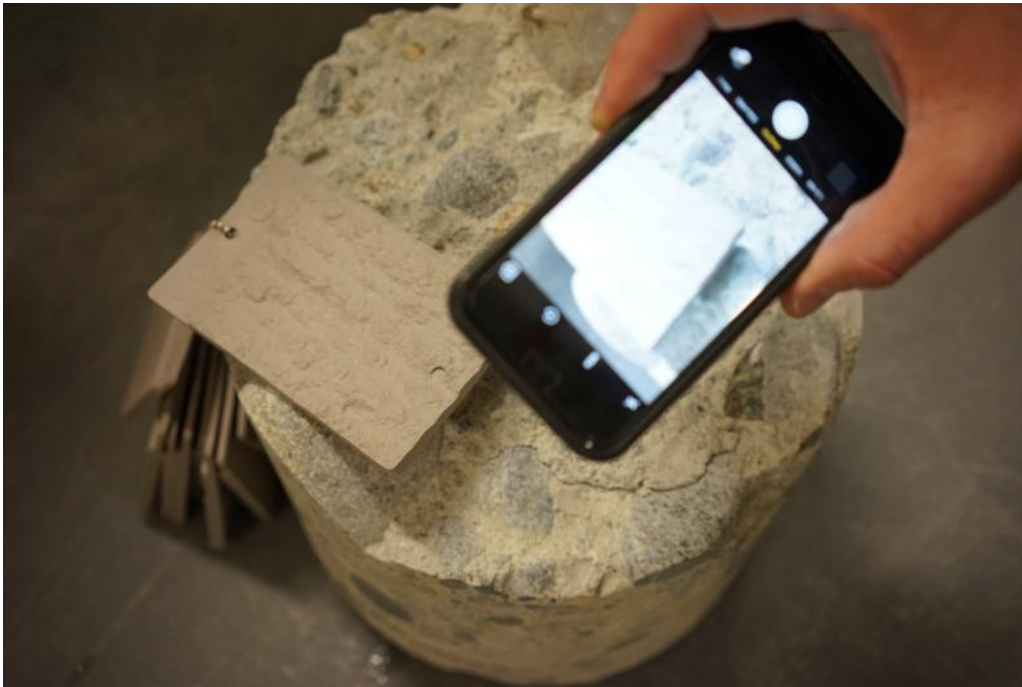


# RECLAMATION

*Managing Water in the West*

## Photogrammetry App for Roughness Profile

Research and Development Office  
Science and Technology Program  
(Final Report) ST-2015-1382-01



U.S. Department of the Interior  
Bureau of Reclamation  
Research and Development Office

December 2015

# **Mission Statements**

The U.S. Department of the Interior protects America's natural resources and heritage, honors our cultures and tribal communities, and supplies the energy to power our future.

The mission of the Bureau of Reclamation is to manage, develop, and protect water and related resources in an environmentally and economically sound manner in the interest of the American public.

