CoastalImageLib: An open- source Python package for creating common coastal image products

Maile McCann, Dylan Anderson, Spicer Bak, Chris Sherwood, Brittany Bruder, Katherine Brodie

Coastal and Hydraulics Laboratory
U.S. Army Engineer Research and Development Center
Duck, NC
U.S. Geological Survey
Woods Hole, MA

Abstract

CoastalImageLib is a Python library that produces georectified images as well as common coastal image products intended for quantitative analysis of coastal environments. This library contains functions to georectify and merge multiple oblique camera views, produce statistical image products for a given set of images, as well create subsampled pixel instruments for use in bathymetric inversion, surface current, run-up calculations, and other quantitative analyses. Additionally, this library contains support functions to DeBayer .raw sensor data collected from the Argus tower in Duck, NC, format camera intrinsic values, and convert extrinsic values from geographical to user defined local coordinates.
Keywords: python, coastal imaging, photogrammetry

Required Metadata

C1	Current code version	1.0
C2	Permanent link to repository used for this version	$https://github.com/mailemccann$
C3	Code Ocean compute capsule	–
C4	Legal Code License	GNU General Public License, version 3
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python, OpenCV
C7	Compilation requirements, operating environments & dependencies	See CoastalImageLib User Manual
C8	If available Link to developer documentation/manual	For example:
C9	Support email for questions	mailemcc@usc.edu

Table 1: Code metadata

The permanent link to code/repository or the zip archive should include the following requirements:

README.txt and LICENSE.txt.

Source code in a src/ directory, not the root of the repository.

Tag corresponding with the version of the software that is reviewed.

Documentation in the repository in a docs/ directory, and/or READMEs, as appropriate.

1. Motivation and significance

Optical remote sensing has become an important tool for coastal scientists and engineers to expand research capabilities in the nearshore, providing low cost, accurate, and flexible methods for characterizing coastal environments. The development of regular, long term optical sampling began in the 1980s with the inception of Argus technology, which created qualitative image products that over time have evolved to become common quantitative analysis tools (3; 4). Recent advances in low-cost video hardware have further broadened the use of optical remote sensing to UAVs (1) and even free surf cameras (?).

Using this technology, algorithms quantifying nearshore dynamics from remotely-sensed products can estimate geophysical parameters such as two dimensional surface currents (5; 6; 7), bathymetry (8; 10), directional wave spectra (? ?), and shoreline position (? ? ?). However, each of these algorithms require common photogrammetry functions during data pre- processing in order to make oblique imagery into compatible input data. The necessary photogrammetry functions can include georectification of an oblique image onto world coordinates, merging multiple camera views, preparing statistical image products, and/ or creating subsampled pixel instruments from input imagery. These preprocessing steps are often outside the expertise of most coastal scientists, engineers, and oceanographers, which limits the availability of optical remote sensing algorithms to those with extensive photogrammetry backgrounds.

Steps have been taken to bridge the knowledge gap and teach photogrammetry fundamentals, such as the creation of Coastal Imaging Research Network (CIRN) and their development of the CIRN Quantitative Imaging Toolbox (9). However, this toolbox is built on Matlab, which can be cost- prohibitive and does not interface with all coastal imaging algorithms such as the CIRN shoreline mapping tool CoastSat (20). CoastalImageLib is adapted from capabilities in the CIRN Quantitative Coastal Imaging Library (9), the ARGUS Coastal Imaging System (4), and the USGS CoastCam System to ultimately provide an open- source package that interfaces with other Python based algorithms. The recent abundance of hardware and nearshore imagery holds considerable potential for coastal monitoring of both chronic and episodic hazards, motivating the need for this open-source toolbox that the entire community can use to derive inter-comparable products.

2. Software description

The CoastalImageLib package is an open- source end- to- end Python package for creating common coastal image products from oblique remotely sensed imagery. The package contains modules to georectify and merge multiple camera views, solve for unknown camera extrinsic values given ground control points, calculate statistical image products, as well as create subsampled pixel timestacks. These products can be utilized in a wide range of optical remote sensing applications, and the library itself can interface directly with Python- based algorithms.

