



Vera C. Rubin Observatory
Data Management

Streak Masking in DM Image Processing

Clare Saunders

DMTN-197

<https://doi.org/10.71929/rubin/2586496>

Latest Revision: 2021-07-15



Abstract

Streaks caused by satellites are a persistent problem in optical images, and their occurrence will likely increase in the coming years. Many Hyper Suprime-Cam images are already affected by streaks, and the same is expected for Rubin Observatory data. In the LSST Data Release Production (DRP), most streaks and other artifacts are already detected and masked by the algorithm `CompareWarpAssembleCoadd`. However, some streaks are not caught at this stage and can contaminate the final coadds and detection catalogs. To find and mask these, we adopt a morphologically-based method for detecting streaks, which uses the Kernel-Based Hough Transform to detect straight lines. Once the lines are detected, the streak profile is fit and the affected portion of the image is masked out. This algorithm, `maskStreaks`, is included in the `meas_algorithms` package and is used in `CompareWarpAssembleCoadd` to remove streaks from coadds. Similar implementation in the Alert Production Pipeline is possible but has not been implemented.

Change Record

Version	Date	Description	Owner name
1	2021-07-15	Version 1.0	Clare Saunders
1	2021-12-09	Version 1.1	Clare Saunders
1	2024-03-13	Version 1.2, Update of task repository	Clare Saunders

Document source location: <https://github.com/lsst-dm/dmtn-197>

Contents

1 Introduction	1
2 Method	3
2.1 The Streak Finding Algorithm	3
2.2 Fitting the Streak Mask	3
3 Implementation in the DRP Pipeline	6
4 References	8
5 Acronyms	8

Streak Masking in DM Image Processing

1 Introduction

Streaks in astrophysical observations due to satellites are an increasing problem in image processing. A significant fraction of Hyper Suprime-Cam (HSC) observations are contaminated by streaks, and while the exposure times for Rubin Observatory images will be much shorter, streaks are expected to appear in a significant fraction of Rubin images as well, particularly in twilight. `CompareWarpAssembleCoadd` [DMTN-080], which performs image coaddition, largely succeeds in removing satellite streaks by identifying transient artifacts through temporal filtering. The temporal filtering removes objects that only appear in a small percentage of images. However, some streaks still persist into the cleaned final coadditions, generally because they overlap with objects such as bright stars. In order to remove any streaks that remain despite temporal filtering, morphological filtering can be used to identify streaks based on their shape. Additionally, while not currently implemented, morphological identification of streaks will be particularly necessary for the Prompt Processing of Rubin data, since temporal filtering is not possible for single images or sets of two snaps.

Streaks in astronomical images caused by satellites are easily identified by the human eye, but the vast quantity of images that will be generated by Rubin obviously requires an automated approach. The Hough Transform (Hough, 1962; Duda & Hart, 1972) is a well-established method for identifying lines in images, but it requires scanning through all possible lines in an image, given a certain discretization, which is computationally expensive. Also, it can detect spurious lines from non-contiguous pixels, such as three approximately collinear galaxies.

Fernandes & Oliveira (2008) provides an efficient alternative to the conventional Hough Transform with the Kernel-Based Hough Transform (KHT). KHT finds clusters of non-zero points and keeps those clusters that meet the given requirements for length and straightness. An efficient voting scheme is then used, in which votes are cast using a Gaussian kernel that models the uncertainty of the best-fitting line to the cluster. Figure 1 shows a comparison of the results from a conventional Hough Transform and from the Kernel-Based Hough Transform, and Figure 2 shows the results on a sample image.