**REPORT DOCUMENTATION PAGE**

*Form Approved  
OMB No. 0704-0188*

<b>T1. REPORT DATE</b> December 2015		<b>T2. REPORT TYPE</b> Research		<b>T3. DATES COVERED</b>	
<b>T4. TITLE AND SUBTITLE</b> Photogrammetry App for Roughness Profile				<b>5a. CONTRACT NUMBER</b> 15XR0680A1-RY1541IR201511382	
				<b>5b. GRANT NUMBER</b>	
				<b>5c. PROGRAM ELEMENT NUMBER</b> 1541 (S&T)	
<b>6. AUTHOR(S)</b> Matthew Klein, Ph.D. mjklein@usbr.gov 303.445.2368  Mark Travers mtravers@usbr.gov 303.445.2340				<b>5d. PROJECT NUMBER</b> 1382	
				<b>5e. TASK NUMBER</b>	
				<b>5f. WORK UNIT NUMBER</b> 86-68530	
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> Concrete, Geotechnical and Structural Laboratory U.S. Department of the Interior, Bureau of Reclamation PO Box 25007, Denver CO 80225				<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b> MERL-2015-105	
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b> Research and Development Office U.S. Department of the Interior, Bureau of Reclamation, PO Box 25007, Denver CO 80225-0007				<b>10. SPONSOR/MONITOR'S ACRONYM(S)</b> R&D: Research and Development Office BOR/USBR: Bureau of Reclamation DOI: Department of the Interior	
				<b>11. SPONSOR/MONITOR'S REPORT NUMBER(S)</b> ST-2015-1382-01	
<b>12. DISTRIBUTION / AVAILABILITY STATEMENT</b> Final report can be downloaded from Reclamation's website: <a href="https://www.usbr.gov/research/">https://www.usbr.gov/research/</a>					
<b>13. SUPPLEMENTARY NOTES</b>					
<b>14. ABSTRACT</b> An alternative to traditional methods of determining roughness of concrete for repair preparation is to use a photogrammetry smartphone app. The app would be installed on a smartphone that can take pictures and process them to create a model. The model is compared against standardized Concrete Surface Profile replicas to find the best match and then report the Concrete Surface Profile number.					
<b>15. SUBJECT TERMS</b> Photogrammetry, Roughness profile, 3D modeling, Direct shear					
<b>16. SECURITY CLASSIFICATION OF:</b>			<b>17. LIMITATION OF ABSTRACT</b>  U	<b>18. NUMBER OF PAGES</b>  18	<b>19a. NAME OF RESPONSIBLE PERSON</b> Matthew Klein
<b>a. REPORT</b> U	<b>b. ABSTRACT</b> U	<b>c. THIS PAGE</b> U			<b>19b. TELEPHONE NUMBER</b> 303-445-2368

# PEER REVIEW DOCUMENTATION

## Project and Document Information

Project Name: Photogrammetry App for Roughness Profile      WOID: Z1382

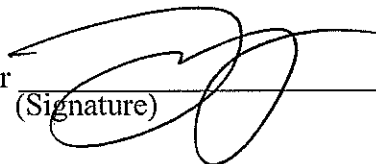
Document: Photogrammetry App for Roughness Profile

Document Author: Matthew Klein and Mark Travers      Document date: 12/21/2015

Peer Reviewer: Kurt von Fay

## Review Certification

**Peer Reviewer:** I have reviewed the assigned items/sections(s) noted for the above document and believe them to be in accordance with the project requirements, standards of the profession, and Reclamation policy.

Reviewer       Date reviewed 12/23/15  
(Signature)

**BUREAU OF RECLAMATION**  
**Technical Service Center, Denver, Colorado**  
**Concrete, Geotechnical, & Structural Laboratory Group, 86-68530**

**Technical Memorandum No. MERL-2015-105**

# Photogrammetry App for Roughness Profile

Mark Travers  
 Prepared by: Mark Travers  
 Civil Engineer, Concrete, Geotechnical, & Structural Laboratory, 86-68530

12/23/15  
 Date

Matthew Klein  
 Checked by: Matthew Klein, Ph.D.  
 Civil Engineer, Concrete, Geotechnical, & Structural Laboratory, 86-68530

12/23/2015  
 Date

Kurt von Fay  
 Peer Review by: Kurt von Fay, BSCE, MBA  
 Civil Engineer, Concrete, Geotechnical, & Structural Laboratory, 86-68530

12/23/15  
 Date

REVISIONS					
Date	Description	Prepared	Checked	Technical Approval	Peer Review

## **Disclaimer**

Information in this report may not be used for advertising or promotional purposes. The data and findings should not be construed as an endorsement of any product or firm by the Bureau of Reclamation, Department of the Interior, or Federal Government. The products evaluated in the report were evaluated for purposes specific to the Bureau of Reclamation mission. Reclamation gives no warranties or guarantees, expressed or implied, for the products evaluated in this report, including merchantability or fitness for a particular purpose.

# Executive Summary

The use of traditional methods to measure and quantify the roughness of a concrete surface that was prepared for repair are subjective and/or expensive. These include comparing the surface to Concrete Surface Profile (CSP) blocks or replicas and the use of Laser profilometers.