2.1. Software Architecture

The following list shows the library structure for CoastalImageLib, expressed in terms of a hierarchical filesystem. Any classes contained in each .py file are included in italics.

CoastalImageLib/

- corefuncs.py

  class Rectifier

  class CameraData

- imageproducts.py

  class ImageStats(Rectifier)

- pixelproducts.py

  class PixelInsts(Rectifier)

- supportfuncs.py

- argusIO.py

corefuncs.py contains a series of sub- functions within the class Rectifier that implement fundamental photogrammetry calculations to merge multiple camera views and georectify oblique imagery onto a user- defined real world XYZ grid. **supportfuncs.py** contains supporting functions to format intrinsic files, convert extrinsic coordinates to and from geographical and local coordinate systems, calculate extrinsic values, and DeBayer .raw Argus files into .avi files collected from the Argus tower (4).

imageproducts.py contains functions to generate statistical image products for a given set of images and corresponding camera extrinsic and intrinsic vales. These image products are timex, brightest, variance, and darkest. **pixelproducts.py** contains functions to generate pixel instruments from a given set of images for use in bathymetric inversion, surface current, or run-up calculations.

2.2. *Software Functionalities:* ***corefuncs.py***

2.2.1. *Georectification and Merging Multiple Camera Views*

The **corefuncs.py** module contains the core functions for merging multiple camera views and georectifying oblique images onto a user- defined XYZ grid. This workflow was in part adapted from CIRN Quantitative Coastal Imaging library by Bruder et al. (9). For rectification tasks, the user would first initialize a **Rectifier** object. The user would specify x and y limits and resolution of the real- world grid in x and y directions. The value given for z should be the estimated water level at the time of data collection relative to the local vertical datum used in specifying extrinsic information. The user can also optionally specify the coordinate system being utilized, with the option of specifying the local origin for a coordinate transform. The resulting Rectifier object is valid for any rectification task using the same xyz grid.

The **Rectifier** class function **mergeRectify** is designed to merge and rectify one or more cameras at one timestamp into a single frame, as shown in 1. For multiple subsequent frames, the user can either loop through **mergeRectify** and rectify each desired frame on the same XYZ grid, or call the function **rectVideos** to merge and rectify frames from one or more cameras stored as videos on the user's drive, sampled at the same time and frame rate.

