost completely based directly off sensor data. To begin with, the position of the camera during capture time is estimated from the accelerometer values. These readings are filtered using a Kalman filter, and this in turn gives us the position estimate. Unfortunately, estimating the actual pixel offset in this situation is not feasible, as the distance from the camera to the image is not known. Instead, the position is used strictly to tell us the direction in which translation occurred.

Once the direction has been estimated, each image is translated within a range of values along this direction until a maximum correlation is found in relation to a reference image. Nothing in this algorithm accounts for the rotational shift already estimated; instead, to account for rotation, the images must be rotated before being sent to this function. With the images being rotated going into the function, the function will yield their post-rotation translational shift. The goal of using these three algorithms is to gauge to what extent the sensor data will be useful to the registration algorithms.

3.4.1.2 Image restoration algorithms. As previously mentioned, these registration algorithms are then combined with four other signal restoration algorithms. These algorithms take the rotational and translational shifts along with the low-resolution signals (images) in order to create a higher resolution signal. There are many methods of doing this, however this research uses a particular set of four. The algorithms explored in this research include an iterative back projection method and a robust method, along with an implementation of projections onto convex sets and the Papoulis Gerchberg algorithm.

To begin with, the iterative back projection and robust methods are extremely similar to each other. Each of them operate on very similar principles, with one small change between the two. These algorithms begin by modeling the low-resolution pictures as a decimation, blurring, and translation of a higher resolution image (along

with an error term). The goal of each algorithm is then to minimize the error between the model specified. Using a derivative, the square error of this model is minimized. This minimization yields a step towards minimizing the error in the high-resolution image, for one of its low-resolution counterparts.

These steps are iteratively taken, until a satisfactory error measurement is achieved. One simple difference remains between the two algorithms. These methods do not use the derivative directly; instead, they compute a gradient over all input images, which then yields the direction of the step used to minimize the error. How this gradient is computed over all input images is where these algorithms differ. For iterative back projection, each derivative is summed together. However, for the robust solution, the pixel-wise median is used instead of the sum. This simple change is what is commonly done in robust algorithms, and it usually has a dramatically important effect. For the purposes of this algorithm, using the median instead of the sum means that outliers affect the resulting image far less than they would in a sum.

The final two algorithms operate under a similar idea as well. Projections onto convex sets, or POCS, operates on the principle that each of the images is a convex set, and that the point at which they intersect, or the point between all sets, will represent the higher resolution image. With this algorithm, the initial working image is assumed to be completely zero. Each of the input images is then upsampled (not interpolated) and its non-zero values are assigned to the working image's equivalent positions (this is the projection). Some pixel values in this working image are zero, as the input images have been upsampled. To account for this, a blur filter is applied to the working image(specifically, a 5x5 matrix with pre-defined values is convoluted with the image). Finally the working image is then projected again onto the input images. This process continues to iterate until the working image fails to improve significantly, or the maximum number of iterations is reached.

The Papoulis Gerchberg algorithm used is very similar to POCS. The same setup is done, along with the same iteration scheme (projecting onto the upsampled input images). The one difference between these two algorithms is the blurring method used. Clearly, the blurring here is what actually accomplishes the interpolation between known pixels. Instead of using a blurring filter with convolution, a low-pass filter is applied. This filter is applied by using a Fast Fourier Transform to move the image into the frequency domain, the high components are knocked down, and then the image is Inverse Fast Fourier Transformed back into the spatial domain. As a result of this operation, a blur results in the spatial domain. One useful and intuitive take away from these two algorithms is the final step is always to project onto the known input images. This means, as long as the shift estimates are accurate, the resulting image will always contain data that is known to be correct.

3.4.2 Data Analysis Implementation

All data analysis is done in Matlab. This is a standard in this field, which also means that open-source code already exists that can be utilized in this research. Particularly, Vandewalle's research on improving the registration algorithms has a great open-source set of code available. This code base has a few registration algorithms in it, as well as all the image restoration algorithms used in this research. Beyond this code base, the sensor registration algorithms were created for this research, and a Kalman filter was implemented as well.

3.4.2.1 Sensor registration algorithms. Two sensor registration algorithms were implemented, one which only used rotational data, and another that only used translational data. The sensor data tells the relative angle of the device at a given time. In order to achieve this all values throughout the capture time of the device are used, and an individual photo's capture time is interpolated for greater accuracy. How this rotational data is achieved is internal to both Android and iOS. Both of

these devices provide high-level sensors, which utilize sensor fusion to give a more accurate or different sensor reading.

The first algorithm implemented is an adaptation of one of the algorithms given by Vandewalle. In particular, the algorithm for estimating rotation and shift is modified slightly. Rather than estimate the rotation inside the algorithm, it is changed to take the rotation estimate from outside. The algorithm then proceeds to estimate the shifts, after accounting for the currently given rotation information. It is trivial then to use the sensor estimates for the rotation estimation, and the algorithm is complete. This is very convenient as the rotation part of the estimation is the most time consuming.

Another algorithm is also implemented as described before. The direction of the translational shift is estimated by using the accelerometer values and the Kalman filter. In Matlab, this function accepts the input, the estimation of direction, and the maximum distance to check (in pixels). Here it is assumed that the direction given is the direction of translation. Using a function provided in the Vandewalle code, the image is translated a non-integer number of pixels until maximum correlation is found. The direction of estimation is obtained through the positions given by the Kalman filter. In particular the arctangent of the vertical values divided by the horizontal is used.

The only other algorithm used is the one that estimates the shift without any sensor feedback. This one is given by the Vandewalle code directly (Patrick Vandewalle and Vetterli 2005).

3.4.2.2 Using restoration algorithms. Scripts are written in Matlab in order to use all the previously mentioned algorithms to produce all the necessary data. All images for a session (as described in the experiment) are loaded into memory. Each of the three rotation and shift algorithms are run on these images, their run time

is recorded, and the estimates are recorded. This data, once generated, is stored in comma separated value (CSV) files.

Once the rotational and translational estimates are done, each algorithm previously mentioned is run for each estimate. The one thing that changes for some algorithms is the images passed in. For the algorithms which only expect a translational shift, the images passed into the algorithm are rotated according to the rotational estimate. This allows for all the algorithms to utilize the same estimation data.

CHAPTER FOUR

Experimental Results

An experiment with this many variables has a variety of results to be evaluated. Here, the algorithms used to restore the higher resolution image will be evaluated first, as considering the results from these algorithms will help in making conclusions about the registration algorithms. The restoration algorithms will be evaluated for which ones work best in the widest variety of cases. Following this, the registration algorithms will be considered for their ability to speed up the super resolution process and for whether or not this speed up affects the quality of the improved image.

Each capture method that follows uses the same input images in order to allow for consistent comparison. The particular set of input images used was chosen to best represent the sample collected. In general the majority of the samples follow that which is shown, unless otherwise specified. All samples (both input and output) are public and available for viewing.

4.1 Restoration Algorithm

The restoration algorithms will be evaluated together as they apply to four different capture methods. These four capture methods are as follows: the phone is mounted to a tripod, the phone is mounted to a tripod and vibrated while capturing, the phone is held in the user's hand, and the phone is held in the user's hand while vibrating. Each of these methods will be evaluated to obtain which restoration algorithms work best under which conditions.

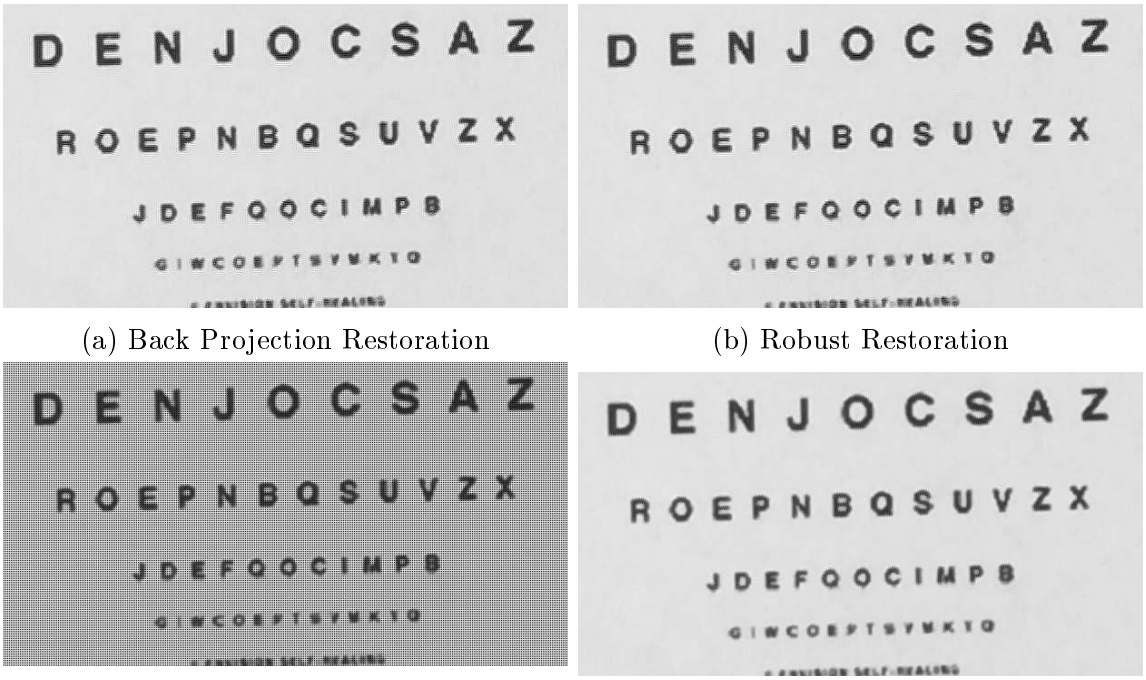
Additionally, this section will serve as a good place to understand the generalities of what works well for super resolution, before evaluating specific variables in greater detail. For instance, the point of this section is not to evaluate the registration

algorithms themselves, but it is quite obvious that the purely algorithmic registration seems to preform consistently well.

4.1.1 Tripod Mount, No Vibration

All figures and images shown in this section will be taken from the same phone trial. Additional image results are available upon request. The images in this section were taken mounted on a tripod with no vibration.

4.1.1.1 Algorithmic registration. Figure 4.1 shows the performance of all the restoration routines when a pure algorithmic approach is taken to register the images.



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

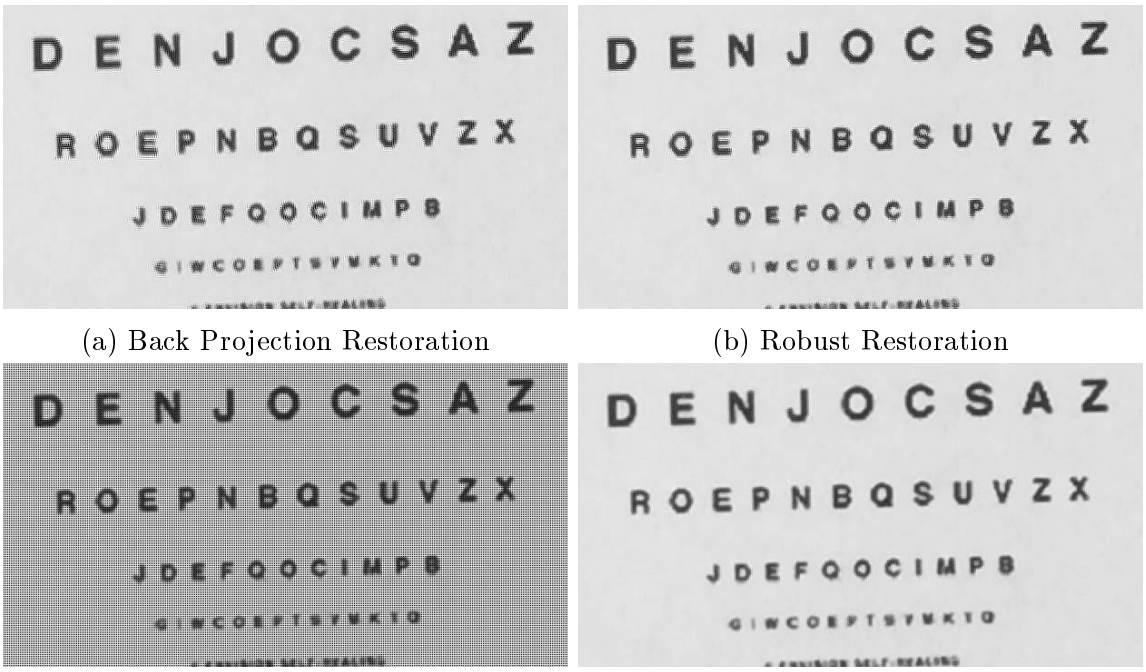
Figure 4.1. Tripod, No Vibration, Algorithm Registered Results

As can clearly be seen, all of these algorithms perform very similarly, other than Papoulis Gerchberg (Figure 4.1c). The images at full size from many of the Papoulis Gerchberg algorithm restorations appear to be darker; however, this is simply due to

these black square artifacts. These squares become evident when zoomed in closely on the image. This can occur when not enough of the higher-resolution image can be accounted for in the lower resolution photos and their phase offsets. In other words, the phase offsets obtained in the burst images were not different enough to restore a high-quality image.

This does not show Papoulis Gerchberg to be a poor algorithm, but rather that it is sensitive and needs a greater number of phase offsets to produce an acceptable image. These images (and others taken in the same manner) support that the Back Projection, Robust, and POCS implementation are more suited for this experiment.

4.1.1.2 Hybrid registration. The next set of photos (Figure 4.2) uses the hybrid registration algorithm described previously.



(a) Back Projection Restoration

(b) Robust Restoration

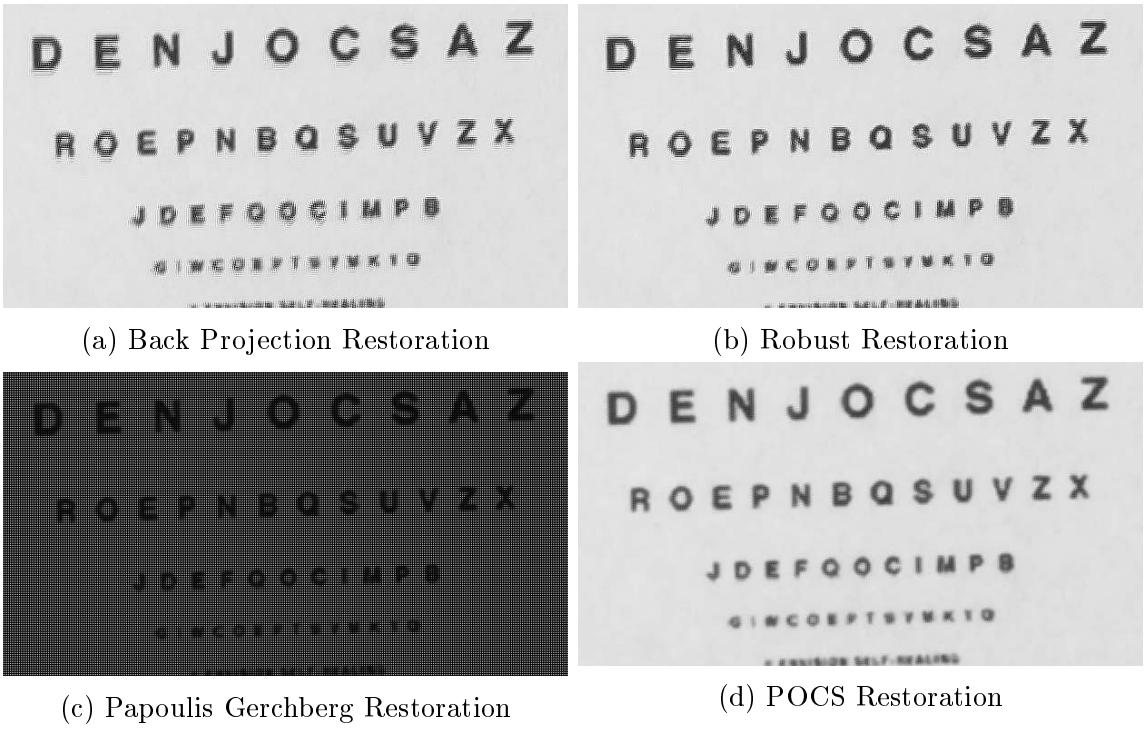
(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.2. Tripod, No Vibration, Hybrid Registered Results

These images appear almost identical to those in Figure 4.1. Again Back Projection, Robust, and POCS restoration work extremely well, while the Papoulis Gerchberg algorithm fails to completely restore all parts of the higher resolution signal (due to poor registration or a lack of phases present). This is a favorable result, as it supports the usefulness of the hybrid algorithm developed.