One potential alternative is to use photogrammetry to determine the concrete surface profile. This procedure would involve the use of a smartphone camera to capture a series of images which would be processed and compared to the standardized CSP replicas for a match giving the resultant CSP number (1 smooth to 10 very rough).

This project was used to learn what would be required to use a smartphone to provide the CSP number. The photogrammetric process was detailed to identify the algorithms required to complete the analysis. Once the photogrammetry was performed, the model would need to undergo additional analysis to provide the result. Different platforms were investigated along with the software developer package that is commonly used to write smartphone apps.

While the project outlines the steps to successfully develop an app to determine the roughness of a concrete surface in preparation for a successful repair, programming expertise within Reclamation is limited to implement the project. In order to develop the app, a contractor would need to be hired to write and debug the code and demonstrate the app in different environments for reliability and accuracy.

# Contents

	<i>Page</i>
<b>Review Certification.....</b>	<b>3</b>
<b>Disclaimer .....</b>	<b>6</b>
<b>Executive Summary .....</b>	<b>7</b>
<b>Contents .....</b>	<b>8</b>
<b>Introduction.....</b>	<b>9</b>
<b>Photogrammetry .....</b>	<b>10</b>
<b>Roughness Profile Modeling .....</b>	<b>10</b>
<b>Photogrammetry App for Smartphones .....</b>	<b>11</b>
<b>Conclusions.....</b>	<b>15</b>
<b>References .....</b>	<b>17</b>



# Introduction

When concrete damage in a structure progresses far enough, repairs will become necessary. Once the affected area has been identified, it can be removed and prepared for replacement. The surface preparation of the old concrete is essential in providing a successful repair. One of the ways to measure the acceptable surface preparation is by determining its roughness (Bissonnette, Vaysburd, & von Fay, 2012). Consequently, when adding new concrete to a deteriorated member, there must be a reliable and easy to use method of determining the roughness of the deteriorated surface to optimize the bond between the old and new concrete.

One of the methods repair workers are employing include non-quantitative methods of determining the roughness of a surface. This method includes a set of rubber blocks, called Concrete Surface Profile (CSP) replicas, molded to the shape of different preparation roughnesses. The blocks are ranked from 1 to 10 according to their roughness – 1 being very smooth and 10 being very rough. The worker compares the surface of the concrete surface to the CSP replicas. Several other techniques are also used, including a technique known as Laser Profilometry. This technique utilizes an array of laser range finders to create a point cloud representing a surface much like a short range LIDAR. However, these devices are can be expensive (International Concrete Repair Institute, 2013, pp. 40-47).

This report outlines an alternative technique which has not been applied to this task and may solve the problems of cost and user subjectivity. The technique of photogrammetry uses data from digital images taken from inexpensive cameras processed through a computer program to develop a 3D model of the surface. In recent years, increasingly powerful computers and high quality digital cameras have popularized the technique of photogrammetry to map a surface or object. Armed with a highly-accurate point cloud describing a surface, it is possible to calculate a quantitative measurement of roughness.

To provide repair workers a practical way to utilize this technique, the process should usable by a a small, handheld device. Portable computing systems like smartphones have increased in power and capability in addition to incorporating quality optics for detailed digital imagery. In theory, an app written for a mobile phone such an iPhone or Android could perform this task. The procedure would involve taking a series of pictures within a specially written application that would process the digital images. The resulting model would be compared against a model of each of the CSP profile standards and the CSP number assigned based on the closest match.

# Photogrammetry

The photogrammetry workflow includes three main steps: feature extraction and categorization, alignment of photos with similar features, and triangulation of the dense point cloud. Feature extraction utilizes an algorithm known as the Scale Invariant Feature Transform or SIFT to detect and describe local features in images. This transform rides on the back of the Hough transform which searches for features in an image by trying to fit lines to similar points (Duda & Hart, 1971) (Fegyver, 2014). Key points are then defined as the center point in each of these lines.