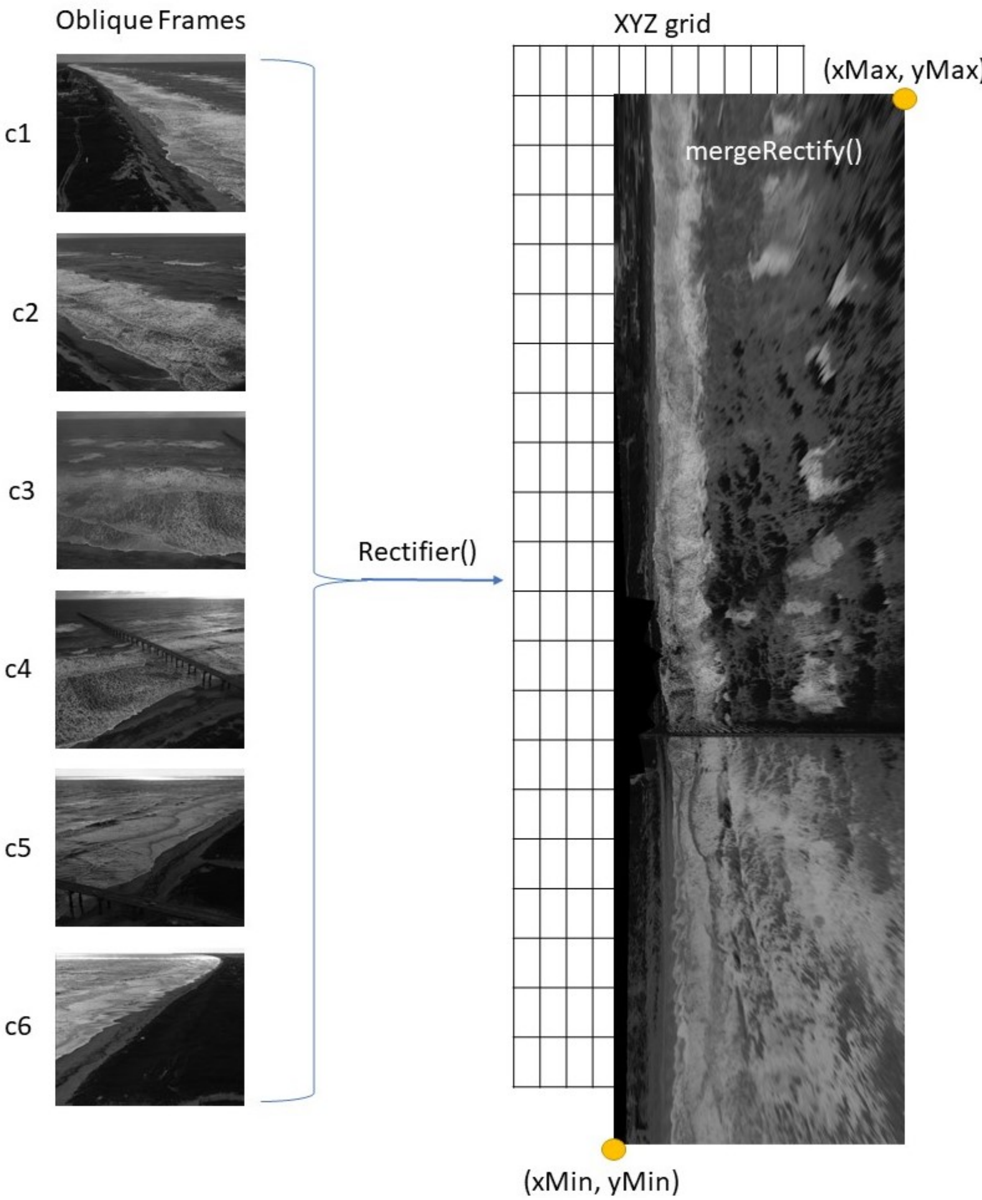


Figure 1: Diagram of the rectification process

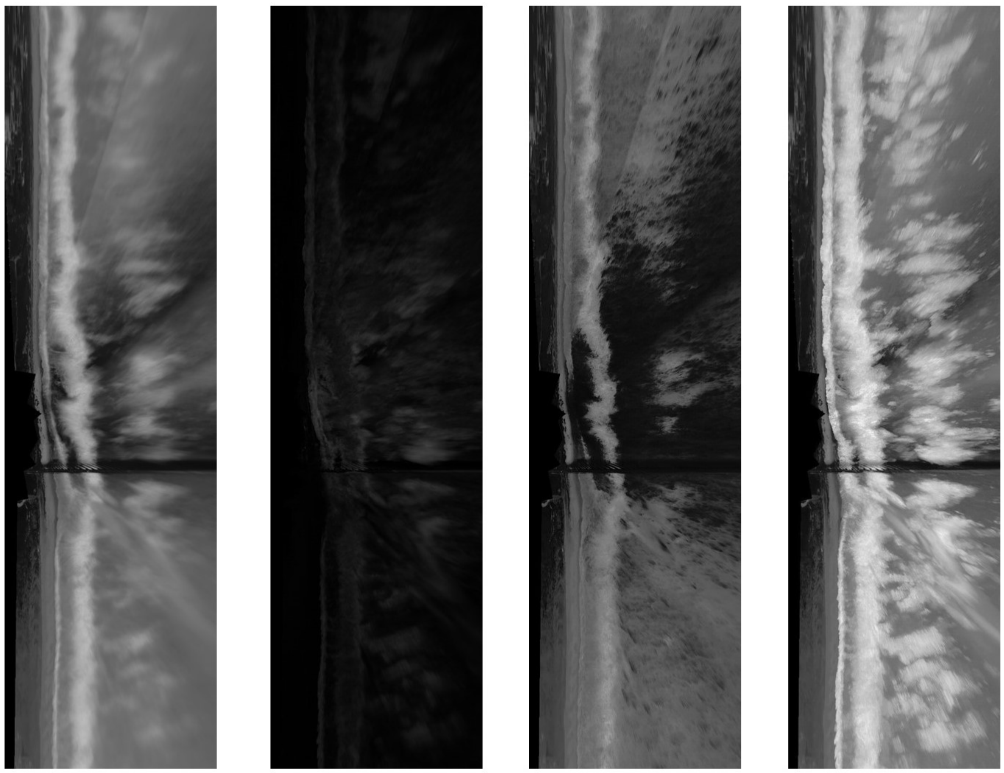
For any rectification tasks, the user is required to provide all camera intrinsic and extrinsic values unique to that device. For cameras that have not yet been calibrated and the intrinsic values are not know, the user is directed to the CalTech Camera Calibration library (12), or other relevant calibration libraries. Intrinsic values are accepted in the CIRN convention (9) or in the direct linear transform coefficient notation (13). See the WAMFlow User Manual for detailed information on calibration and intrinsic value formatting.