4.1.1.3 Sensor registration. Finally, Figure 4.3 shows the results from using a pure sensor approach to estimating the rotations and shifts in the images.



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.3. Tripod, No Vibration, Sensor Registered Results

Using the sensor estimation scheme, a bigger difference can be seen between the algorithms. The Papoulis Gerchberg algorithm still performs very sensitively compared to the other three. This is not unexpected. Additionally, in this specific case the back projection (Figure 4.3a) and robust (Figure 4.3b) both generate duplicate artifacts, clearly seen outlining the letters. The back projection result shows this

much more clearly than the robust solution, however both contain this artifact. It is worth noting that this does not occur in every data point of sensor registered images on the tripod, though it is found in more than this individual case. Additionally, this does not fault the restoration algorithms, rather poor registration or blurring in the input images is to blame.

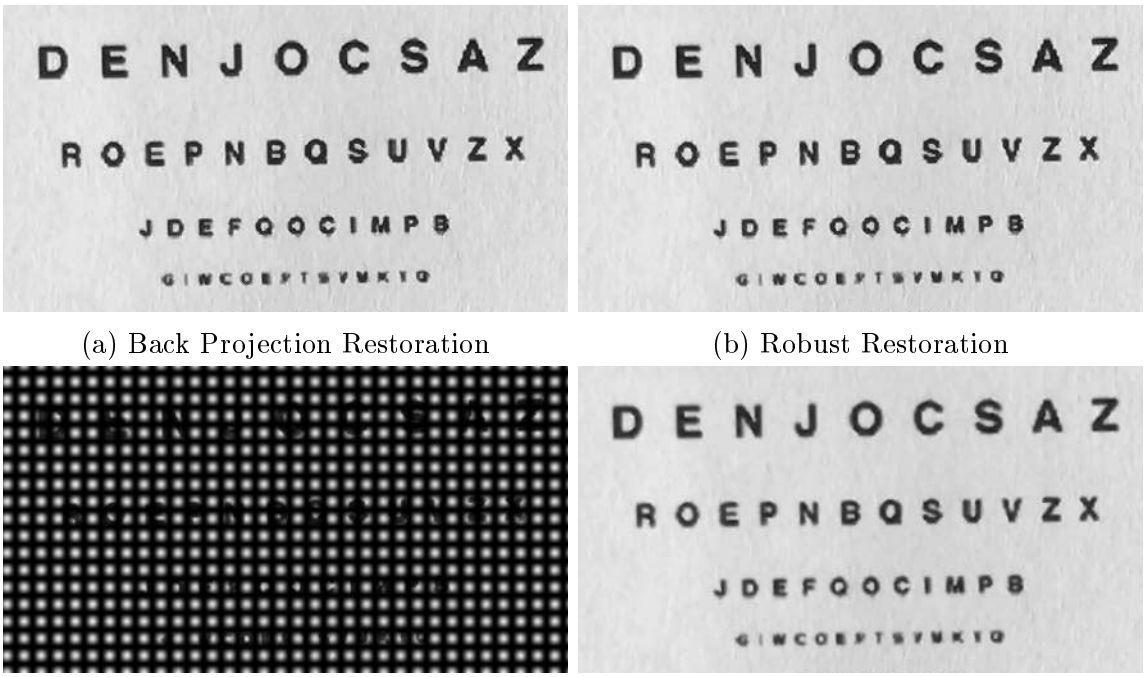
It is notable that POCS performs with consistent quality as shown here. Figure 4.3d clearly does not have any extra artifacts, though it is not as sharp as the other images presented. This makes sense as well, since part of the POCS algorithm involves blurring the image to cause a sort of interpolation between known pixels. A trade-off exists in this set that is often found in imaging algorithms, either sharpness is achieved at the cost of some noise (artifacts) or a lack of noise for a more blurry image. Each of these algorithms has a different ability to restore a signal in the face of a blurry, input, noisy input or poor registration.

4.1.1.4 Tripod, no vibration conclusion. After evaluating all these algorithms for the tripod without vibration case, it is clear that back projection, robust and POCS solutions perform satisfactory for this experiment, except in some cases for sensor registered images. Papoulis Gerchberg is clearly shown to perform poorly in all cases, and this is evident in the rest of the data set as well.

4.1.2 Tripod Mount, Vibration

The next sections will follow the format of the previous section, evaluating each algorithm's performance for the phone mounted to the tripod, and vibrating during capture time. All algorithms will again be shown under each registration algorithm. In general the pictures shown here can be seen to be more consistently improved by the super resolution process. This occurs as the algorithm approach to registration will find a registration that yields a high correlation value.

4.1.2.1 Algorithmic registration. Figure 4.4 contains an example set of results from burst images taken on a tripod with vibration during capture, registered purely with an algorithm.



(a) Back Projection Restoration

(b) Robust Restoration

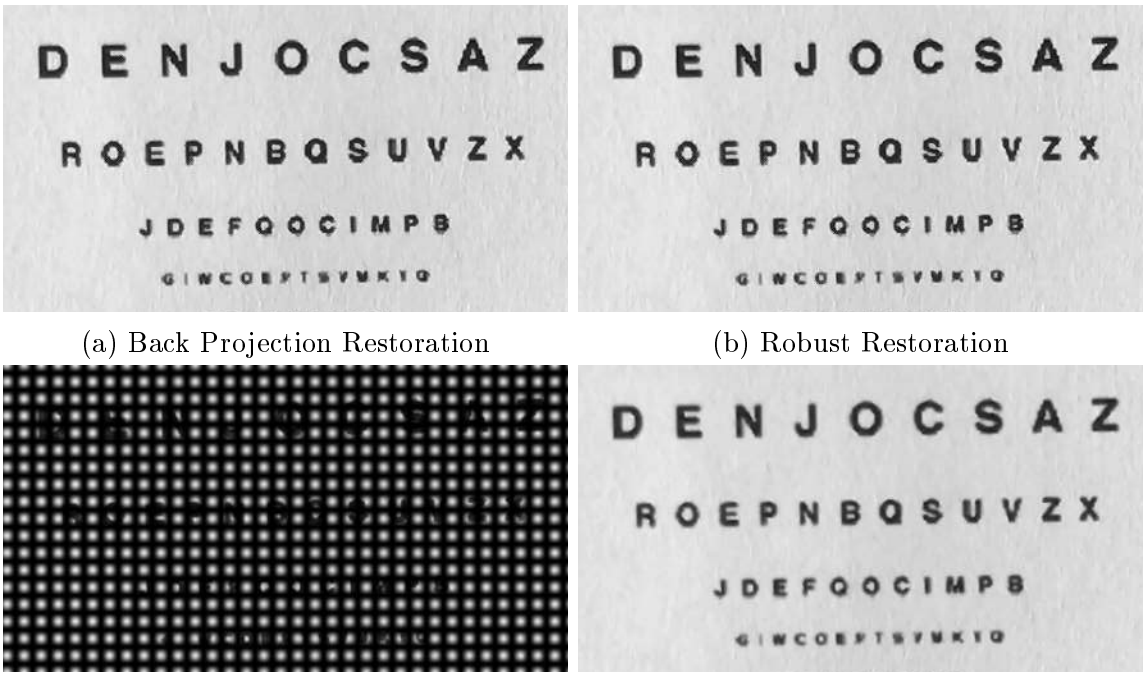
(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.4. Tripod, Vibration, Algorithm Registered Results

One of the obvious things about this set of images is that again the Papoulis Gerchberg restoration is sensitive again to the phase offsets of the input images. It has small black artifacts all over the picture, which causes an overall darkening of the photo. As before, the back projection and robust solutions are very consistent here, producing a very sharp image, where even the texture of the paper can be seen in the photo. Also as seen already, the POCS algorithm produces a very smooth image; however, the sharpness of the letters is lost with this restoration.

4.1.2.2 Hybrid registration. The hybrid registration photos look almost exactly the same as their algorithm registration counter-parts.



(a) Back Projection Restoration (b) Robust Restoration

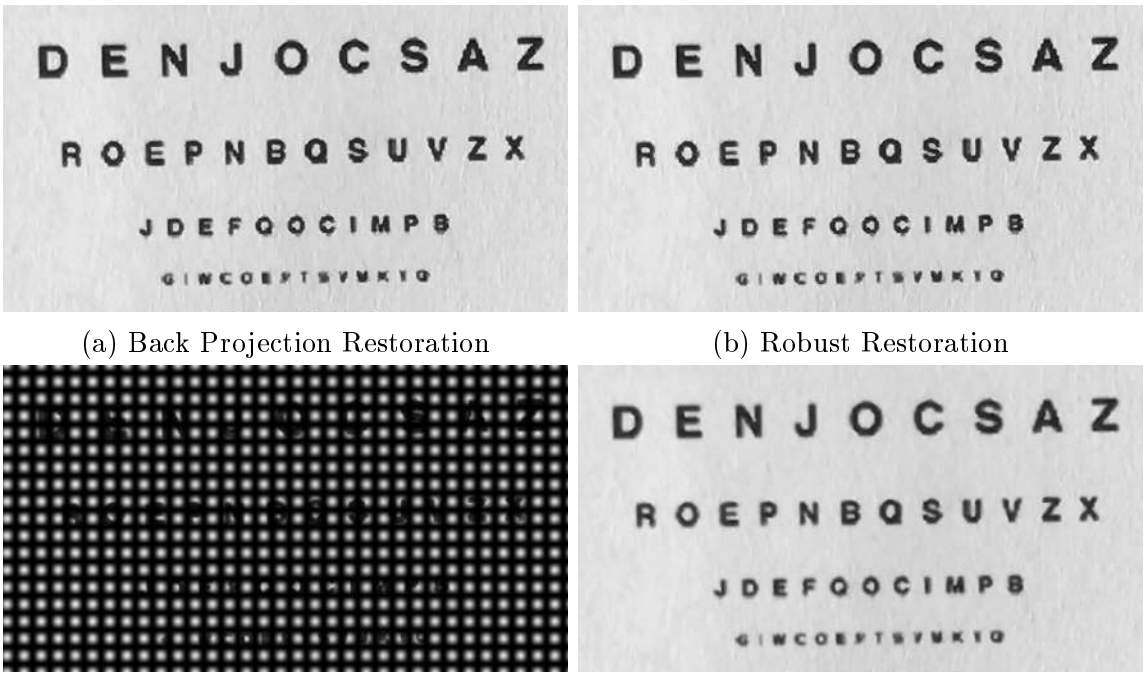
(c) Papoulis Gerchberg Restoration (d) POCS Restoration

Figure 4.5. Tripod, Vibration, Hybrid Registered Results

Here again the texture of the paper can be seen, as well as fairly sharp letters from the back projection and robust solutions. POCS, as before is not as sharp as these photos, with a slight blur being present at the edges of the letters. Finally, Papoulis Gerchberg again yields a very dark image, due to its large amount of black spots on the photo.

4.1.2.3 Sensor registration. The results (Figure 4.6) again don't change with the sensor registration algorithm for the tripod and vibration case. This differs from not using vibration on the tripod, for which none of the following algorithms produced crisp results.

Here the back projection and robust restoration algorithms produce very nice results, very comparable to the rest of the images taken with vibration taken on a tripod. It is observed here again that the Papoulis Gerchberg algorithm produces a very dark image, while the back projection, and robust solutions produce sharp



(a) Back Projection Restoration (b) Robust Restoration

(c) Papoulis Gerchberg Restoration (d) POCS Restoration

Figure 4.6. Tripod, Vibration, Sensor Registered Results

images. POCS again yields a slightly blurred, less sharp image than back projection and robust.

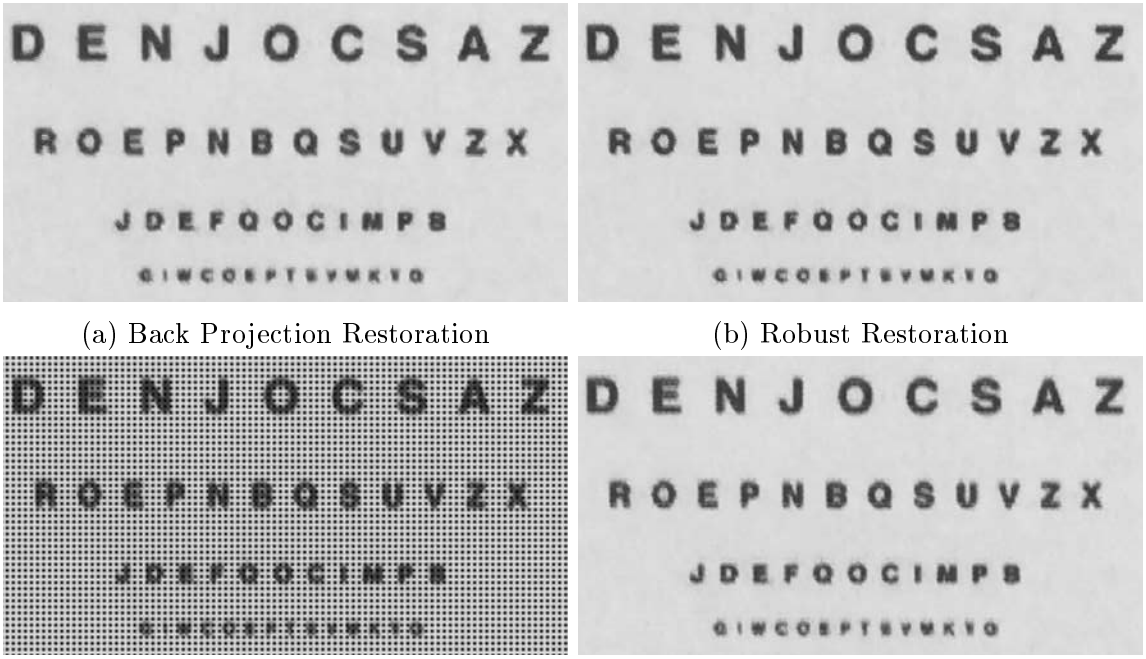
4.1.2.4 Tripod, vibration conclusion. The conclusion for the tripod vibration case supports what was found in the tripod without vibration case, that the algorithms that seem to work best for these setups of super-resolution are back projection, robust and POCS. These algorithms will now be evaluated without using a tripod, in order to determine what will be the best universal algorithm for mobile devices to use.

4.1.3 Handheld, No Vibration

In this section, the way in which the photos are taken changes dramatically. The phone is now handheld, rather than mounted on a tripod. This method is much more prone to movement, shaking, and blurring. Because of these factors, both the registration and restoration algorithms will behave with different qualities. Here the quality of the image restoration algorithms will continue to be the focus. Many of the

samples from this type of photo capture are very blurry; since this will be evaluated later, a sample which is not as blurry is used for algorithm evaluation in this section.