Once streaks are identified, their approximate profile can be fit so that the portion of the image contaminated above a given level can be masked out. The brightness at which the wings of

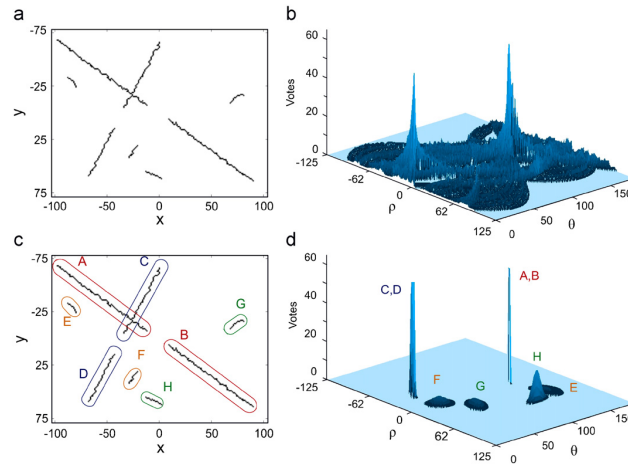


Figure 1: Differences in the conventional versus Kernel-based Hough Transforms: (a) shows simple image with approximately straight line segments; (b) shows the voting map in the two dimensions determined by the angle and radius from the image center of the candidate line; (c) shows the clusters of pixels detected in KHT; (d) shows the voting map produced with the KHT technique, where the height of the A-B and C-D peaks have been truncated to improve the visualization. Reprinted from Fernandes & Oliveira (2008).

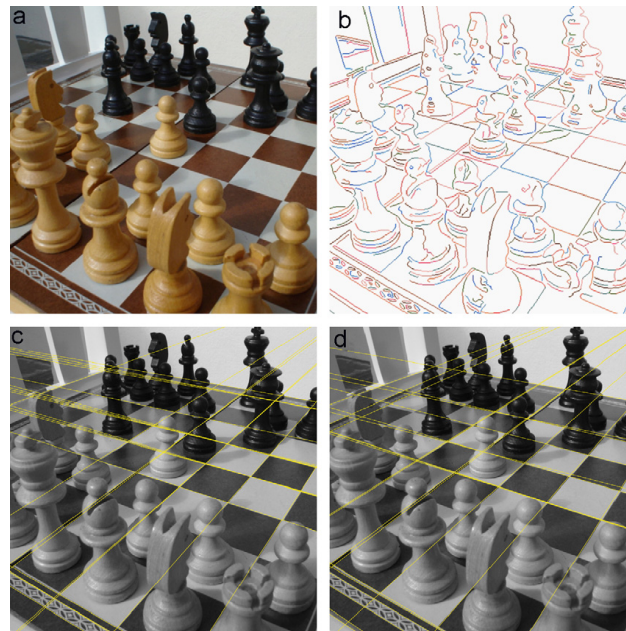


Figure 2: Line detection using the Kernel-based Hough Transform and conventional Hough Transform. (a) shows the input image; (b) shows the results of using a Canny edge detector; (c) shows the results produced by a conventional Hough Transform, with too-many lines detected in some places, while other lines are missed; (d) shows the results of the Kernel-based Hough Transform. Reprinted from Fernandes & Oliveira (2008)

the streak are masked out must be balanced with the need to preserve as much of the image as possible.

The procedure used to identify streaks is described in Section 2.1. The method to refine the fit to the identified streaks and to mask them out is described in Section 2.2. The method's current implementation in the Rubin pipeline is described in Section 3.

2 Method

2.1 The Streak Finding Algorithm

In practice, some preprocessing of astronomical images helps KHT work smoothly. Figure 3 demonstrates these steps on a sample image, where panel (a) shows the original image. The map of “detected” pixels is used to produce a binary image (panel (b)), which removes background noise and significantly speeds up runtime by removing spurious clusters of points; then, a Canny filter is applied to detect the edges of objects in the image (panel (c)), which turns a streak into two approximately straight, parallel lines. At this point, KHT can be run on the processed image and will result in a small set of potential lines in the image (blue lines in panel (d)).

KHT parameterizes lines with their angle and radius from the center of the image, and both KHT and the original Hough Transform discretize the angle-radius space in order to restrict possible lines to a finite number. Thus, further fitting is required to reconcile multiple lines fit to the same streak and to refine the exact position of the streak, for which the discretization may not be sufficiently accurate.