If oblique imagery was captured using a non-stationary camera, for example an unmanned aerial vehicle mounted camera, the user is directed to the CIRN Quantitative Coastal Imaging library for calibration and stabilization (9). Note that this library requires stationary ground control points (GCPs) and stabilization control points (SCPs). See the CIRN Quantitative Coastal Imaging library User Manual (9) for detailed information on GCPs and SCPs.

2.3. Software Functionalities: ***imageproducts.py***

The **imageproducts.py** module contains the functions to generate statistical image products for a given set of georectified images. The class ImageStats() is a child class of Rectifier(), and stores rectified images in a class object. An instance of the class ImageStats() can be initialized within the Rectifier() object if the user indicates that statistical image products are desired during the rectification process, through the mergeRectify() keyword argument stats_flag. Images can then be appended to the object directly after a rectification task. Once all images have been added, the user can call the function calcStats() to produce the statistical image products and their accompanying metadata. All image products are taken from the Argus video monitoring convention (4). The products and their descriptions are as follows:

1. Brightest: These images are the composite of all the brightest pixel intensities at each pixel location throughout the entire collection.
2. Darkest: These images are the composite of all the darkest pixel intensities at each pixel location throughout the entire collection. In regions of intermittent breaking, Darkest images have historically been used to look through the water column (**?**).
3. Timex: Time- exposure (timex) images represent the mathematical time- mean of all the frames captured over the period of sampling. Moving features, including waves and vessels, are averaged out and only mean brightness is returned. Areas of repeated wave breaking in the surf zone appear as white bands, which can help locate and determine the morphology of sand bars and rip channels (**?**).
4. Variance: Variance images are found from the variance of image intensities of all the frames captured over the period of sampling. Variance images are the brightest where they have the most variation. Variance images are primarily used to delineate the surf zone and regions of wave breaking (4).



a) Timex b) Variance c) Darkest d) Brightest

Figure 2: Examples of each of the statistical image products calculated in **imageproducts.py**

2.4. Software Functionalities: ***pixelproducts.py***

The **pixelproducts.py** module contains the functions to create subsampled pixel timestacks for use in algorithms such as bathymetric inversion, surface current estimation, or run-up calculations. Pixel timestacks show variations in pixel intensity over time. The main pixel products are included below, however additional instruments can be created from these main classes. For example, a single pixel, which may be useful for estimating wave period (?), can be generated by creating an alongshore transect of length 1.

1. Grid (also known as Bathy Array in Holman and Stanley 2007 and other publications that reference Argus image products (4)): This is a 2D array of pixels covering the entire nearshore, which can be utilized in bathymetry estimation algorithms (8). Example grid products are shown in Figures 3 and 4
2. Alongshore/ Y Transect (sometimes referred to as Vbar (4?): This product is commonly utilized in estimating longshore currents (Chicadel et. al. 2003)
3. Cross- shore/ X Transect (sometimes referred to as Runup Array (4?): Cross- shore transects can be utilized in estimating wave runup. Alongshore and cross- shore pixel instruments are depicted in Figure 5