4.1.3.1 Algorithmic registration. Following the same pattern as the previous sections, the algorithmic registration is evaluated first for all the restoration algorithms. The same theme found in the previous two capture methods is found in this one as well.



(a) Back Projection Restoration

(b) Robust Restoration

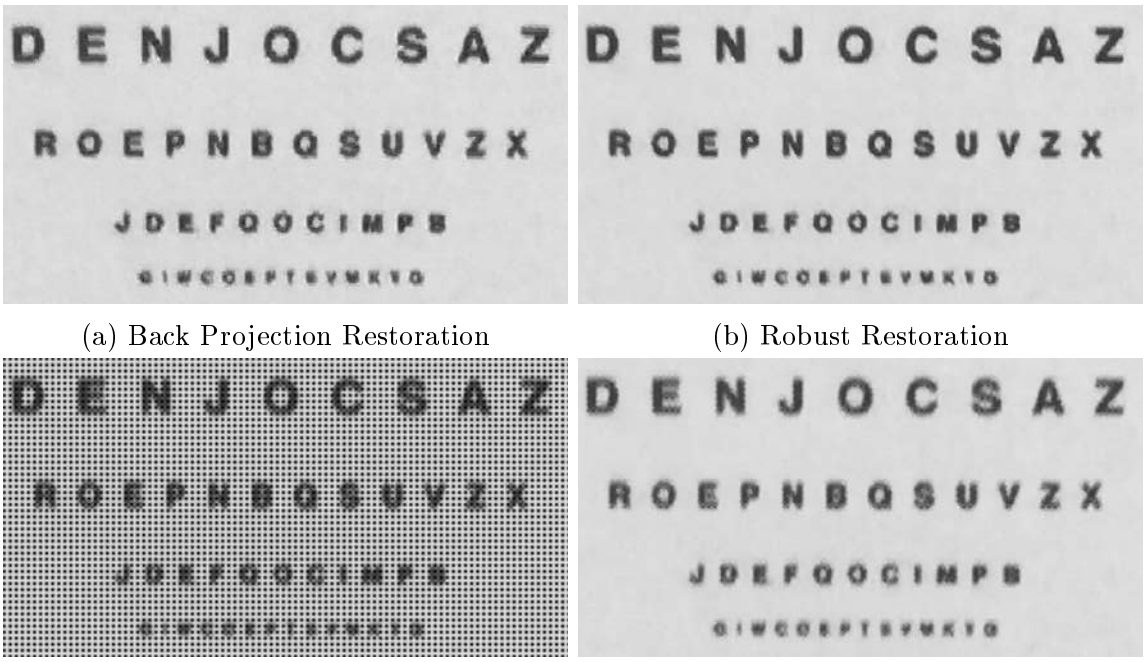
(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.7. Handheld, No Vibration, Algorithm Registered Results

Here (Figure 4.7) back projection and robust solutions still produce favorable results; however, they are no longer crisp and sharp as before on the tripod mount. POCS remains consistent, and produces an almost identical image to the robust and back projection solutions, though it does appear to be slightly less sharp. Papoulis Gerchberg still yields a darker over-all tone to the image; however, it is not as dark in this method (meaning a greater number of phases should be present in the input images).

4.1.3.2 Hybrid registration. Figure 4.8 shows the results when the hybrid algorithm is used to register the images; again, these results are very consistent with all previous results.



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

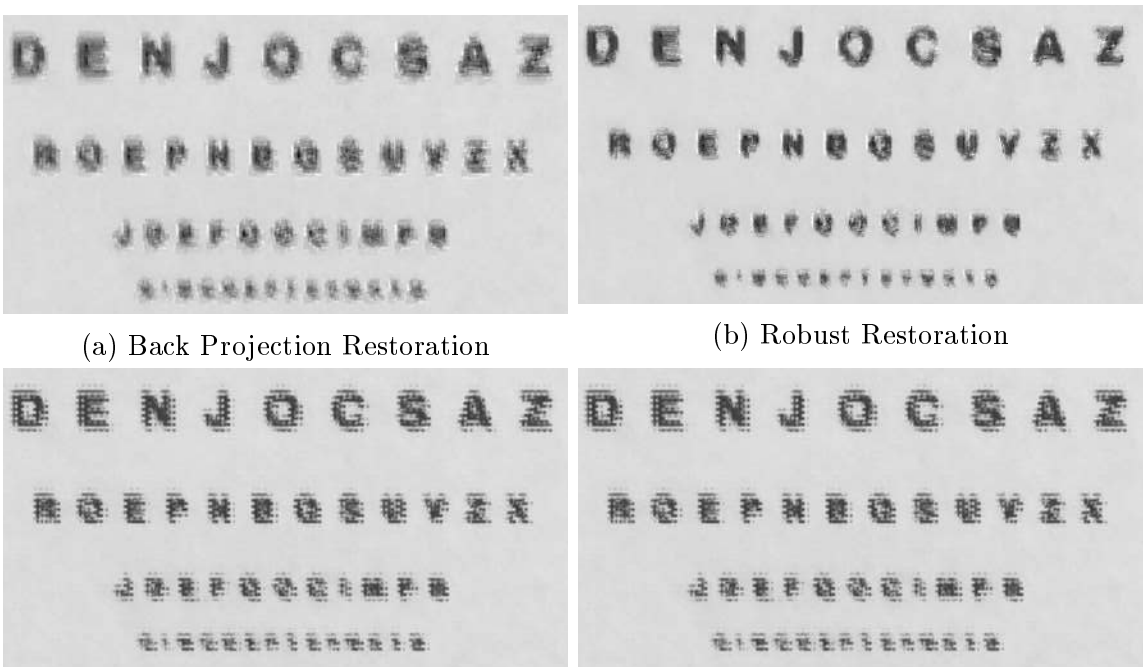
(d) POCS Restoration

Figure 4.8. Handheld, No Vibration, Hybrid Registered Results

As before, back projection and robust produce the sharpest of the images, while POCS produces a comparable photo of slightly lesser quality. Papoulis Gerchberg still yields a darker image as before.

4.1.3.3 Sensor registration. The images presented in Figure 4.9 are clearly of worse quality then those already presented, whether this happens due to poorly captured input images, or poor registration. This will be explored later.

One very interesting result here is that the Papoulis Gerchberg does not have the dark tone as seen in all images presented so far. This leads to the conclusion that while registration may be inaccurate here, it might result in a greater number of predicted phases. Even with the lighter tone however, it very apparent that this is



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.9. Handheld, No Vibration, Sensor Registered Results

not a great photo, containing a large amount of blur, making even the second line of letters hard to read.

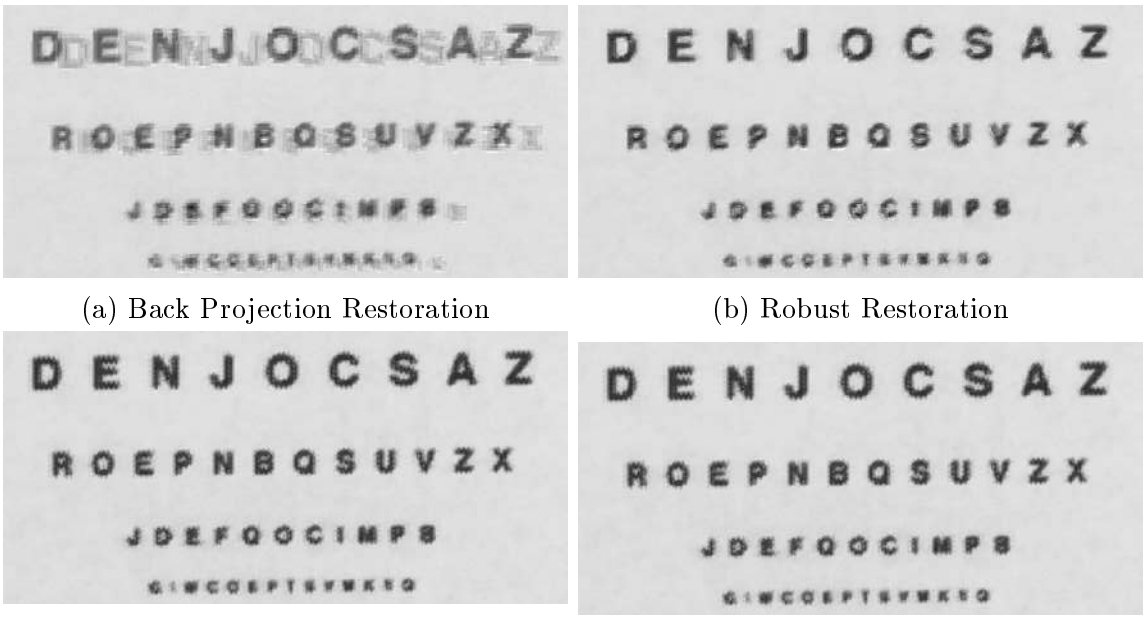
In this case, back projection, Papoulis Gerchberg and POCS all perform equally, producing an image that is hard to read, and about equally blurred. The robust image however (Figure 4.9b) is slightly less blurry, though it is not of satisfactory quality (in that it is very difficult to read still). Particularly in this image the letters are darker and tend to "stretch" less then the other provided images.

4.1.3.4 Handheld, no vibration conclusion. Concluding the handheld, without vibration section, again the back projection, robust and POCS solutions all preform very similarly; however, robust in out-performed these algorithms in the sensor registered case. This result is useful as other variables are evaluated later.

4.1.4 *Handheld, Vibration*

In this section the final capture method is considered, which is a phone held in a hand, while vibrating during capture time. This final case will help bring out which of these restoration algorithms produces the best super-resolution image. Similarly to the last section, many of the resulting images here are blurry, and for the purposes of this section of evaluation, a sharper result is used.

4.1.4.1 Algorithmic registration. Figure 4.10 shows the first results from this method of capture. These photos use the algorithm for registration and therefore should have very accurate registration parameters.



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

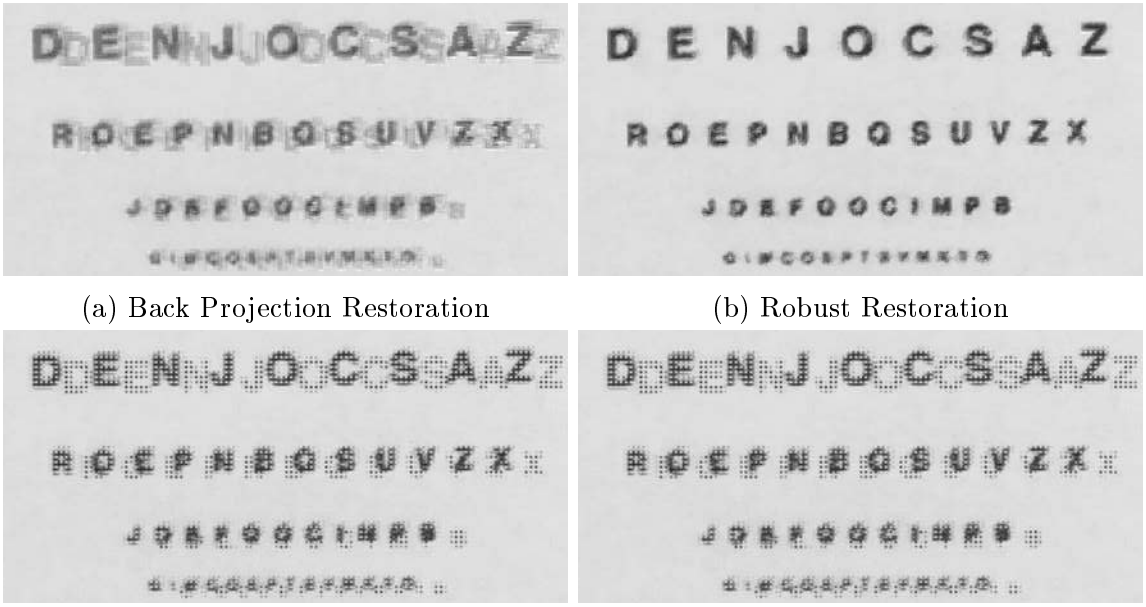
Figure 4.10. Handheld, Vibration, Algorithm Registered Results

In all previous cases, the Papoulis Gerchberg algorithm failed to produce a satisfactory result. Here is the first presented case where Papoulis Gerchberg produces a quality image. Figure 4.10c is not darkened by the algorithm and appears to be just as sharp as the other images. Sadly however, this is not consistent across other

samples taken in the same method. Back projection performs the worst here, as a ghost image can be seen shifted off to the bottom right of the darker letters. Clearly the registration algorithm has failed to properly register all input images, and this "ghost" image appears.

The robust and POCS algorithms in this case perform well, and yield an almost identical image to the Papoulis Gerchberg algorithm. For this sample robust and POCS give the best results, since the Papoulis Gerchberg algorithm does not consistently perform well across additional samples.

4.1.4.2 Hybrid registration. In figure 4.11 the results for handheld vibration registered with the hybrid algorithm are shown. The results here are strikingly different from the last set of images (Figure 4.10).



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

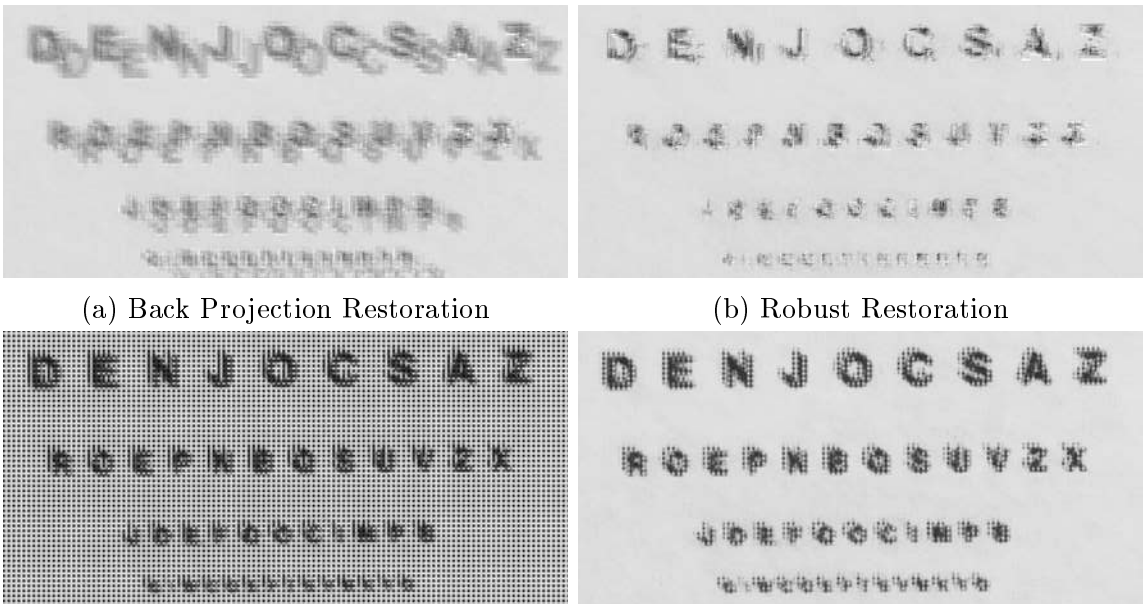
(d) POCS Restoration

Figure 4.11. Handheld, Vibration, Hybrid Registered Results

Using the hybrid registration algorithm, back projection, Papoulis Gerchberg and POCS all produce a "ghost" secondary image. This is obviously not satisfactory,

and all these images have poor quality, again this is due to poor registration rather than poor restoration algorithms. The robust algorithm however, yields a sharp image compared to the other algorithms. While this is certainly not the sharpest reconstruction shown so far, it certainly sets the robust algorithm apart from the others in the presence of poorly registered or captured images.