2.2 Fitting the Streak Mask

For a given streak in an image, KHT will typically result in a small set of likely lines, generally with at least one detection for each edge of the detected streak. From this set of likely lines, we want to produce a mask covering the portion of the image affected by the streak above a certain level. To do this, the results from KHT are first separated into clusters using the K-means clustering algorithm implemented in `sklearn.cluster.KMeans` (Pedregosa et al., 2011). The best guess for the center of the streak is then estimated to be the average of the

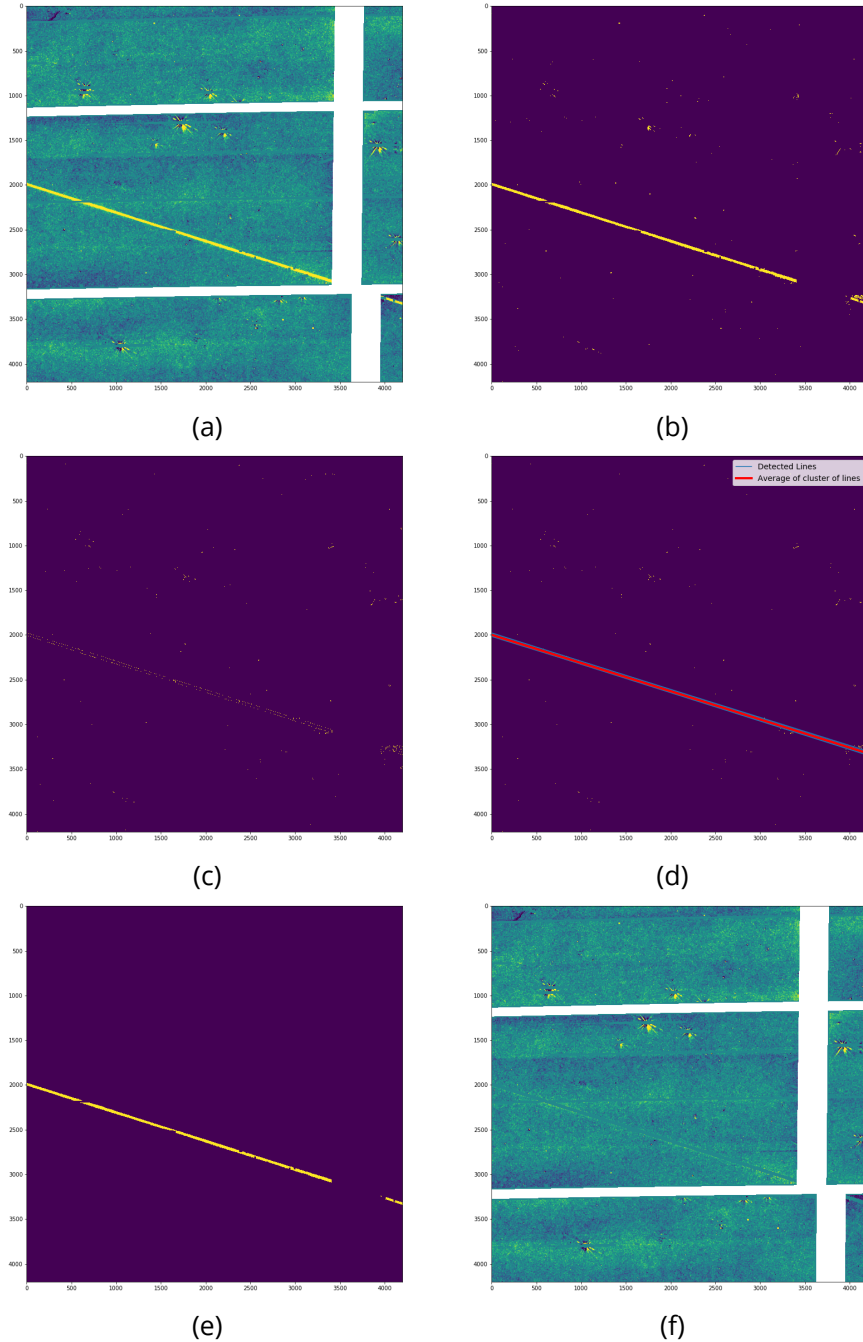


Figure 3: Demonstration of the streak masking: (a) shows the original image, which is a warped difference image created from subtracting a clipped deepCoadd from a deepCoadd_directWarp; (b) shows the mask of detected pixels in the image; (c) shows the result of using a Canny edge detector; (d) shows the lines detected by KHT in blue, with the line corresponding to the average position of the KHT lines is in red; (e) shows the mask fit to the detected streak; (f) shows the final image with the streak masked out.

parameters for the likely lines in the cluster, shown as a red line in panel (d) of Figure 3.