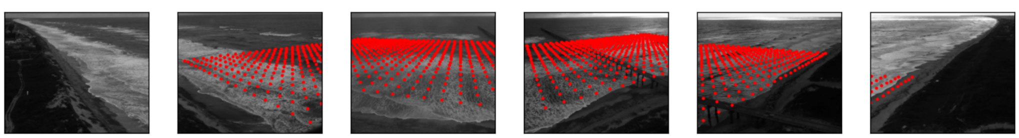


Figure 3: Pixel locations plotted on input oblique images from each of the six Argus cameras

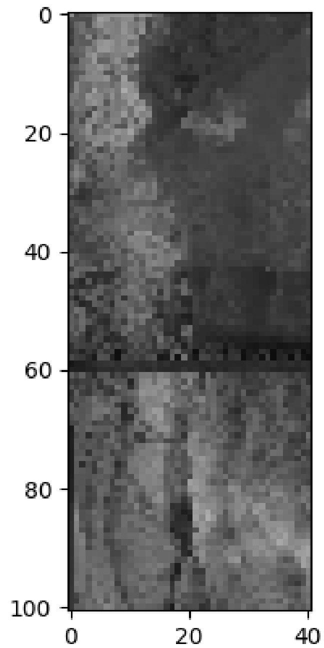


Figure 4: Output pixel grid at the pixel locations shown in Figure 3, with a resolution of 5m

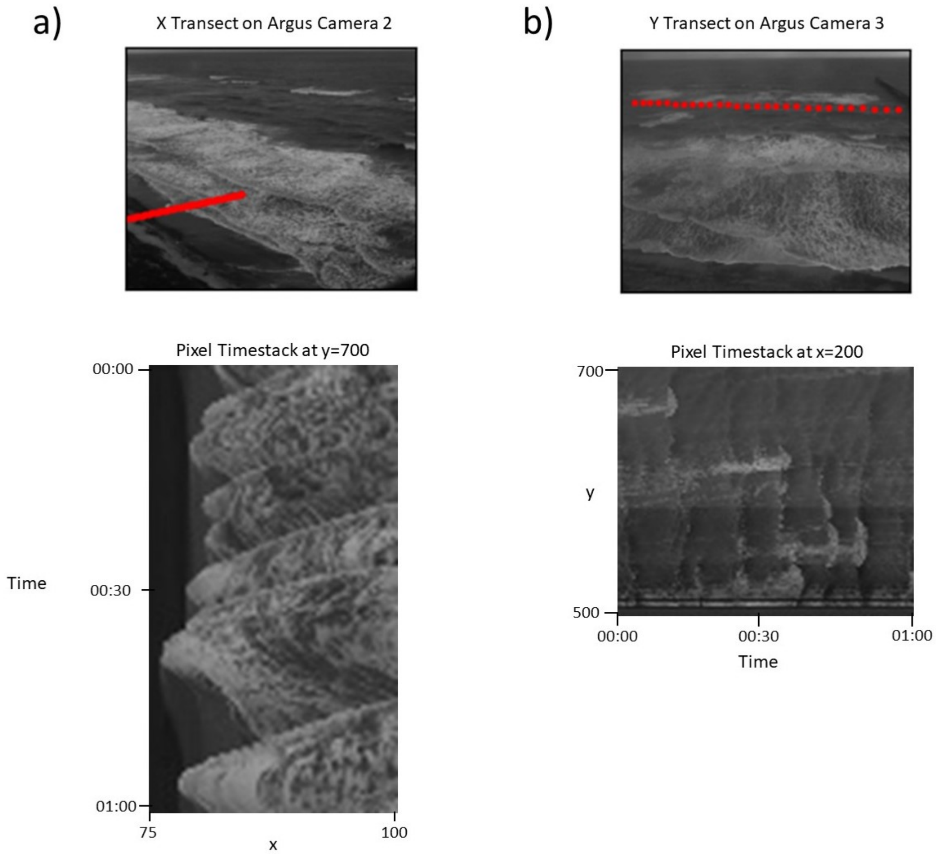


Figure 5: a) Pixel locations of an x transect shown on an oblique image taken from Argus camera 2, and the pixel timestack taken from that transect over the course of 1 minute, b) Pixel locations of a y transect shown on an oblique image taken from Argus camera 3, and the pixel timestack taken from that transect over the course of 1 minute