4.1.4.3 Sensor registration. The last set of images (figure 4.12) evaluating the various restoration algorithms is vibrating a handheld (not on a tripod) phone and using only the sensors to register the images. These images clearly are the worst set seen yet, but will be evaluated for completeness nonetheless.



(a) Back Projection Restoration

(b) Robust Restoration

(c) Papoulis Gerchberg Restoration

(d) POCS Restoration

Figure 4.12. Handheld, Vibration, Sensor Registered Results

Here the back projection algorithm performs as poorly as it has for the other registration schemes with the same capture method. The robust solution severely chops up the image, rendering it unreadable. Papoulis Gerchberg, as before creates a dark image, meaning not all phases are predicted by the sensor algorithm. POCS

creates the most satisfactory result; however, it is still of very poor quality compared to other methods.

4.1.4.4 Handheld, vibration conclusion. Unfortunately, this analysis does not tell much about the restoration algorithms, but rather points to back registration parameters. This section therefore suggests that the robust solution is the best for images captured on a hand-held phone with vibration. The robust solution consistently gave an image that was readable, and tended to give results that were sharper than POCS and the other algorithms. Papoulis Gerchberg also performed very well in one case when using the pure algorithmic approach, but broke down in other similar samples and when using the hybrid registration algorithm. Finally POCS often produced "ghost" images and is less consistent then robust.

4.1.5 Restoration Algorithm Conclusion

When considering all capture methods, and all registration methods, robust seems to be the obvious universal choice. This result should not be too surprising, since robust solutions in general tend to hold up in the face of signal error. The tripod cases in general found back projection, robust and POCS to be the best choices, and the off-tripod cases narrowed these algorithms down to robust.

4.2 Sensor-Based Estimation

In this section, the new algorithms created for image registration will be evaluated. Considering the results from the previous section it is fairly obvious that the purely algorithmic approach to registering images tends to most satisfactory in the most number of cases. This leads to using this as a baseline to compare the sensor-based registration schemes against.

These algorithms are evaluated in a couple ways. For the rotational estimation, this can be directly compared to the purely algorithmic approach. The sensor algorithm described previously utilizes the sensors to find the direction of the translational

shift; therefore, this will be directly evaluated as well. This is done by computing $arctan\left(\frac{y}{x}\right)$ for each algorithmic estimation. Additionally, the translational x-shifts and y-shifts can be independently compared.

4.2.1 Rotation Estimation Results

This section evaluates using the gyroscope and accelerometer present on phones to estimate the degree of rotation the phone experiences during capture time, and therefore the degree of rotation each image should be from each other. From here, the rotation estimation is then fed in to the algorithm developed by Vandewalle, letting the algorithm presented in said research estimate the translation between photos (Patrick Vandewalle and Vetterli 2005). Three examples will be given for each capture method, in order to display the consistency of this algorithm; additional data samples are available as well.

4.2.1.1 Tripod mount, no vibration. The samples that have been chosen

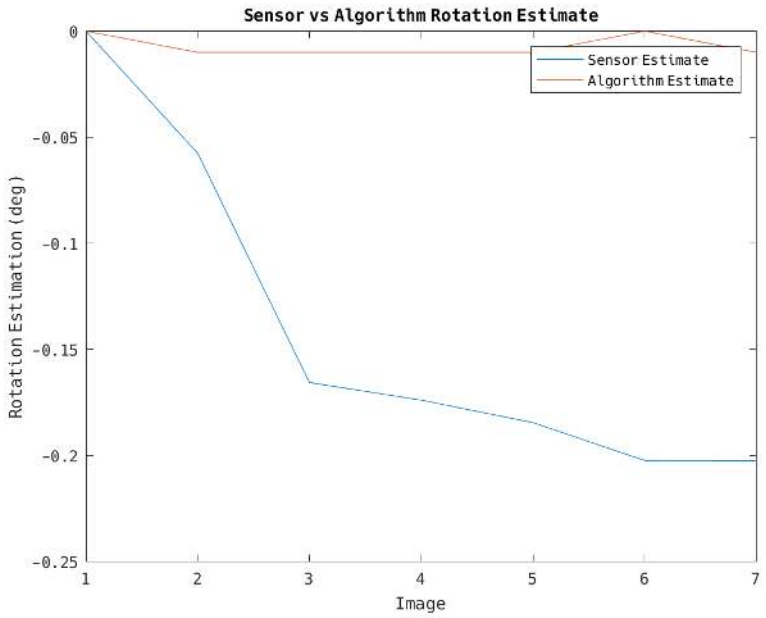


Figure 4.13. Tripod Mounted, No Vibration, Sensor Rotation Estimation Sample 1

here are an attempt to represent the various results seen in the larger data set. This first graph (Figure 4.13)is what is typical of most of the data set, an estimate in the correct direction, with a very small amount of error. Another point of interest evident here, which continues throughout this capture method is that the algorithm tends to yield few rotational changes. Each of these measurements is relative to the first image captured, and most of the algorithm registrations have straight lines for their estimates. This again is an expected result, since the phone is mounted to a fixed object.

While the graph appears to have a large amount of error, it is important to notice the scale of the y-axis. In this example the y-axis is in degrees, and the largest error that can be seen is about 0.2°. This is not a large error considering how small of a shift is being measured. The accuracy of this measurement too is fairly good, considering that accelerometers and gyroscopes in phones tend to be meant to detect larger movements (as in tracking landscape vs horizontal orientation). The

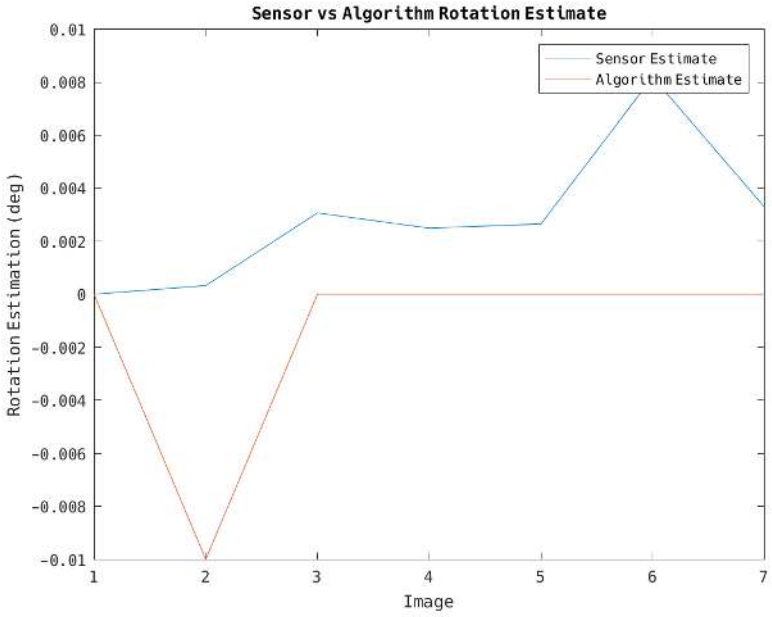
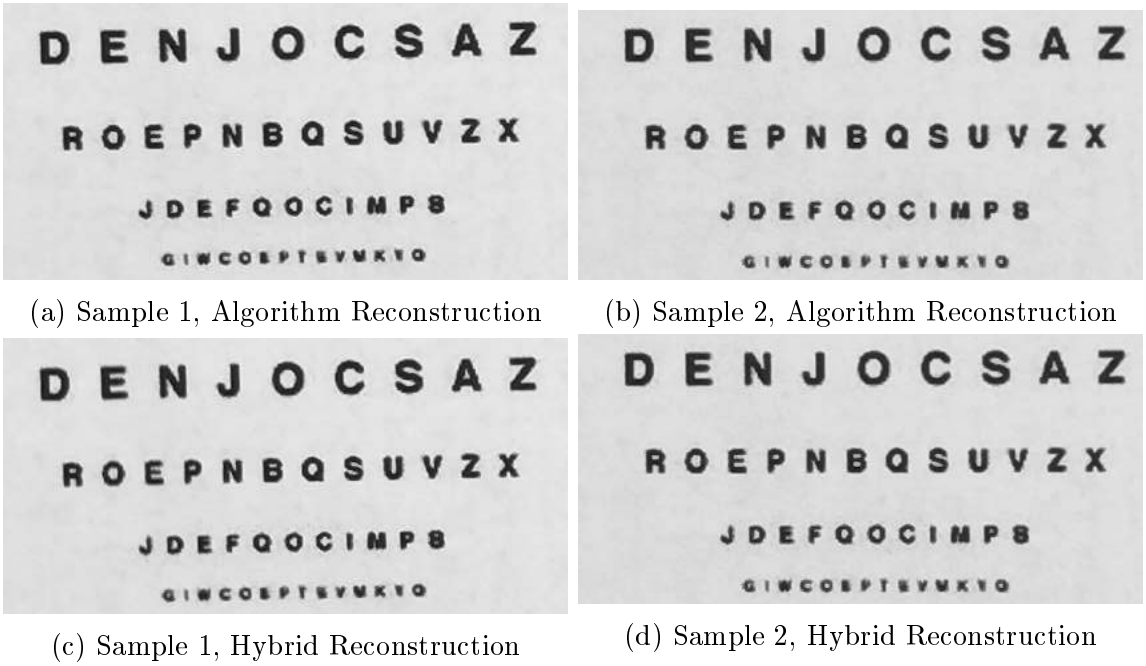


Figure 4.14. Tripod Mounted, No Vibration, Sensor Rotation Estimation Sample 2

least typically occurring situation is illustrated by Figure 4.14. This does occur with some frequency; however, it will be shown to not have a great effect on the final constructed image. Here the big problem is that the direction is being predicted incorrectly by the sensor (assuming the algorithm yields the truth). Rotation is always corrected for when generating the super-resolution image, so when this occurs the image actually becomes less correlated, since the image is rotated in the incorrect direction. Fortunately, when this happens the greatest error tends to still be small (in this case slightly above 0.01°). When error is this small, rotation tends to have a very minimal effect on the reconstruction.

4.2.1.2 Tripod mount, no vibration conclusion. In this section it was shown



(a) Sample 1, Algorithm Reconstruction

(b) Sample 2, Algorithm Reconstruction

(c) Sample 1, Hybrid Reconstruction

(d) Sample 2, Hybrid Reconstruction

Figure 4.15. Tripod Mount, No Vibration, Hybrid Registration Sample Reconstructions

that the angle obtained from the gyroscope and accelerometer for the tripod mounted without vibration case works very well. While one of unfavorable result from using the accelerometer and gyroscope is when the sensors indicate a rotation in the wrong

direction (Figure 4.14). Figure 4.15 shows the corresponding reconstructions for each sample (using the robust solution, since that was shown to work consistently).

What is notable here is that despite the estimation being in the wrong direction, the result (Figure 4.15d) is not any different than the other images reconstructed under the same pattern. The difference in overall tone of the image here comes from different lighting conditions, and is not an artifact of the reconstruction process. Therefore all rotational sensor predictions for the tripod mounted, without vibration case supports using these sensors for rotational estimation.

4.2.1.3 Tripod mount, vibration. Even with vibration added, the results in this section do not change much from the previous section. The results in this section consistently have small error, and the images resulting are also satisfactory.

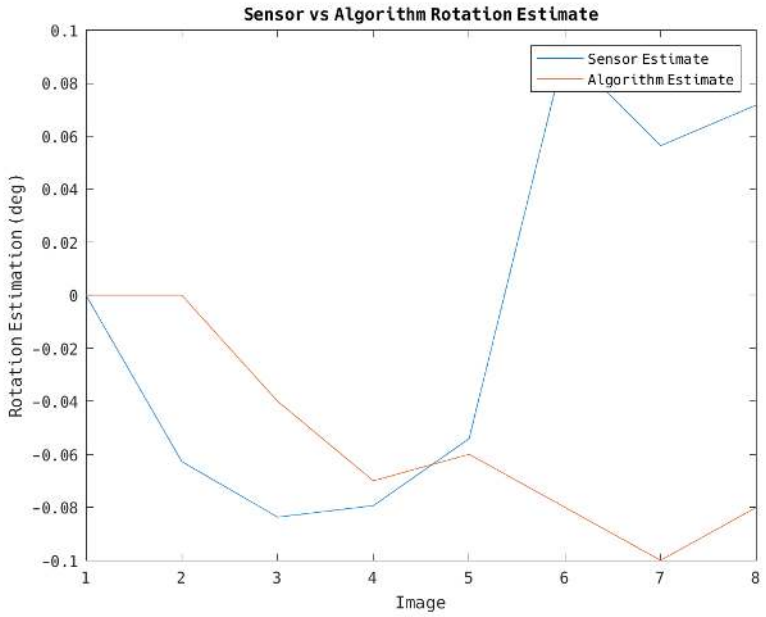


Figure 4.16. Tripod Mounted, Vibration, Sensor Rotation Estimation Sample 1

In some cases, adding vibration to the capture method results in the registration algorithm itself finding a greater change in rotation from picture to picture.

Notice in Figure 4.16 how the algorithm estimate does not remain flat through the majority of the burst capture.

Here the sensor detects many of the rotational differences well; however, on the last three images the sensors predict rotations in the wrong direction (according to the algorithm). These errors are still very small though, with the most being an error of about 0.16°. As before, this error is small enough that a satisfactory and usable picture results from using this estimation.

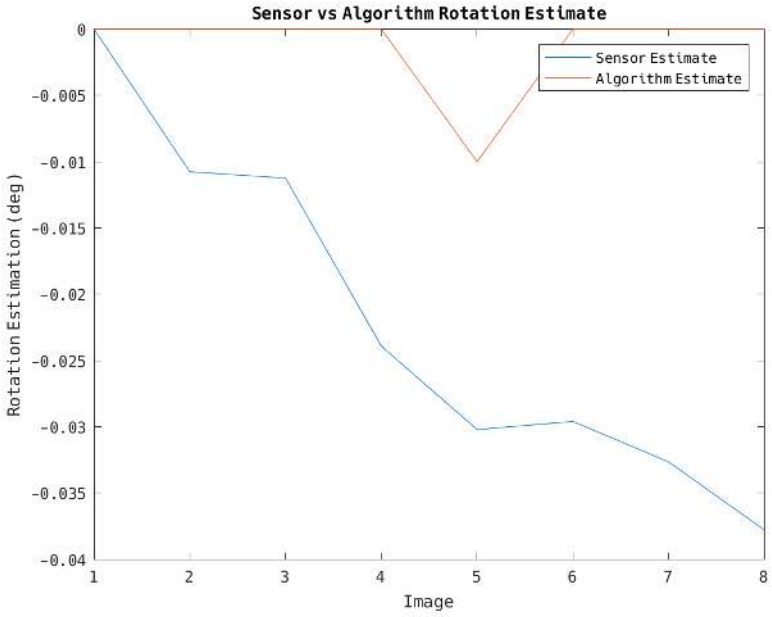
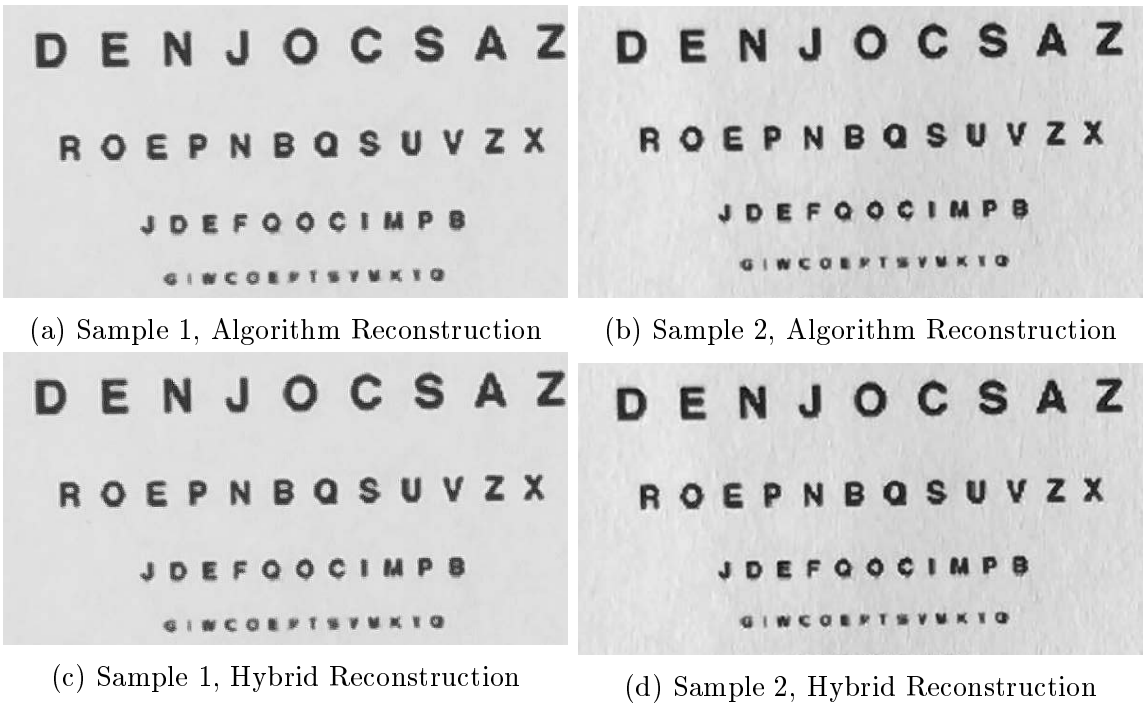


Figure 4.17. Tripod Mounted, Vibration, Sensor Rotation Estimation Sample 2

The next result (Figure 4.17) shows the sensor continuously registering rotation, even though the algorithm measures near constant (no rotation). The largest error here is again less then a degree (about 0.04°). Again this will be shown to not effect the general over-all image quality.