Given this best guess for the position of the streak, we then return to the original image to both refine the fit of the position and to fit its approximate profile. The streak is modeled as a line convolved along the perpendicular with a Moffat point spread function (PSF). The angle and radius of the line, plus the width of the Moffat profile, are all fit parameters, while the flux of the streak is determined by finding the best fit to the data given the other parameters of the model. The best fit of the model is determined by using the Newton-Raphson minimization method. Figure 4 shows the profile of a streak and the model fit to it. Given this model, the pixels above a given threshold in mJansky are used to make a mask (panel (e) of Figure 3).

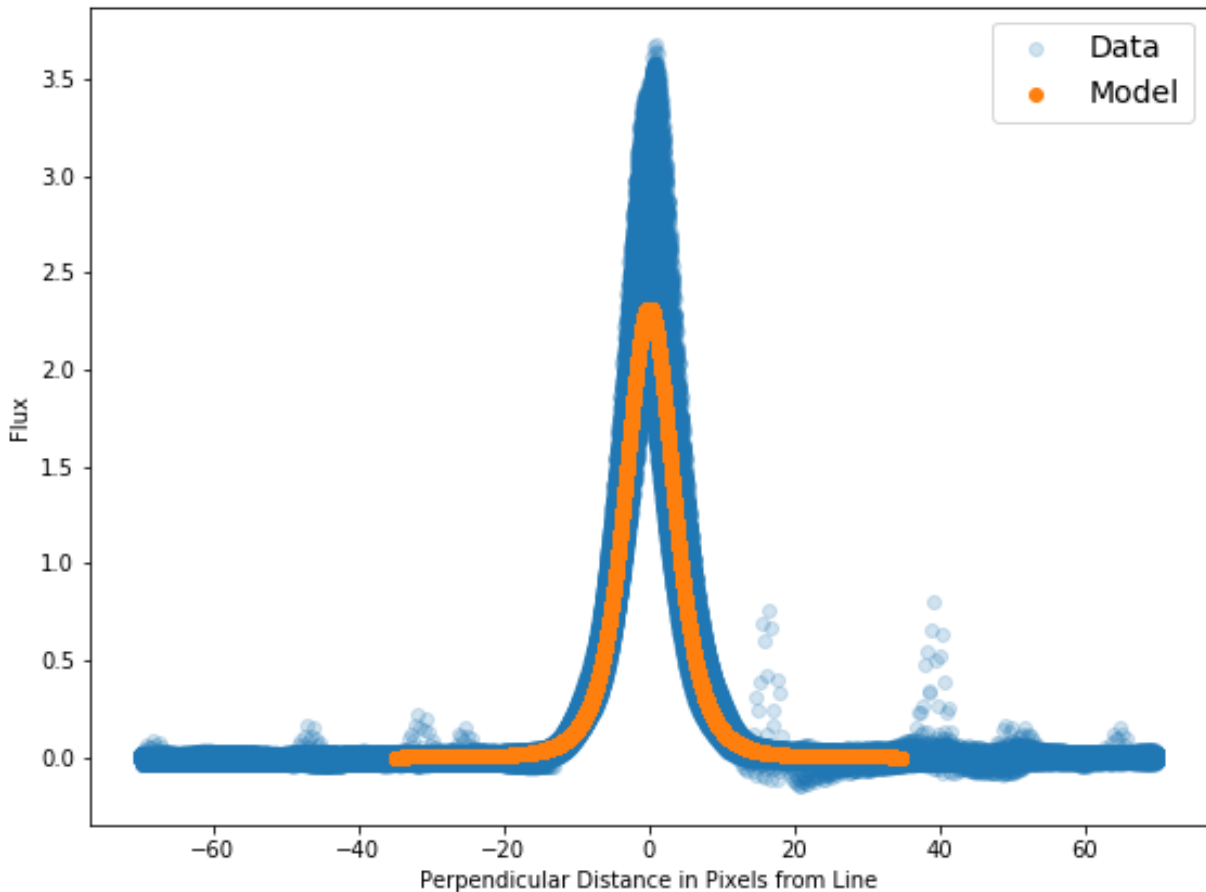


Figure 4: Image data at the location of a streak, with the model fit to the streak profile.