Once features in a set of images are recognized and matched to each other, the respective coordinates of each feature in each picture are plugged into a collinearity equation to solve for the x, y, and z position of that feature and the x, y, and z positions and the  $\omega$  (pitch),  $\phi$  (yaw), and  $\alpha$  (roll) rotations of each picture relative to the others. This process includes an iterative least squares operation limiting the accuracy of the final result (Mingt Lu, 2011, pp. 18-25).

With the relative positions of the cameras and a few basic points calculated, it is a matter of repetitive triangulation to build a point cloud to the desired density.

Early applications of photogrammetry included processes like aerial survey and earth crustal movement detection (Tewinkel, 2011). Points were chosen manually and the photogrammetric calculations were performed by hand. The linear algebra and least squares adjustment in the process made each point calculation a formidable process. With the advent of modern computers, it became much more feasible to calculate point clouds with thousands or even millions of points.

## Roughness Profile Modeling

After the photogrammetric process has been completed, there remains one more task for this specific application. The point cloud generated by photogrammetry needs to be analyzed to return a CSP number represented by the CSP replicas mentioned above (International Concrete Repair Institute, 2013, p. 42). There are several possible methods for this process as discussed below.

A cross-section perpendicular to the surface being measured should be extracted from the point cloud. This data should then be fitted with a regression line. The roughness can then be expressed as the standard deviation of the data set. The standardized CSP replicas would also be analyzed in a similar manner with corresponding roughness in terms of the standard deviation. Then the standard deviations would be compared for the best possible match defining the roughness as the CSP value. While this method only requires a cross section and therefore requires significantly less photogrammetry, a cross section is not a very good representation of an entire surface.

The entire point cloud should be fitted with a regression plane. Then, similar to the first method, the roughness can be expressed as the standard deviation of the data set across the area. This method essentially reverses the advantages and disadvantages of the first method. While an entire point cloud is a much better representation of the surface, the photogrammetric process would be significantly higher.

The last method would include solving for the entire point cloud and comparing the point cloud of the sample with the point clouds from the standard CSP replicas. A least squares routine would be performed between the sample model and each CSP replica model. The match with the least error would define which CSP number to assign to the sample model. This represents the closest precision for determining the roughness, though the computation required would be greatest of all three alternatives.

## **Photogrammetry App for Smartphones**

In order to provide a practical measurement of roughness with photogrammetry, two requirements must be met. First, the app must run natively. In places like deep underwater, at the bottom of a massive structure, underground, or in rural locations, there is no option for processing photos on a remote server as internet connectivity is at best limited. Second, the smartphone must possess the ability to perform the needed photo processing and calculations within a reasonable timeframe.

Currently, at least one app for mobile devices has a function similar to photogrammetry. 123D Catch is a photogrammetry app published by AutoDesk. However this app provides only a smoothed 3d mesh and not the necessary point cloud for profiling the roughness of a surface. In addition, the data for the modeling is processed in the cloud. Consequently, it becomes obvious that a custom app for this application must be developed. There are two options when developing code. 1) Develop from scratch. 2) Search for an open-source code library. Due to the mathematical complexity of the photogrammetric process, option 1 should only be chosen by a team including an experienced mathematician or programmer.

Option 2 is using an open source code library for photogrammetry. Created by a community of independent programmers, OpenMVG is a library for computer-vision scientists and targeted for the Multiple View Geometry or MVG community. This code library has many perks:

- The photogrammetry process is broken into many small building blocks which can be assembled in a way that suits the specific application.
- The library of OpenMVG files covers the entire process of photogrammetry plus several other features.

- The entire library is well documented and the code files are easy to read with the help of commented expressions in the code.
- OpenMVG has no external dependencies.
- OpenMVG is released under the MPL2 (Mozilla Public License 2.0)
  - Some sub-parts are released under the MIT (Massachusetts Institute of Technology) and the BSD (Berkeley Software Distribution) license.

(Moulon, Monasse, Marlet, & Others, 2013)

At the time of this writing, the OpenMVG library is available at:

<https://github.com/openMVG/openMVG/>

Below is a list of the specific code libraries and their functions which are possibly the most relevant to the app for roughness profiling.