4.2.1.4 Tripod mount, vibration conclusion. Figure 4.18 shows clips of the corresponding images for the rotation estimations shown. It should be noticed that

none of these have any strange artifacts, or "ghost" images. Each image is very sharp, and most lines of the eye chart are easily readable. These images again are generated using the robust algorithm.



(a) Sample 1, Algorithm Reconstruction

(b) Sample 2, Algorithm Reconstruction

(c) Sample 1, Hybrid Reconstruction

(d) Sample 2, Hybrid Reconstruction

Figure 4.18. Tripod Mount, Vibration, Hybrid Registration Sample Reconstructions

This is one of the advantages of using the robust algorithm. Previously it was shown that the robust algorithm works well against error, and is able to produce clean images in the face of error. Here it is shown that even when the rotation estimates are clearly wrong (even in the wrong direction) then a satisfactory image can still be generated. Additionally, it is difficult to tell which images in Figure 4.18 are registered with the hybrid algorithm from the ones registered with the pure algorithmic approach.

4.2.1.5 Handheld, no vibration. The capture method of holding the phone in one's hand, rather than mounting it on a tripod will now be evaluated. In this case it is much easier for the phone to rotate, and these estimations become more

important to the quality of the overall image. This section will show however, that the resulting image quality relies on accurate rotation estimations very little; rather, quality restorations rely on well-focused and well-taken input images.

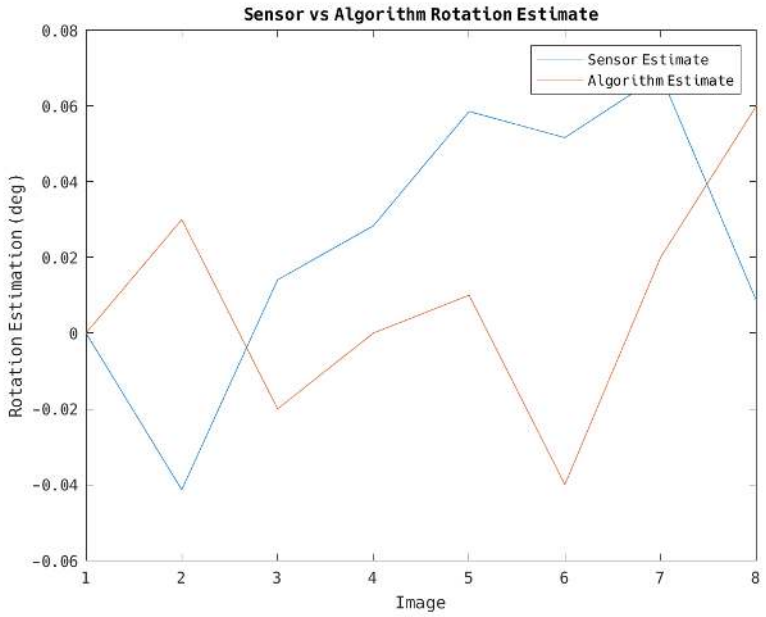


Figure 4.19. Hand-held, No Vibration, Sensor Rotation Estimation Sample 1

As the phone is taken off the tripod, the accuracy of the sensor measurements can be seen to decrease, often predicting the wrong direction frequently. Figure 4.19 shows a situation in which most of the rotations have been incorrectly predicted by the sensors (the maximum error here is about 0.1°). It will turn out however, that this estimation yields the best result in this section. This again does not point to a good estimate, but rather that another factor has a bigger influence on the final result.

To contrast this, Figure 4.20 shows another estimation, and this estimation is clearly much better. Five of the eight photos are registered in the correct direction, and the largest total error is also about 0.1°. The resulting image however for this, is much worse. This is due to a few of the input images being somewhat blurry. In this

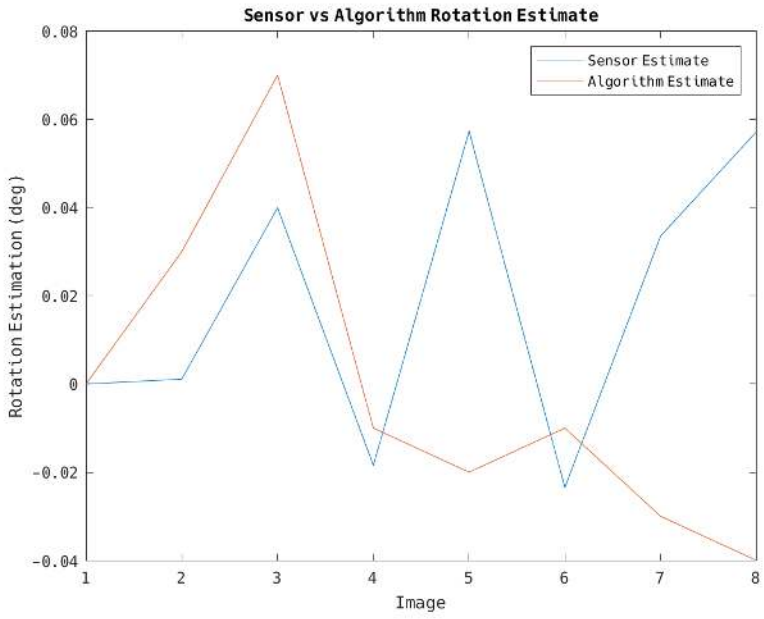


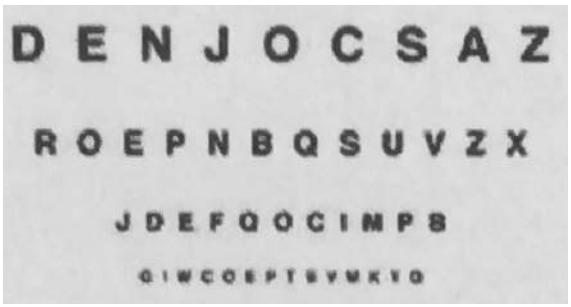
Figure 4.20. Hand-held, No Vibration, Sensor Rotation Estimation Sample 2

case it appears to be motion blur; however, other photos have the same issue, when the focus is poor.

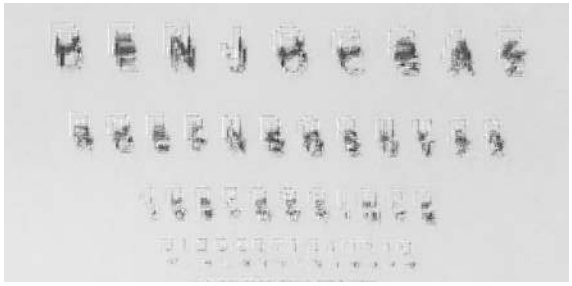
4.2.1.6 Handheld, no vibration conclusion. Figure 4.21 displays the previously mentioned photos. Specifically, Figure 4.21c shows the resulting image for the case in which most of the images had an incorrectly predicted direction, while on the other hand Figure 4.21d displays the picture which appears to have better rotation estimations. It is clear to see that the one with incorrectly predicted rotation is of better quality. This again does not show a failing of the estimation algorithms, but rather that their effect is minimal. As mentioned already, the cause of the terrible result in Figure 4.21d is rather the poor quality of the input images.

Further support for this idea is also evidenced by the fact that the algorithm and hybrid reconstructions produce nearly identical images.

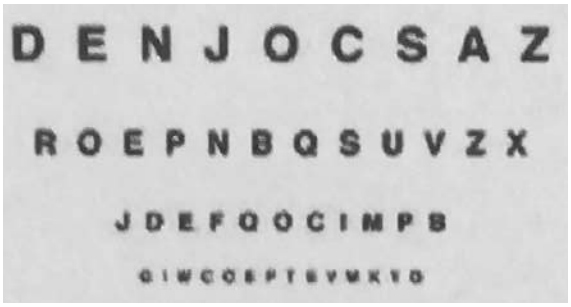
4.2.1.7 Handheld, vibration. As vibration is added to the handheld capture method, the results seem to support all previous results. The data captured in these



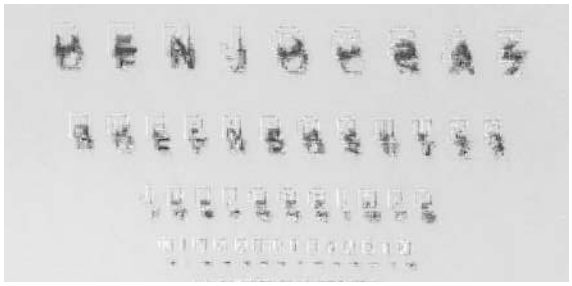
(a) Sample 1, Algorithm Reconstruction



(b) Sample 2, Algorithm Reconstruction



(c) Sample 1, Hybrid Reconstruction



(d) Sample 2, Hybrid Reconstruction

Figure 4.21. Hand-held, No Vibration, Hybrid Registration Sample Reconstructions

sessions have about the same amount of error as previously seen, and the end result seems to not be affected by any increase in error between trials. Any ill-constructed photos tend to also have blurry, unfocused, or motion blurred input images.

The first result (Figure 4.22) shows that all but one of the estimations are in the correct direction, and the error gets no larger than 0.25°. Just as before, this estimation is associated with the worst photo from this given set.

Finally, Figure 4.23 shows another sample from this set. Here quite a few of the images are estimated in the incorrect direction, and the error also never grows above 0.25°. In this case however, the picture produced is actually readable.

4.2.1.8 Handheld, vibration conclusion. As Figure 4.24 shows, it is clear that the first capture is rendered completely unreadable, while the second one remains well constructed. This is further evidence to suggest that the accuracy of the rotation

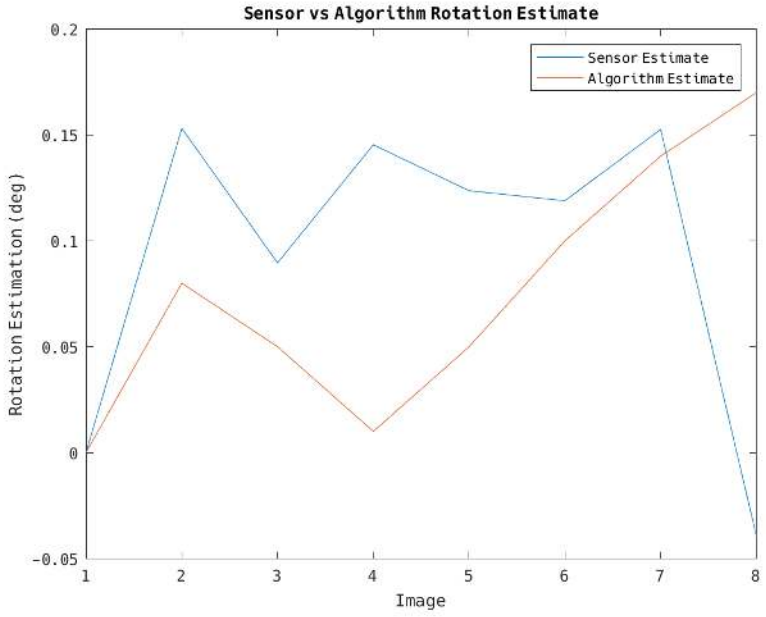


Figure 4.22. Hand-held, Vibration, Sensor Rotation Estimation Sample 1

estimate has little impact on the end result. Similarly in this case, the first set if input images contain a few with motion blur and loss of focus.

4.2.2 Rotational Estimation Conclusion

The data in this section supports the idea that the estimations created by the rotation estimation algorithm provided by Vandewalle, can be easily replaced by readings taken from the accelerometer and gyroscope of a phone (Patrick Vandewalle and Vetterli 2005). Another important result found in this section is that the overall quality of the image is drastically affected by the state and quality of the original input images. Remedying this situation will be considered later.

4.2.3 Translational Estimation Results

This section will consider the validity of the using the accelerometer on mobile devices to track the direction of the translational shifts found in photos. To review how this works, the acceleration found from the sensor is double integrated in order to obtain the position read. However due to the accelerometer being prone to noise,

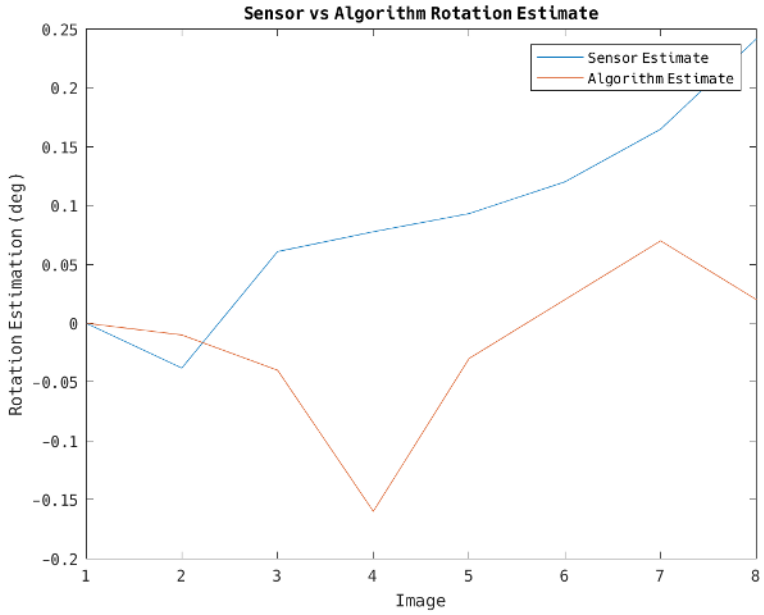


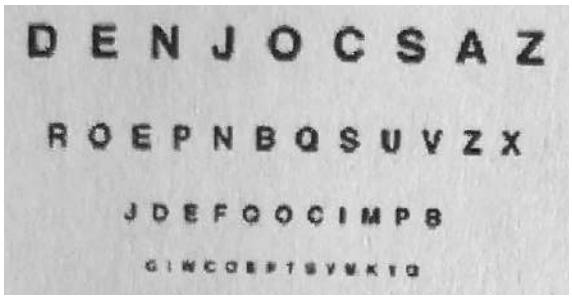
Figure 4.23. Hand-held, Vibration, Sensor Rotation Estimation Sample 2

and the double integration increasing error from these readings this position cannot be trusted precisely.