The final mask determined by this procedure is the intersection of the original map of “detected” pixels and the masks of any streaks detected in the image. The masked image is

shown in panel (f) of Figure 3.

In the example shown, there is still a slight brightness around the masked area. This is currently tolerated because it is at or below the level of instrumental effects, such as the differences between amplifiers visible in the example image. However, the brightness threshold used to set the edge of the streak mask can be adjusted as needed to meet survey needs.

3 Implementation in the DRP Pipeline

Streak masking is currently only attempted in the Data Release Production, where it is used in the construction of the coadded images. In `CompareWarpAssembleCoaddTask`, a naive model of the static sky is subtracted from each of the PSF-matched warps. Artifacts are then found by looking at detections in the difference image. Since the naive model of the sky is computed as a clipped mean, streaks, appearing in only one image, do not appear in it, and thus show up in the difference images. The algorithm `maskStreaks` is run on these warped difference images and any detected pixels are marked by a mask plane with the default name "STREAK". For temporal artifact detection, apparent artifacts are filtered out if they appear in less than a certain fraction of difference images. However, artifacts that are determined to be streaks are filtered out without regard to how many images they appear in. This helps to ensure that streaks that overlap with other static objects, such as bright stars, are still removed. In the coadded image, streaks are included with other artifacts in the "CLIPPED" mask plane. The streak-masking step adds about 4 seconds per input image when run on one node through the batch processing service on the NCSA machine `1sst-dev1`, compared to about 35 seconds for the rest of the `assembleCoadd` task.

An example of the results of the image coaddition in the Gen3 pipeline with and without using streak masking is shown in Figure 5. A small portion of a streak appears to still be visible after the streak masking procedure. This may have been because the streak crossed the corner of the difference image, making it difficult to detect as a linear feature. Neighbouring images could be used to help detect streaks in this case, but that will be saved for future work.

Future work may also add this streak masking to the prompt processing pipeline for Rubin, but this has not yet been tested or implemented. This should hypothetically be simple, but may require taking steps to speed up the runtime.