2.5. *Software Functionalities:* ***supportfuncs.py***

This module contains functions independent of any overarching class, which serve to assist the user in utilizing the core functions, image product functions, and pixel instrument functions. **supportfuncs.py** includes functions to format intrinsic values and extrinsic values into CIRN convention for use in the core functions, transform coordinates from geo to local, format filenames, DeBayer .raw Argus data, and other steps necessary to utilize the core functions of the **CoastalImageLib** library. See the CoastalImageLib User Manual for more detailed documentation of **supportfuncs.py**.

3. Illustrative Example: Fixed Multi Camera Demo Data

FixedMultiCamDemo data is from six fixed cameras on top a 43-m tower (known as the Argus Tower (4)) at the Army Corps of Engineers Field Research Facility in Duck, NC. Each oblique video is 17 minutes long and captured at 2 frames per second. Each video was recorded simultaneously. Six separate .avi files, corresponding to each camera collection, are included. Extrinsic and intrinsic values for each camera are provided in both direct linear transform coefficient format as well as in CIRN convention. For an interactive example script working through through the example data, users are directed to the Jupyter Notebook file contained in the **CoastalImageLib** repository entitled *multicam_demo.ipynb* and explained in the **CoastalImageLib** User Manual.

4. Impact

The recent abundance of hardware and nearshore imagery holds considerable potential for coastal monitoring of both chronic and episodic hazards, motivating the need for this open-source toolbox that the entire community can use to derive inter-comparable products. A direct impact of **CoastalImageLib**, consistent with the CIRN ideology (9), will be to reduce barriers of entry to photogrammetry that coastal engineers, geoscientists, and oceanographers may face when exploring quantitative video analysis. This library aims to provide an accessible software package that can lead to an increase in quantitative coastal studies from optical remote sensing in expanded locations, environmental conditions, and spatial or temporal scales. Unlike previous optical remote sensing packages for coastal environments, **CoastalImageLib** is open- source, as Python and the additional required packages are free and publicly available. Additionally, this library can easily interface with other Python modules such as CoastSat (20), and improve upon any packages that interface with Python compatible programs like ArcMap.

5. Conclusions

The CoastalImageLib package is an accessible open- source collection of Python modules to produce common coastal image products intended for quantitative analysis of coastal environments. This library contains functions to georectify and merge multiple oblique camera views, produce statistical image products for a given set of images, as well create subsampled pixel instruments for use in bathymetric inversion, surface current, run-up calculations, and other quantitative analyses. Additionally, this library contains support functions to DeBayer .raw sensor data collected from the Argus tower in Duck, NC, format camera intrinsic values, and convert extrinsic values from geographical to user defined local coordinates.

6. Conflict of Interest

We confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

This work is funded by USACE Coastal And Ocean Data Systems Program and (maile what project are you funded under). The authors would like to thank Kent Hathaway, and John Stanley for their help in collecting the data. [any other stuff?].

References

R. A. Holman, K. L. Brodie and N. J. Spore, "Surf Zone Characterization Using a Small Quadcopter: Technical Issues and Procedures," in IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 4, pp. 2017-2027, April 2017, doi: 10.1109/TGRS.2016.2635120.

K. T. Holland, R. A. Holman, T. C. Lippmann, J. Stanley and N. Plant, "Practical use of video imagery in nearshore oceanographic field studies," in IEEE Journal of Oceanic Engineering, vol. 22, no. 1, pp. 81-92, Jan. 1997, doi: 10.1109/48.557542.

Lippmann, T. C., Holman, R. A. "Quantification of sand bar morphology: A video technique based on wave dissipation." Journal of Geophysical Research: Oceans. 1989, 94, 0148-0227 https://doi.org/10.1029/JC094iC01p00995

Holman R.A., Stanley J. ”The history and technical capabilities of Argus”. Coastal Eng, 54 (6) (2007), pp. 477-491, 10.1016/j.coastaleng.2007.01.003

Holman, R.; Haller, M. ”Remote sensing of the nearshore.” Annu. Rev. Mar. Sci. 2013, 5, 95–113.