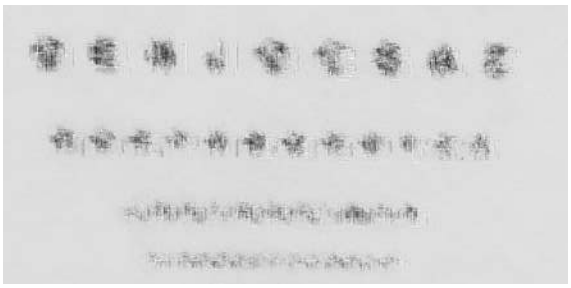
Previous research suggests that these readings can be used not to determine the distance traveled between photos, but rather only direction (Vsihwanath 2014). To accomplish this, the inverse tangent is used on the position obtained to turn this result into an angular estimation of direction traveled. An algorithm was developed from this that "slides" images along this direction until the maximum correlation is found. The results of these estimates will be explored.

In addition to comparing this sensor estimate against the algorithm estimate, the rotation angle predicted by the previous section is also used with the hybrid algorithm to make a prediction on the direction traveled. Finally it is worth noting that the samples used in the last section will be the same as used in this section, to allow for further evaluation if needed.

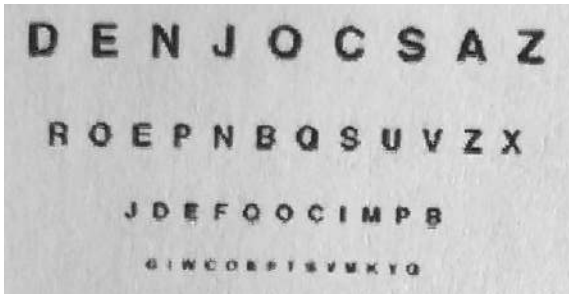
(a) Sample 1, Algorithm Reconstruction



(b) Sample 2, Algorithm Reconstruction



(c) Sample 1, Hybrid Reconstruction



(d) Sample 2, Hybrid Reconstruction

Figure 4.24. Hand-held, Vibration, Hybrid Registration Sample Reconstructions

4.2.3.1 Tripod mount, no vibration. Figure 4.25 shows the results of the sensor algorithm compared against the other previously mentioned methods. The pattern seen here is the pattern seen across almost all trials. The rotational estimates still provide a very accurate estimate of the direction of translational shift, while the sensor estimate tends to always predict the same angle for translation on all images.

The same result can be found in Figure 4.26; however, in this case the hybrid estimation yields an even more accurate result than previously seen. The error of the sensor algorithm in these cases is obviously very large, up to a maximum of about 100° in Figure 4.25.

4.2.3.2 Tripod mount, no vibration conclusion. The interesting result here, as Figure 4.27 shows, is that the sensor reconstruction image is still of high quality. The sensor reconstruction image cannot be easily identified without labels from the pure algorithmic reconstruction. This will later be shown to not be the case for every capture method; however, for this method the result is the same.

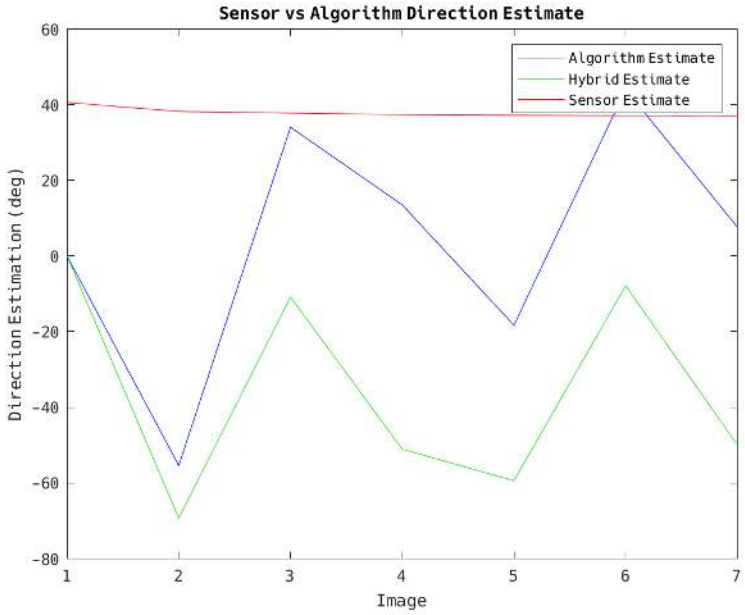


Figure 4.25. Tripod Mounted, No Vibration, Sensor Direction Estimation Sample 1

4.2.3.3 Tripod mount, vibration. Unsurprisingly, the sensor, hybrid, and algorithmic estimations of the direction of translation mirror the results found for tripod mounted without vibration. The sensor again predicts a constant direction for all images, and the hybrid and algorithmic approach predict the same directions. This again is expected, since the part of the hybrid algorithm that predicts the direction is exactly the same as the algorithmic approach.

Figure 4.28 is the typical case already described, and does not provide any particularly interested new insights. Figure 4.29 however shows a case when the hybrid prediction is very much different than the algorithmic approach. The error approaches a maximum of about 1.5°. While this is not a huge difference, it does yield a change in its predictions as shown in Figure 4.31. Fortunately, the graphs between the hybrid and algorithm still retain the same shape, even with their translational shifts being different. These shifts again are only minor errors, and therefore do not cause great differences in the resulting constructed image. This can also be verified by checking Figure 4.18c.

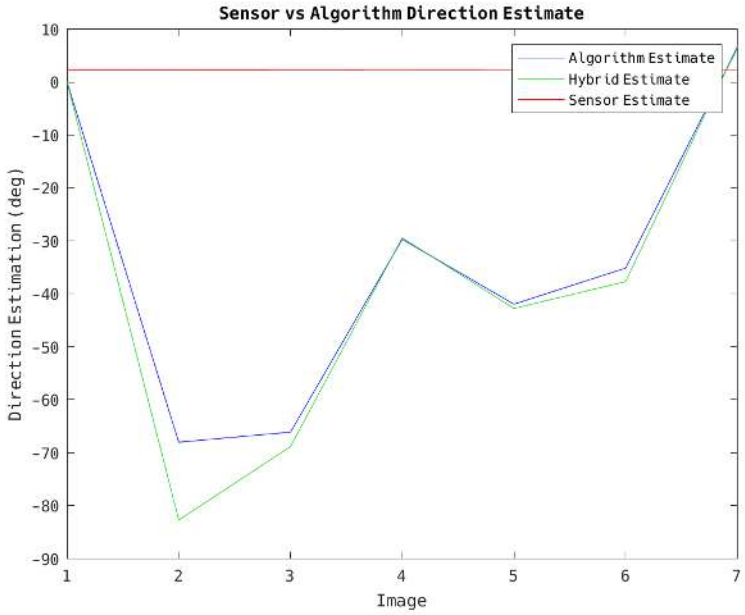
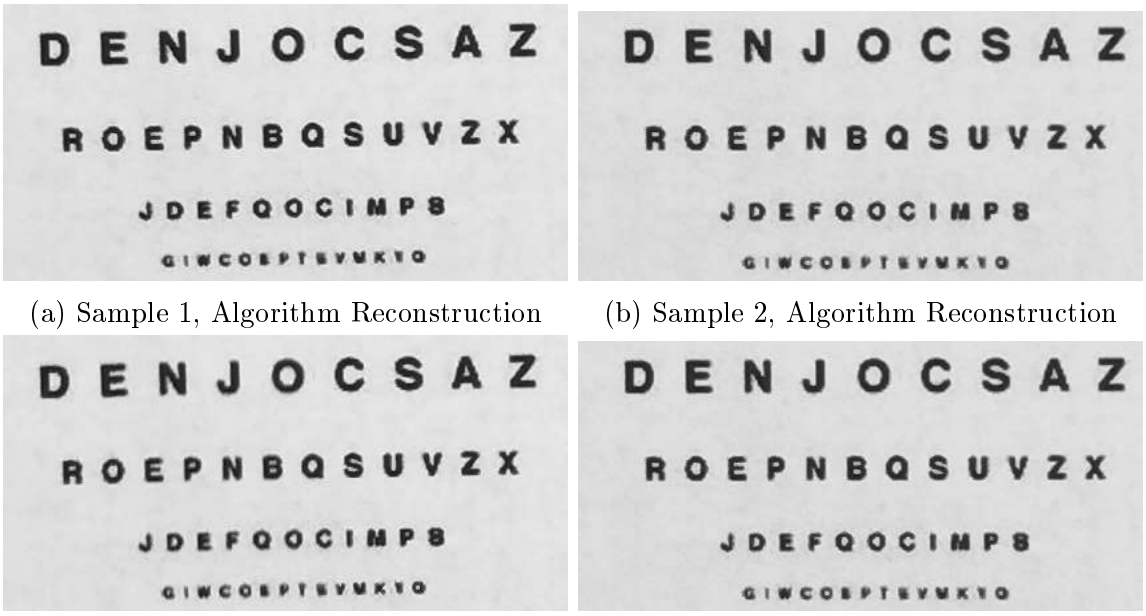


Figure 4.26. Tripod Mounted, No Vibration, Sensor Direction Estimation Sample 2

4.2.3.4 Tripod mount, vibration conclusion. The final results (Figure 4.32) show that the sensor-based translations are not very accurate as they cause a huge distortion in the image (Figure 4.32c). The text here has become unreadable, and is clearly not comparable to the trusted algorithm. Figure 4.32d however shows a satisfactory picture for the sensor prediction. Figure 4.30 and 4.31 show why this occurs. The direction predicted for the sample 2 is good enough to allow a decently close prediction of the shift values. Since these shift values generally follow the same curve as the algorithmic approach, a good reconstruction can occur. The takeaway here is that the sensor predictions for direction of translation are not entirely reliable.

4.2.3.5 Handheld, no vibration. When the capture method is changed to the phone being handheld without vibration, the hybrid shift registration becomes strikingly accurate. Figure 4.33 is the first evidence of this. In this it can be seen that the hybrid estimate sticks to the algorithm at every point. Even the points that look to be completely different are really the same angle (about 90°) just in different



(a) Sample 1, Algorithm Reconstruction

(b) Sample 2, Algorithm Reconstruction

(c) Sample 1, Sensor Reconstruction

(d) Sample 2, Sensor Reconstruction

Figure 4.27. Tripod Mount, No Vibration, Sensor Registration Sample Reconstructions

directions. Since this angle will be swept in both directions, this direction estimate is actually spot on. To show this, the x and y shift estimates are shown in Figure 4.34 (notice how the algorithm and hybrid estimate stick to each other at practically every point). Again the sensor estimate here predicts a constant directional shift for each image.

The next sample (Figure 4.35) gives yet another example of the hybrid algorithm producing very favorable results. The lines for the algorithm and hybrid estimate in this graph are almost indistinguishable from each other, while the sensor again fails to give any good estimate for direction.

4.2.3.6 Handheld, no vibration conclusion. So far this section has shown that the hybrid algorithm serves well in estimating translational shifts, while the pure sensor method fails to give any good result. In this section the reconstruction result will confirm this as well.

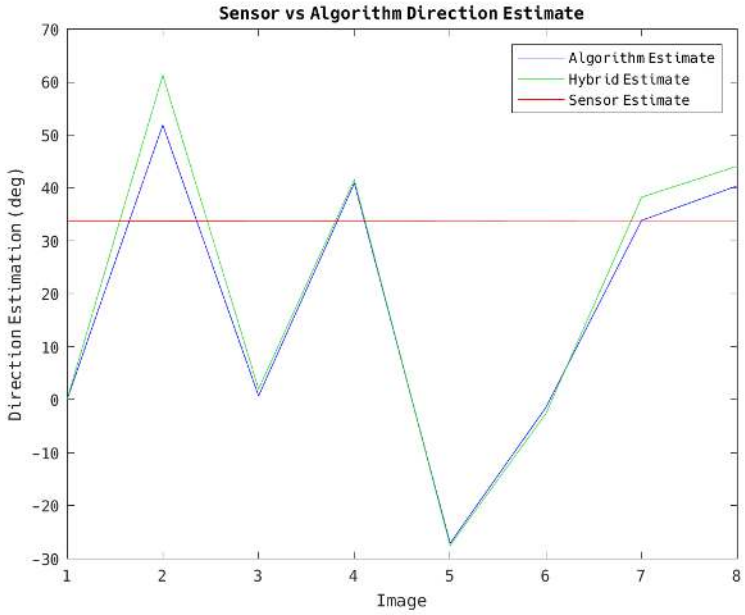


Figure 4.28. Tripod Mounted, Vibration, Sensor Direction Estimation Sample 1

Clearly according to Figure 4.36 the sensor methods fail to produce any good result. Both sensor registered images are blurred and unreadable, suggesting further that sensor estimation is not reliable and not very useful. Figure 4.36d gives a better image than its algorithm counter-part; however, this is still not very useful, as the end goal of this research is to create higher-quality photos on mobile devices. This does support previous IMU registration research, which uses accelerometer information to fairly accurately register blurred images (Vsihwanath 2014).

4.2.3.7 Handheld, vibration. Finally, the handheld vibration case is evaluated in this section. The results in this summary are mostly a confirmation of the previous section, that pure sensor based direction prediction fails to deviate from a constant prediction, while the hybrid and algorithmic approach give almost the same estimate at all points.

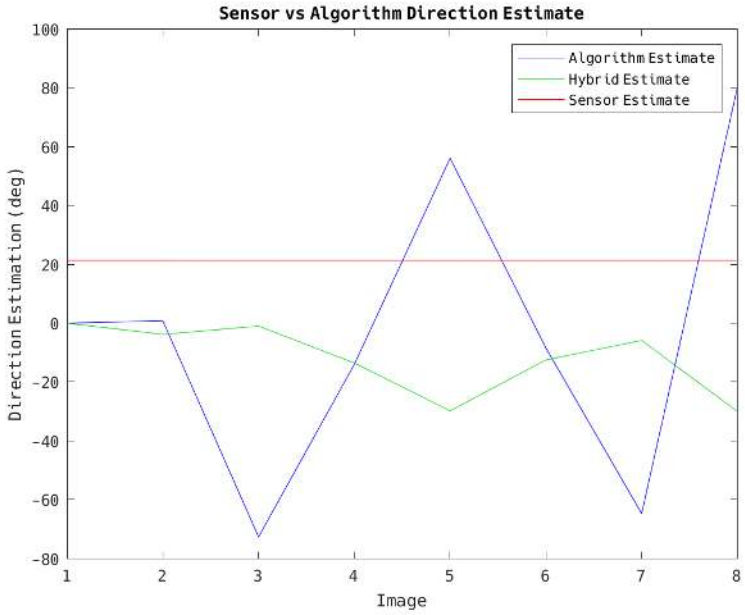


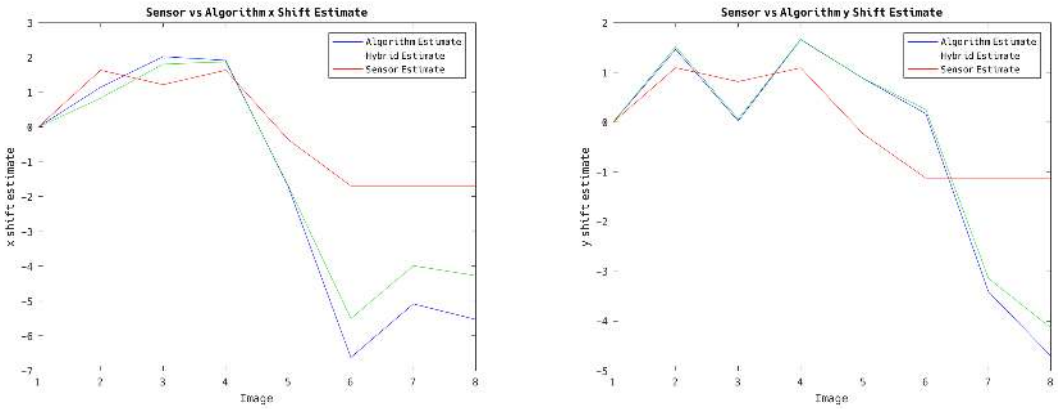
Figure 4.29. Tripod Mounted, Vibration, Sensor Direction Estimation Sample 2

Figures 4.37 and Figure 4.38 both give extremely similar results. The hybrid follows the algorithm at every point except for the 7th image in Figure 4.38. Fortunately this does not affect the end result at all (Figure 4.24d) since using a robust algorithm severely reduces the impact of errors like this.

