The Journal of Open Source Software

CoastalLens: A MATLAB UAV Video Stabilization & Rectification Framework

Athina M. Z. Lange[1], **Holger Lange**[2], **Julia W. Fiedler**[3], **and Brittany L. Bruder**[4]

1 Scripps Institution of Oceanography, University of California, San Diego, USA **2** AI Werkstatt, USA **3** University of Hawai'i Sea Level Center, USA **4** Coastal and Hydraulics Laboratory, US Army Engineer Research and Development Center, USA

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 11 March 2024
Published: unpublished

Statement of need

Uncrewed aerial vehicles (UAVs) are an important tool for coastal monitoring with their relatively low-cost and rapid deployment capabilities. To generate scientific-grade image products, to use for wave runup observations, for bathymetry inversions, or tracking surfzone currents, the hovering UAV images/videos must be stabilized and rectified into world coordinates. Due to the limited stationary region of coastal images suitable for control points, the processing of UAV-obtained videos can be time-consuming and resource-intensive. The CIRN Qualitative Coastal Imagining Toolbox(Bruder & Brodie, 2020) provided a first-of-its-kind open-sourced code for rectifying these coastal UAV videos. Limitations of the toolbox, however, prompted the development of CoastalLens with an efficient data input procedure, providing capabilities to obtain drone position (extrinsics) from LiDAR surveys, and using a feature detection and matching algorithm to stabilize the video prior to rectification. This framework reduces the amount of human oversight, now only required during the data input processes. Removing the dependency on threshold stability control points provides more stable results and can also result in less time in the field. We hope this framework will allow for more efficient processing of the ever-increasing coastal UAV datasets.

Summary

CoastalLens is set up as 4 scripts (with an optional 5th script) run sequentially from a main entry point script (`UAV_rectification.m`). This allows users to execute parts or all of the full framework depending on their workflow. The first script, `input_day_flight_data.m`, prompts user input and returns all the user-specified required input data organized in stuctures to be used by the subsequent scripts. The user is required to input data for each day and flight to process. Required user inputs are the video timezone, camera intrinsics, the Products (types of images) to be generated (e.g. Grid/Rectified Image, xTransect or yTransect), and the ground control points to determine the camera world position (via GPS points or pointcloud) ((Hartley & Zisserman, 2004), (Xiao-Shan Gao et al., 2003), (Conlin et al., 2020)). Users can load in pre-set values for the relevant day-specific information from a configuration file. This is useful if similar UAV missions are flown repeatedly at the same location. The second script, `extract_images_from_UAV.m`, extracts images from the video files at the specified frame rates. This is done via a system command to the ffmpeg command line tool. The third script, `stabilize_video.m`, accounts for the UAV movement and returns the 2D projective transformation of the image to improve image stabilization through flight. We take an approach similar to constructing a panorama image. Static features (e.g. corners, windows, lines on the ground) are found in every frame. In subsequent frames, these features are matched and the movement/change in location of these features between the frames is used to estimate the change in position of frame 2 versus frame 1. This is used to warp the image into fitting into the full 'panorama' image. This approach allows for good estimates to be obtained even in cases where the UAV drift substantially ((Brown & Lowe, 2007), (Torr & Zisserman, 2000)). From these stabilized images we can produce standard ARGUS products, like time-averaged images, brightest and darkest image (Holman & Stanley, 2007). In the final main script, `get_products.m`, the image coordinates corresponding to the world coordinates of the previously defined products are determined. These image coordinates are used to extract the pixels from each frame. `save_products.m` is an optional code to save the resulting rectified images as png's.

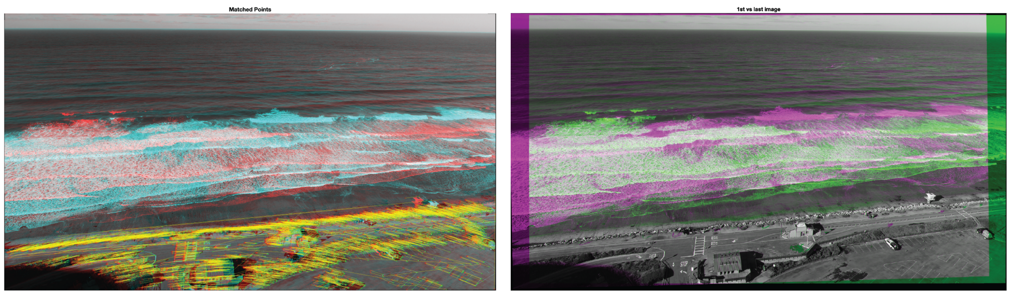

Figure 1: Figure 1: Example of matched features and the 2D projective transformation of the image. (left) Image 1 (red) and Image 2 (blue) are taken 1 minute apart (extreme case) and features have been detected and matched between the two frames. Note the shift in the lifeguard tower on the right, or the pedestrian crosswalk in the middle of the image. (right) Image 1 (green) and Image 2 (purple) shown after they have been warped into the 'panorama' image. Note large grey region at the bottom and the horizon at the top of the image where the two frames match and the stabilization has succeeded.

Figure 2: Figure 2: Example of 17 minutes of video stitched together. Extreme drift in the UAV can be seen, but horizon at the top and road at the bottom of the image remain stable.

Acknowledgements

This study was funded by the U.S. Army Corps of Engineers (W912HZ1920020) and the California Department of Parks and Recreation (C19E0026). Thank you to Rob Grenzeback, Julia Fiedler, Alex Simpson and Holden Leslie-Bole for help collecting this data and testing the framework.

References

Brown, M., & Lowe, D. G. (2007). Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, *74*(1), 59–73. https://doi.org/10.1007/s11263-006-0002-3

Bruder, B. L., & Brodie, K. L. (2020). CIRN quantitative coastal imaging toolbox. *SoftwareX*, *12*, 100582. https://doi.org/10.1016/j.softx.2020.100582

Conlin, M. P., Adams, P. N., Benjamin, W., Dusek, Gregory, Palmsten, M. L., & Brown, J. A. (2020). SurfRCaT: A tool for remote calibration of pre-existing coastal cameras to enable their use as quantitative coastal monitoring tools. *SoftwareX*, *12*. https://doi.org/10.1016/j.softx.2020.100584

Hartley, R., & Zisserman, A. (2004). *Multiple view geometry in computer vision* (Second edition). Cambridge University Press. ISBN: 978-0-511-18711-7

Holman, R. A., & Stanley, J. (2007). The history and technical capabilities of argus. *Coastal Engineering*, *54*(6), 477–491. https://doi.org/10.1016/j.coastaleng.2007.01.003

Torr, P. H. S., & Zisserman, A. (2000). MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, *78*(1), 138–156. https://doi.org/10.1006/cviu.1999.0832

Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, & Hang-Fei Cheng. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(8), 930–943. https://doi.org/10.1109/TPAMI.2003.1217599