Haller, M.C.; Honegger, D.A.; Catalan, P.A. ”Rip Current Observations via Marine Radar”. J. Waterw. Port Coast. Ocean. Eng. 2014, 140.

Shen, C.; Huang, W.; Gill, E.W.; Carrasco, R.; Horstmann, J. An algorithm for surface current retrieval from X-band marine rader images. Remote Sens. 2015, 7, 7753–7767.

Holman, Rob, Nathaniel Plant, and Todd Holland. ”cBathy: A robust algorithm for estimating nearshore bathymetry.” Journal of Geophysical Research: Oceans 118.5 (2013): 2595-2609.

B. L. Bruder and K. L. Brodie, “CIRN Quantitative Coastal Imaging library,” SoftwareX, vol. 12, p. 100582, 2020, doi: https://doi.org/10.1016/j.softx.2020.100582.

Derian, P.; Almar, R. Wavelet-Based Optical Flow Estimation of Instant Surface Currents From Shore-Based and UAV Videos. IEEE Trans. Geosci. Remote Sens. 2017, 55, 5790–5797.

Anderson, D.; Bak, A.S.; Brodie, K.L.; Cohn, N.; Holman, R.A.; Stanley, J. Quantifying Optically Derived Two-Dimensional Wave-Averaged Currents in the Surf Zone. Remote Sens. 2021, 13, 690. https://doi.org/10.3390/rs13040690

Bouguet J.-Y. Camera calibration library for Matlab, vol. 1080 (2008). http://www.vision.caltech.edu/bouguetj/calib_doc

Abdel-Aziz, Y.I.; Karara, H.M. ”Direct Linear Transformation from Comparator Coordinates into Object Space Coordinates in Close-Range Photogrammetry”. Photogrammetric Engineering Remote Sensing. 81 (2): 103–107. doi:10.14358/pers.81.2.103

Farneback, G. Two-Frame Motion Estimation Based on Polynomial Expansion. In Scandinavia Conference on Image Analysis; Springer: Berlin/Heidelberg, Germany, 2003; pp. 363–370.

Horn, B.K.; Schunck, B.G. Determining optical flow. Tech. Appl. Image Underst. 1981, 281, 319–331.

Bradski, G. The OpenCV Library. Dobb’s J. Softw. Tools 2000, 3, 1–81.

Plant, N.; Holman, R.; Freilich, M.; Birkemeier, W. A simple model for interannual sandbar behavior. J. Geophys. Res. 1999, 104, 15755–15776.

Long, C.E.; Oltman-Shay, J.M. Directional Characteristics of Waves in Shallow Water; Technical Report Coastal Engineering Research Center; 91-1-1; United States Army Corps of Engineers: Vicksburg, MS, USA, 1991; pp. 1–130.

Rodríguez-Padilla, I.; Castelle, B.; Marieu, V.; Bonneton, P.; Mouragues, A.; Martins, K.; Morichon, D. Wave-Filtered Surf Zone Circulation under High-Energy Waves Derived from Video-Based Optical Systems. Remote Sens. 2021, 13, 1874. https://doi.org/10.3390/rs13101874

Kilian Vos, Kristen D. Splinter, Mitchell D. Harley, Joshua A. Simmons, Ian L. Turner, CoastSat: A Google Earth Engine-enabled Python toolkit to extract shorelines from publicly available satellite imagery, Environmental Modelling Software, Volume 122, 2019, 104528, ISSN 1364-8152. https://doi.org/10.1016/j.envsoft.2019.104528.

conlin Matthew P. Conlin, Peter N. Adams, Benjamin Wilkinson, Gregory Dusek, Margaret L. Palmsten, Jenna A. Brown, SurfRCaT: A tool for remote calibration of pre-existing coastal cameras to enable their use as quantitative coastal monitoring tools, SoftwareX, Volume 12, 2020, 100584, ISSN 2352-7110. https://doi.org/10.1016/j.softx.2020.100584.