4.2.3.8 Handheld, vibration conclusion. The sensor algorithm in Figure 4.39c proves to actually register the images better than the algorithm (as the reconstruction is slightly clearer). This again supports previous research in registering blurry images using IMU sensors (Vsihwanath 2014). While this is an interesting result, it does not help a great deal in the end goal of this research. Figure 4.39d shows that the sensors fail to properly register non-blurred images.

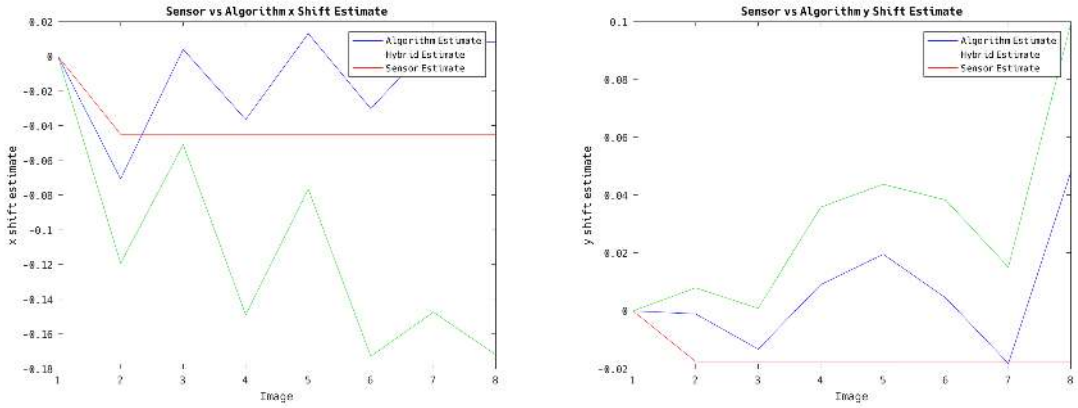
4.2.4 Translational Estimation Conclusion

Throughout this section it was shown that the sensor fails to universally predict even the correct direction of translational shift. Because of this, it is unlikely to be very useful in computing a quick and accurate super resolution image on a mobile



(a) x-axis predictions (b) y-axis predictions

Figure 4.30. Tripod Mounted, Vibration, Sample 1 Shift Estimations

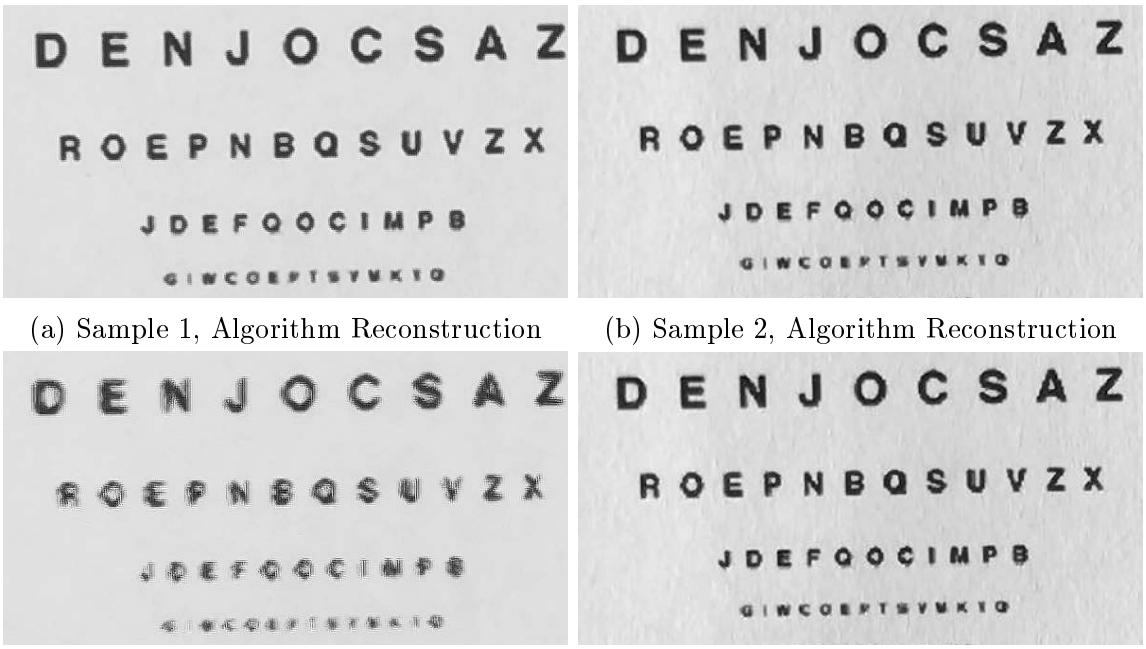


(a) x-axis predictions (b) y-axis predictions

Figure 4.31. Tripod Mounted, Vibration, Sample 2 Shift Estimations

device. Evidence was given however for the sensors being able to register blurry images well, and even aided in the reconstruction of blurry images.

The hybrid algorithm's usefulness was supported in this section as well, as it was shown to yield extremely comparable results to the pure algorithmic approach to finding translational shifts. This section shows that complete sensor dependence for estimating registration parameters is not feasible, though a hybrid approach is definitely usable.



(a) Sample 1, Algorithm Reconstruction (b) Sample 2, Algorithm Reconstruction

(c) Sample 1, Sensor Reconstruction (d) Sample 2, Sensor Reconstruction

Figure 4.32. Tripod Mount, Vibration, Sensor Registration Sample Reconstructions

4.2.5 Hybrid Speed Improvement

Here a quick analysis of the hybrid algorithm's speedup will be given. This algorithm greatly improves upon the time it takes to register a set of eight images, as the algorithm presented by Vandewalle takes quite some time (Patrick Vandewalle and Vetterli 2005).

Due to each type of capture method capturing a possibly different set of photos, each case is evaluated individually. All algorithm and hybrid values are in seconds, speedup is unitless.

As Figure 4.40 shows, the hybrid algorithm gives a considerable universal speedup. While this looks very impressive, non-default parameters were passed into the algorithm given by Vandewalle in order to better compare it against the rotation algorithms. The hybrid algorithm should yield a speedup in any case however, as it can be computed while capturing images, reducing the strain on the post-processing of input images.

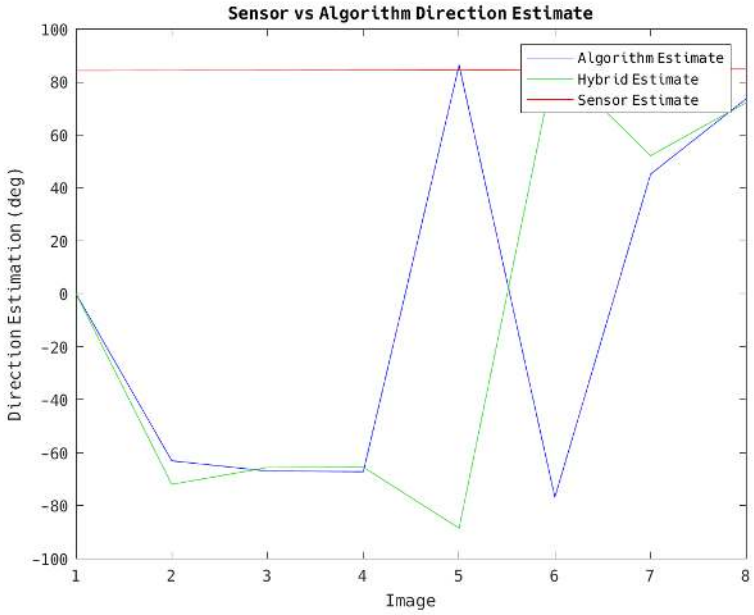


Figure 4.33. Hand-held, No Vibration, Sensor Direction Estimation Sample 1

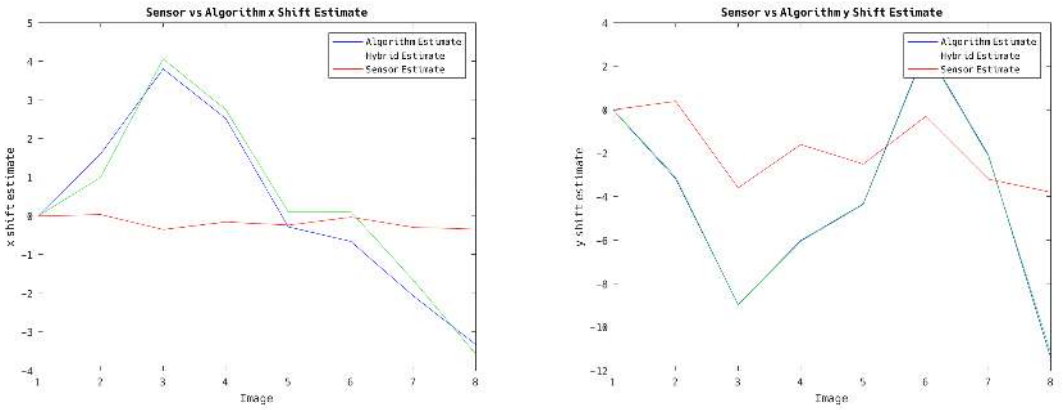
4.3 Input to Output Comparison

For completeness, here are a few examples of a single input image compared against its hybrid registered and robust reconstructed counterpart.

Some of these images look worse from input to output. This occurs for several reasons. This can occur when not enough phase offsets are present in the input images, as there is net enough data to fill in the missing samples accurately. Additionally, many of these phone cameras are already of such quality that very little improvement can be made. In particular little improvement can be made as the sample signal (the paper object) is itself a digital printing. It might be a better idea in the future to use a different object as a sample signal.

4.4 Conclusion

In this chapter relevant data was evaluated and the best algorithms and scheme for computing super-resolution on a phone was considered. For the reconstruction algorithm, the robust solution presented appeared to give the best reconstructed image



(a) x-axis predictions (b) y-axis predictions

Figure 4.34. Tripod Mounted, Vibration, Sample 1 Shift Estimations

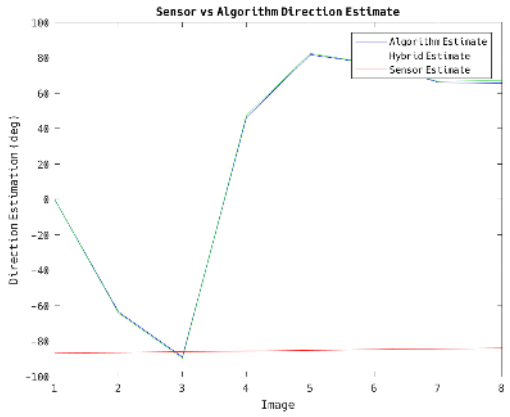
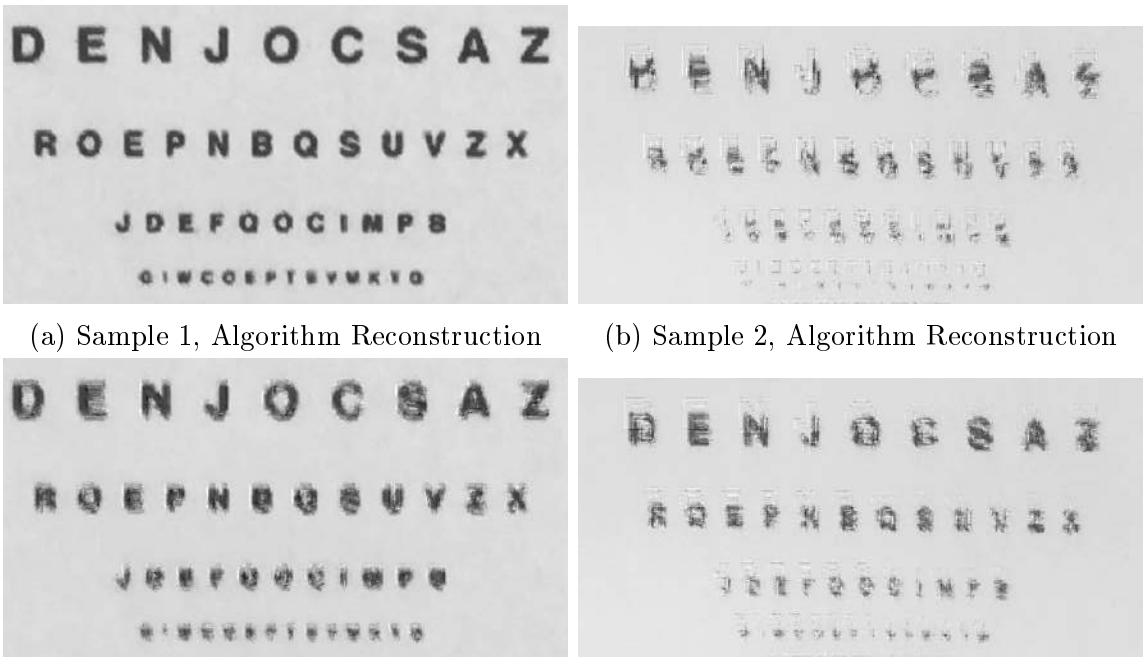


Figure 4.35. Hand-held, No Vibration, Sensor Direction Estimation Sample 2

most consistently. After this, the registration algorithms based on sensor feedback were considered. Here it was shown that the rotation retrieved from these sensors was useful for replacing algorithms to do the equivalent measurement. Using sensors to predict the direction of translation between photos was shown to be inaccurate for complete sensor estimation. The hybrid algorithm however showed to be very accurate at predicting the translational direction. Finally, the hybrid registration also was shown to yield a speed up over the algorithmic solution.