- The Ceres-solver library (...\\openMVG-master\\src\\third\_party\\ceres-solver) is a portable C++ library that allows for modeling and solving large complicated nonlinear least squares problems.
- The Eigen library (...\\openMVG-master\\src\\third\_party\\eigen) is a high-level C++ library of template headers for linear algebra, matrix and vector operations, and numerical solvers.
- The SIFT or Scale Invariant Feature Transform library <sup>1</sup> (...\\openMVG-master\\src\\nonFree\\sift) computes corresponding points between image pairs using SIFT keypoints and their associated descriptors.
- The Feature library (E:\\...\\openMVG-master\\src\\openMVG\\features) identifies SIFT keypoints and their associated descriptors and compiles them as a collection.
- The Geometry library (E:\\...\\openMVG-master\\src\\openMVG\\geometry) tests for camera frustum intersection (determine whether two cameras share some visual content).
- The Image library (E:\\...\\openMVG-master\\src\\openMVG\\image) imports photo files of the format ppm/pgm, jpeg, png, or tiff in either color or grayscale. The following drawing operations are also available: lines, circles, and ellipses.
- The Multiview library (E:\\...\\openMVG-master\\src\\openMVG\\multiview) accomplishes the trigonometry needed to generate a point cloud to a desired density using either Direct Linear Transform or Iterated Least Squares.
- The Numeric Library (E:\\...\\openMVG-master\\src\\openMVG\\numeric) harmonizes the Eigen library with the rest of the OpenMVG library by redefining some Eigen basis types (points, vectors, matrices).
- The Robust Estimation library (E:\\...\\openMVG-master\\src\\openMVG\\robust\_estimation) reduces the possibility of error in

---

<sup>1</sup> 1) Some features in the SIFT library are under patent. 2) The SIFT library is written in C.

the photogrammetric process by finding the best model within all the possible ones.

- The SFM library (E:\...\openMVG-master\src\openMVG\sfm) handles storage of SFM (Structure from Motion: another name for MVG) related data and includes a method to solve SFM problems.

For more information about the listed and unlisted libraries, please check the OpenMVG library package file at the link above. Note that there are sometimes several libraries for the same function.

There are many mobile device platforms available including but not limited to Apple's iOS, Google's Android OS, and Microsoft's Windows Mobile. Reclamation's IT currently only supports Apple iOS and thus is the targeted platform for this project. In the event that this app becomes a practical and successful tool, the code could be modified to support other operating systems.

The proposed app requires several features on a mobile device: 1) a built-in or modular, high-quality camera 2) processing power to handle the complex photogrammetry algorithms. Apple's latest iPhone offers both of these features in its iSight camera with A9 chip, 12 megapixel camera, f/2.2 aperture and access to most manual controls (Apple Inc., 2015).

If the OpenMVG photogrammetry libraries are to be utilized, it is necessary to code the app in C++.

Several requirements must be met to create apps for iOS. A developer's license must be purchased from Apple in order to compile and build apps. An IDE (Integrated Development Environment) is also required. There are several IDEs capable of code development for iOS.

- XCode was Apple's main SDK (Software Development Kit) until their release of Swift in 2014. XCode utilizes Objective C code to compile apps for iPad, iPhone, and iPod. Unfortunately, since XCode is designed mainly for the Objective C programming language, utilizing the OpenMVG libraries would be relatively difficult. XCode requires a Macintosh computer to run and compile code.
- Swift is Apple's newest SDK available for iOS development. Like XCode, this development environment operates on Objective C, hindering the use of the OpenMVG libraries. However, Apple has announced that Swift will be made open source later this year. (Apple Inc., 2015) This means that development on a Windows machine will be possible, eliminating the need for a Macintosh computer.
- Microsoft has incorporated several new plugins and features into their development environment Visual Studio 2015 which make it the

environment of choice. These plugins allow code to be written in C++ or C# and compiled to iOS, Android, and Windows Mobile. This feature is what makes the Visual Studio IDE the most attractive development environment. Note that all of these plugins when coding for iOS require a Macintosh computer on a Local Area Network (LAN) to act as a Compile server. Also, every app must be signed with a developer's certificate purchased from Apple.