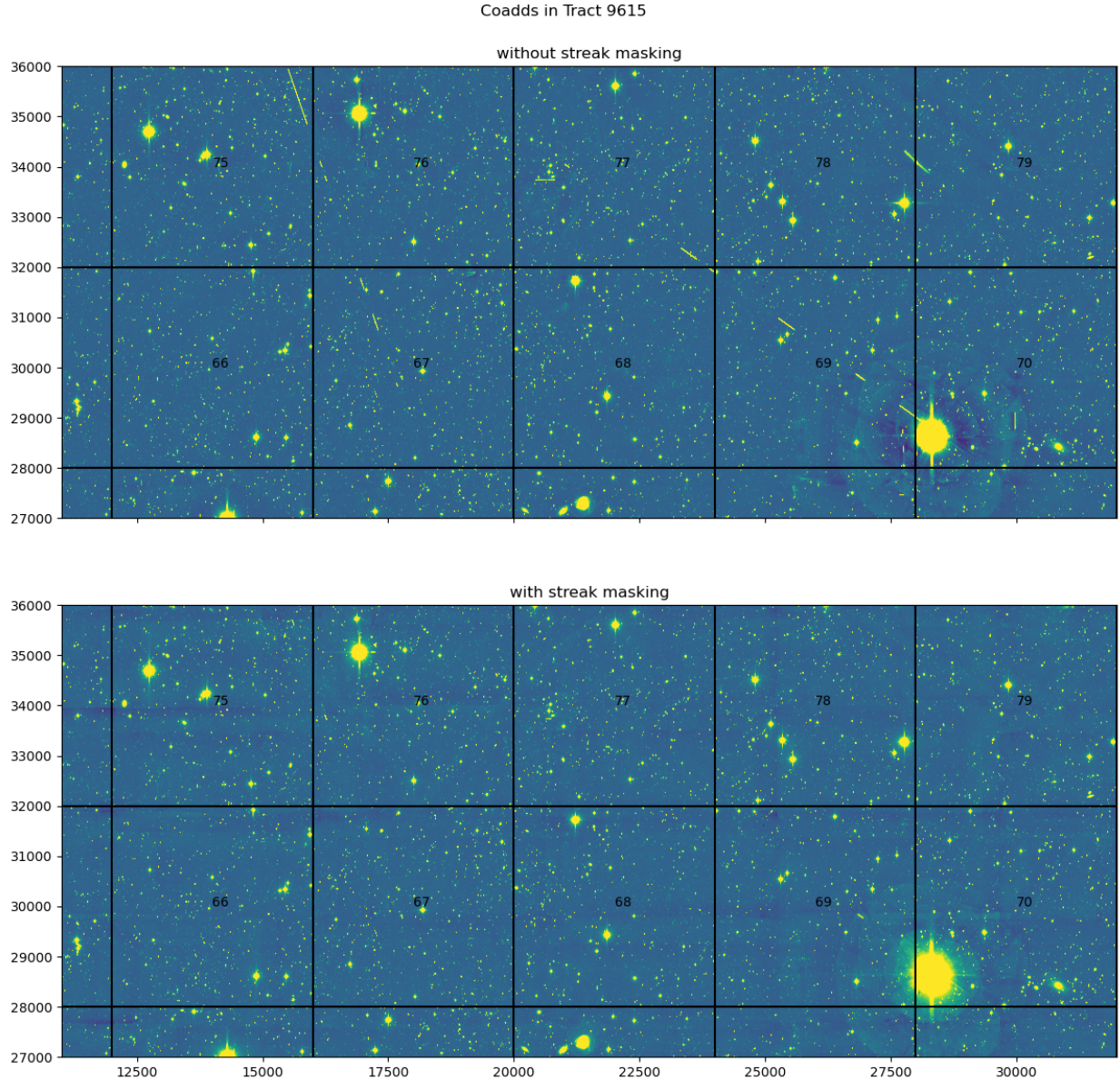


Figure 5: The top panel shows a mosaic of several HSC-Y band coadds in Tract 9615, where patches 67, 69, 75, 77, 78, and 79 are affected by satellite trails, or possibly optical ghosts in the case of the streak on the border of patches 78 and 79. The lower panel shows the result of processing the data after streak masking was added to the pipeline. The red box in the lower panel shows a small section of a streak that was not successfully masked.

4 References

- [DMTN-080], AlSayyad, Y., 2018, *Coaddition Artifact Rejection and CompareWarp*, Data Management Technical Note DMTN-080, NSF-DOE Vera C. Rubin Observatory, URL <https://dmtn-080.lsst.io/>, doi:10.71929/rubin/2583441
- Duda, R.O., Hart, P.E., 1972, Commun. ACM, 15, 11, URL <https://doi.org/10.1145/361237.361242>, doi:10.1145/361237.361242
- Fernandes, L.A.F., Oliveira, M.M., 2008, Pattern Recognition, 41, 299, URL <http://www.ic.uff.br/~laffernandes/projects/kht>, doi:10.1016/j.patcog.2007.04.003, ADS Link
- Hough, P.V., 1962, URL <https://www.osti.gov/biblio/4746348>, US Patent
- Pedregosa, F., Varoquaux, G., Gramfort, A., et al., 2011, Journal of Machine Learning Research, 12, 2825, URL <http://jmlr.org/papers/v12/pedregosa11a.html>

5 Acronyms

Acronym	Description
B	Byte (8 bit)
DM	Data Management
DMTN	DM Technical Note
DRP	Data Release Processing
HSC	Hyper Suprime-Cam
KHT	Kernel Hough Transform
LSST	Legacy Survey of Space and Time (formerly Large Synoptic Survey Telescope)
NCSA	National Center for Supercomputing Applications
PSF	Point Spread Function