(a) Sample 1, Algorithm Reconstruction

(b) Sample 2, Algorithm Reconstruction

(c) Sample 1, Sensor Reconstruction

(d) Sample 2, Sensor Reconstruction

Figure 4.36. Hand-held, No Vibration, Sensor Registration Sample Reconstructions

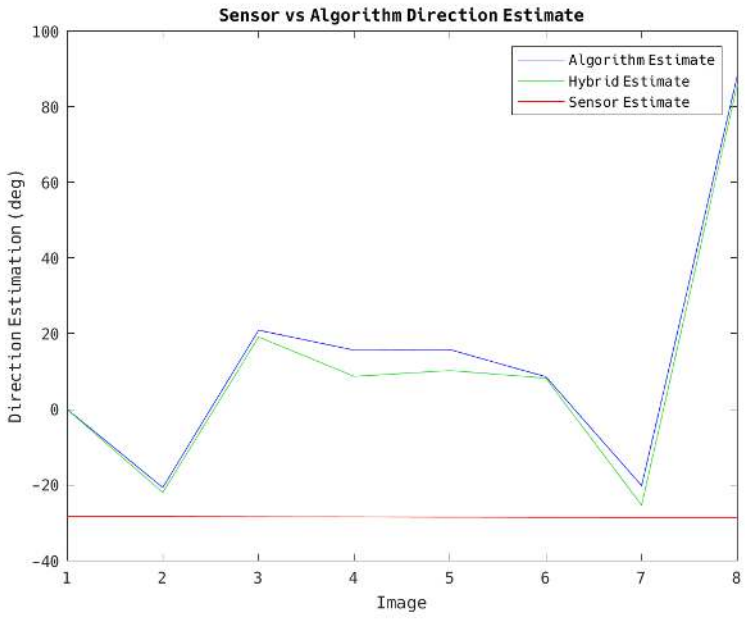


Figure 4.37. Hand-held, Vibration, Sensor Direction Estimation Sample 1

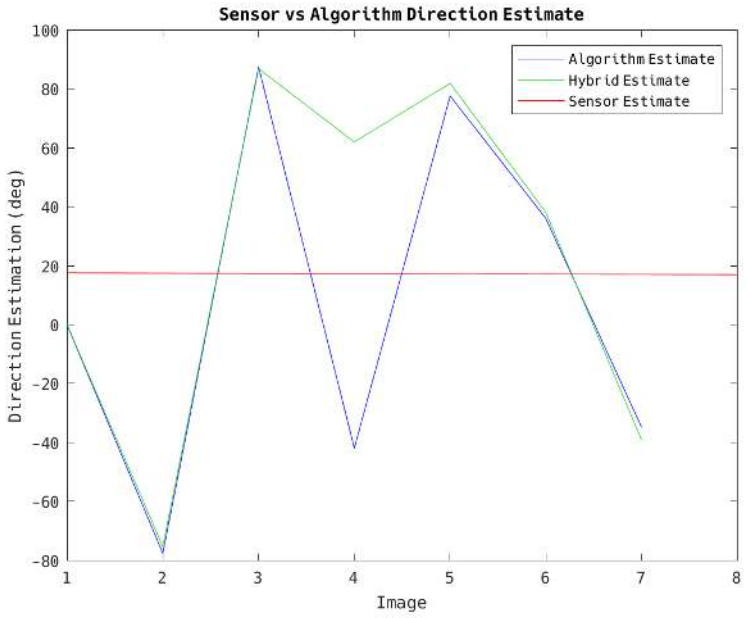
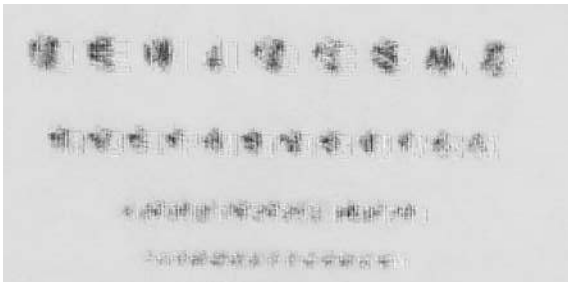
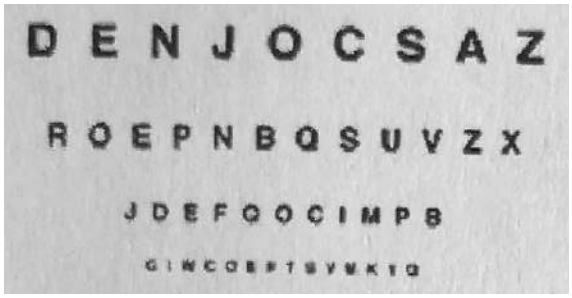


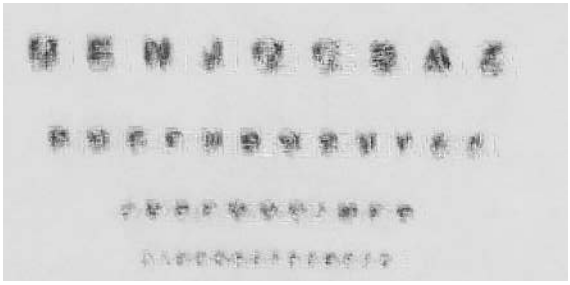
Figure 4.38. Hand-held, Vibration, Sensor Direction Estimation Sample 2



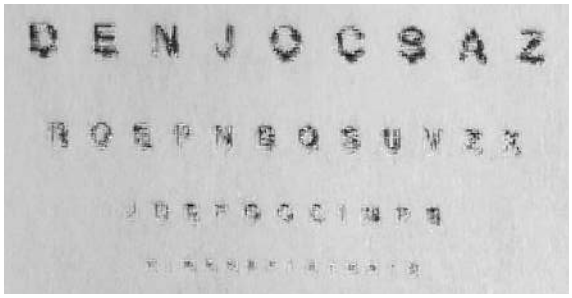
(a) Sample 1, Algorithm Reconstruction



(b) Sample 2, Algorithm Reconstruction



(c) Sample 1, Sensor Reconstruction



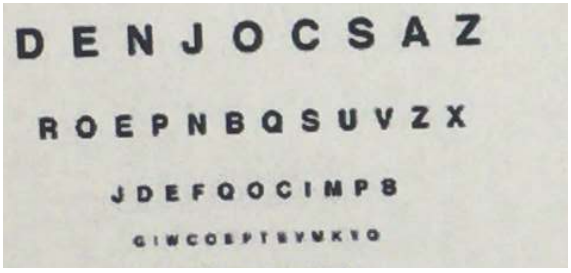
(d) Sample 2, Sensor Reconstruction

Figure 4.39. Hand-held, Vibration, Sensor Registration Sample Reconstructions

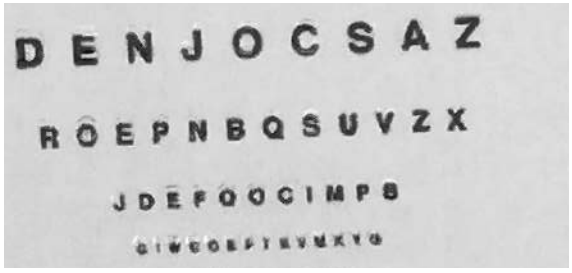
	Tripod Mount, No Vibration	Tripod Mount, Vibration
Algorithm	70.46	97.80
Hybrid	3.83	4.10
Speedup	16.13	30.52

	Handheld, No Vibration	Handheld, Vibration
Algorithm	75.29	97.56
Hybrid	3.89	4.03
Speedup	319.01	360.74

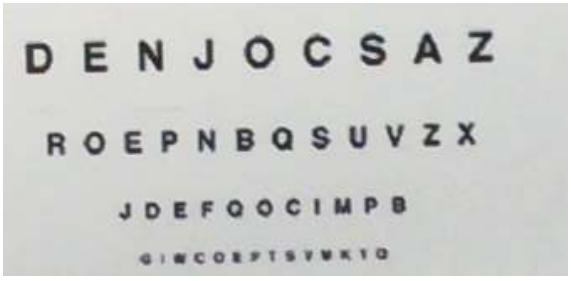
Figure 4.40. Average time and speedup for given algorithm



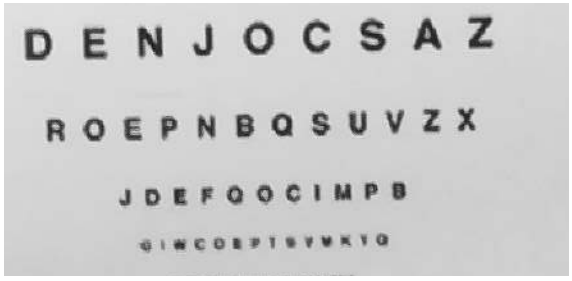
(a) Tripod Mount, No Vibration Input



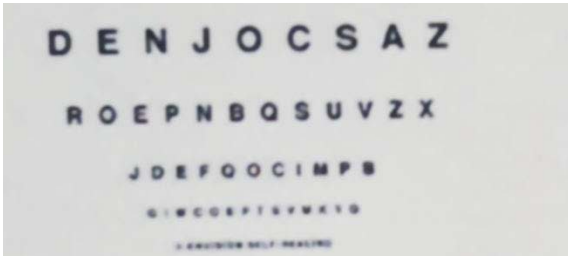
(b) Tripod Mount, No Vibration Output



(c) Tripod Mount, Vibration Input

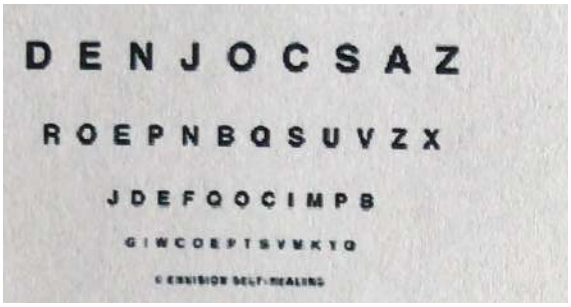


(d) Tripod Mount, Vibration Output

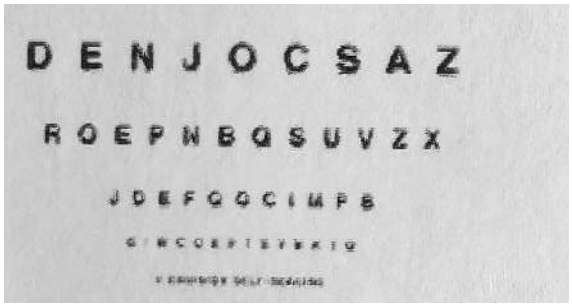


(e) Hand-held, No Vibration Input

(f) Hand-held, No Vibration Output



(g) Hand-held, Vibration Input



(h) Hand-held, Vibration Output

Figure 4.41. Input to Output Comparison

CHAPTER FIVE

Summary

This chapter serves two purposes, to summarize what work was done in this research, and to speak to the next logical places to take this research. Most of this research focuses on using IMU sensors on phones to quickly register images for the purposes of super resolution. This research has also sparked more ideas for practically accomplishing super-resolution on mobile devices. These ideas will be outlined here.

5.1 Tools Developed

For this research several tools were developed. The back-end tool that was used was an Azure storage account. This allowed for quick and easy development of a client application. Specifically, the application used Azure Mobile Services with "Easy Tables". The Azure storage account was also used for blob storage of all the burst images which were captured.

A separate application was developed for mobile phones to utilize this back end. The application was developed for both iOS and Android devices, using the Xamarin platform. This platform allows for all native components of the devices to be accessed, and allowed the application to share code and be developed in the same project. The application was designed to take burst photos with and without the vibrator in the phone being activated during capture time. Additionally, the phone recorded various sensor data during the capture time, and uploaded all data to the Azure backing store.

Once the application was developed, it was used on volunteer's phones to collect data from various phones. Each phone was mounted to a tripod, and aimed at a printout of various registration marks and images. 4 different captures were taken,

specifically combinations of the phone being mounted to a tripod, or in the user's hand, as well as with and without vibration.

After data collection, another Node.js application was developed for retrieving the data from the Azure database. Node.js was used as it was the only cross-platform solution to accessing the data in the Azure backing store. This small application simply queries the database, and stores all relevant readings in CSV files on the host computer.

Matlab was used for all data analysis, utilizing many scripts and functions provided by the Vandewalle research (Patrick Vandewalle and Vetterli 2005). The data analysis scripts developed ran through each session the phone captured, and used this data to register each set of images in three different ways. The same set of photos were also used with four different reconstruction algorithms. The combination of the registration and reconstruction algorithms yielded 12 different photos from each session. Additional useful information and graphs are recorded in the scripts.

5.2 Results

The data just mentioned was then pulled apart piece by piece, in order to discover what was useful from the research conducted. Initially all data was picked apart to determine what was the best reconstruction algorithm to use for quality. The robust solution was determined to have the highest universal quality, and was the most resilient to error and noise in the system. This algorithm was then used in evaluating the other results from the experiment.

Each sensor-based registration algorithm was then considered, comparing them to the pure algorithmic solution. Here the hybrid algorithm was shown to estimate rotation in a satisfactory manner, and more importantly almost matched the pure algorithm in translational shift prediction. This was a valuable discovery as this means the rotation of each photo can be determined while capturing the photos, rather than evaluating the photos afterwards for rotational shifts.

Accelerometers in the phones also were used to attempt to track a phone's translational movement between photos. Due to the high amount of noise present in accelerometers, and the growing error from double integrating this signal, this data proved to be fruitless. The technique used however had already been shown to accurately estimate the registration of blurred images, and a similar result was found in this research (Vsihwanath 2014).

5.3 Future Work

Here is some interesting work that can be done to improve upon the research presented here. Many of these things should help overcome challenges already seen in this project, and would be steps towards making super-resolution relevant on mobile devices.

5.3.1 Pre-Processing

One of the biggest issues seen for poorly constructed images involved the original input images being blurry or out of focus. Several things can help remedy this. To begin with, more advanced burst cameras can be coded for iOS and Android. Both of these platforms expose fine grained control over the camera that could allow for better focusing and initial image quality.

Additionally, photos can be processed before they are put through the super-resolution routine. Motion blur can be detected and corrected accurately since motion data are already being collected. Additionally, motion blur should be easy to fix and the accelerometer should be able to provide more useful data for this purpose, as it does not need to be double integrated to estimate the direction of the blur.

5.3.2 Photo Selection

In addition to pre-processing each photo, another tool that can be used to gather higher quality input images is to only select good photos. With this scheme one can collect as many photos as the phone will allow, and then parse through the

photos for those that do not contain blur, or ones with a favorable phase offset for super-resolution.

Combining this scheme with some pre-construction image processing should allow for a very favorable set of input images to be used with the super-resolution algorithm. When this is combined with a robust algorithm consistently improved images should not be hard to achieve.

5.3.3 Reconstruction Algorithm Speedup

Another very important area to investigate is speeding up the reconstruction algorithms. These are currently the bottleneck of the super-resolution process. For mobile specifically, evaluating how the GPU can be leveraged, and to what degree it can speed up the algorithms is critical to computing these photos on the phone. An alternative to this would be to host an online service with a more powerful computer to do the computing. Nonetheless, achieving this on a mobile device is an interesting problem.

Other algorithms can be explored as well, such as straight interpolation algorithms. These algorithms are likely to be faster on mobile devices; however, they probably would not be as resilient as the robust solution presented. It may be possible to combine some of the ideas of the interpolating algorithms with some of the resilience of the robust algorithm used in this research. This would certainly improve the runtime of the algorithms, hopefully without loosing quality.

5.3.4 Registration Improvement

Finally, further investigation can be put into the registration schemes presented here. While this research presents a good first pass at combining sensor data and algorithms, other interesting hybrid algorithms can likely be discovered. For instance, the hybrid algorithm presented here simply uses some sensor data as a replacement for part of an algorithm. Instead of this, the sensor data can be used as a "suggestion" to

the algorithm, and serve as a good base estimate, allowing the algorithm to converge on a more accurate estimate in less time.

Additionally, more research can be put into filtering the accelerometer and gyroscope data to obtain better data readings. This would certainly vastly improve the value of this data, and allow more interesting algorithms to be explored.

BIBLIOGRAPHY

Park, S. C., M. K. Park, and M. G. Kang (2003, May). Super-resolution image reconstruction: a technical overview. *IEEE Signal Processing Magazine 20*(3), 21–36.

Patrick Vandewalle, S. S. and M. Vetterli (2005). A frequency domain approach to registration of aliased images with application to super-resolution. *EURASIP Journal on Applied Signal Processing*.

Romano, Y., J. Isidoro, and P. Milanfar (2016). RAISR: rapid and accurate image super resolution. *CoRR abs/1606.01299*.

Vsihwanath, S. R. V. (2014). Utilizing motion sensor data for some image processing applications. Master's thesis, Indian Institute of Technology Madras.