- The Xamarin plugin for Visual Studio 2015 allows users to compile C# code to iOS (with a Macintosh build server) The business license, necessary for the Visual Studio plugin, is currently priced at \$83/month or \$999/year, making this a rather expensive option. Also, since Xamarin compiles C#, employing the OpenMVG library would be difficult. Please refer to the Xamarin website for more information at: <https://store.xamarin.com/>
- The Marmalade SDK is similar in many ways to the Xamarin plugin. One exception is that the Marmalade SDK can be compiled on a Windows machine and loaded onto an iOS device through iTunes for debugging. When the app is ready to publish, however, a Macintosh must be used to upload the app to the App Store. This plugin necessary for this particular app is currently priced at \$15/month or \$149/year, making it much more affordable than Xamarin. Additionally, the Marmalade SDK supports the C++ programming language.
- The third option seems the most viable. Visual Studio 2015 includes a built-in iOS and Android SDK without any plugin necessary. This feature uses C++. Just like Xamarin, however, a Macintosh computer must be used as a build server. This plugin is free with Visual Studio 2015 RC.

# Conclusions

The proposed development of an app for roughness profiling has a high chance for success and such an app would most likely enjoy widespread use. Two requirements still have yet to be met, however.

1. While mobile computing systems are growing quickly in performance, they still are no match for non-portable computing systems. The process of photogrammetry even for relatively undetailed point clouds strains even the best computers. Without significant optimization in the photogrammetric process or significant improvement in mobile computing, the performance of mobile computing machines would render the app unpractical and resistant to widespread adoption. Perhaps in a few years, the mobile computing will have improved enough to make such an app practical.
2. Due to the complex nature of photogrammetry and point cloud manipulation, a team building this app should include an expert in at least mathematics or software development. However, with enough time, team members with a background of mathematics and a familiarity with programming could be successful in creating the app.





# References

- Apple Inc. (2015). *iPhone 6s Tech Specs*. Retrieved December 22, 2015, from apple.com: <http://www.apple.com/iphone-6s/specs/>
- Apple Inc. (2015). *Swift*. Retrieved June 18, 2015, from developer.apple.com: <https://developer.apple.com/swift/>
- Barton, N. (2012). Shear strength criteria for rock, rock joints, rockfill and rock masses: Problems and some solutions. *Journal of Rock Mechanics and Geotechnical Engineering*, 249-261.
- Bissonnette, B., Vaysburd, A. M., & von Fay, K. (2012). *Best Practices for Preparing Concrete Surfaces Prior to Repairs and Overlays*. Denver: US Bureau of Reclamation.
- Duda, R. O., & Hart, P. E. (1971). *Use of the Hough Transform to Detect Lines And Curves In Pictures*.
- Fegyver, Z. (2014, March 15). *Understanding the Hough Transform*. Retrieved June 17, 2015, from Matlabtricks.com: <http://matlabtricks.com/post-39/understanding-the-hough-transform>
- International Concrete Repair Institute. (2013, October). *Selecting and Specifying Concrete Surface Preparation for Sealers, Coatings, Polymer Overlays, and Concrete Repair. Technical Guidelines*. Rosemont, Illinois, United States of America: International Concrete Repair Institute.
- Mingt Lu, F. D. (2011). *Applied Close-Range Photogrammetry in Construction*. Saarbrücken: Lap Lambert Academic Publishing GmbH & Co. KG.
- Moulon, P., Monasse, P., Marlet, R., & Others. (2013). *OpenMVG*. Retrieved June 17, 2015, from readthedocs.org: <https://openmvg.readthedocs.org/en/latest/>
- Tewinkel, C. G. (2011). *History of Photogrammetric Mapping In C&GS*. Retrieved June 17, 2015, from National Geodetic Survey: [http://www.ngs.noaa.gov/web/about\\_ngo/history/history\\_of\\_photogrammetric\\_mapping.pdf](http://www.ngs.noaa.gov/web/about_ngo/history/history_of_photogrammetric_mapping.pdf)



