Final Documentation Draft

Jeren Suzuki

Last Edited June 13, 2013

Contents

1 Introduction

Starting with an image of at most three suns on a fiducial grid, we find the centers of the suns and their relative position to the fiducials (which provide a physical distance calibration). The program deems suns too close to the edge or suns partially cut off as unfit for centering.

2 Introduction cont.

1. Load Image
2. Read parameters from pblock.txt
3. Sort image and cut off top .1% of pixels (top 1% was actually too much)
4. Smooth, take deriv, smooth again, take deriv again of sorted array, find peaks that correspond to difference solar regions and their thresholds
5. Mask image above thresholds to find centers of every shape, regardless of partial or not
6. If center of shape is within a certain distance to edge of image, mark as partial and cease further analysis
7. Crop remaining whole suns
8. Extract 5 strips centered around cropped solar center for both X and Y direction
9. Extract a pair of limb strips for each long strip
10. Applt linear fit to limb profile
11. Mark position where fit crosses threshold
12. Use new threshold-crossing position to calculate chord lengths
13. Average midpoints of chords to find limb-fitted centers
14. Analyze the cropped image for fiducials
15. Using the fiducial positions, we compare the solar positions we calculated to a position defined by the physical setup.

This is the form of the fiducial structure containing the positions and sub-pixel positions of fiducials for each solar region.

```
>> help,*(bbb[0])
** Structure <260a348>, 2 tags, length=180, data length=178, refs=1:
   REG             INT              1
   FIDARR          STRUCT  -> FIDPOS Array[11]
>> help,(*(bbb[0])).fidarr,/str
** Structure FIDPOS, 4 tags, length=16, data length=16:
   X               FLOAT          50.0000
   Y               FLOAT          132.000
   SUBX            FLOAT          50.8438
   SUBY            FLOAT          133.291
```

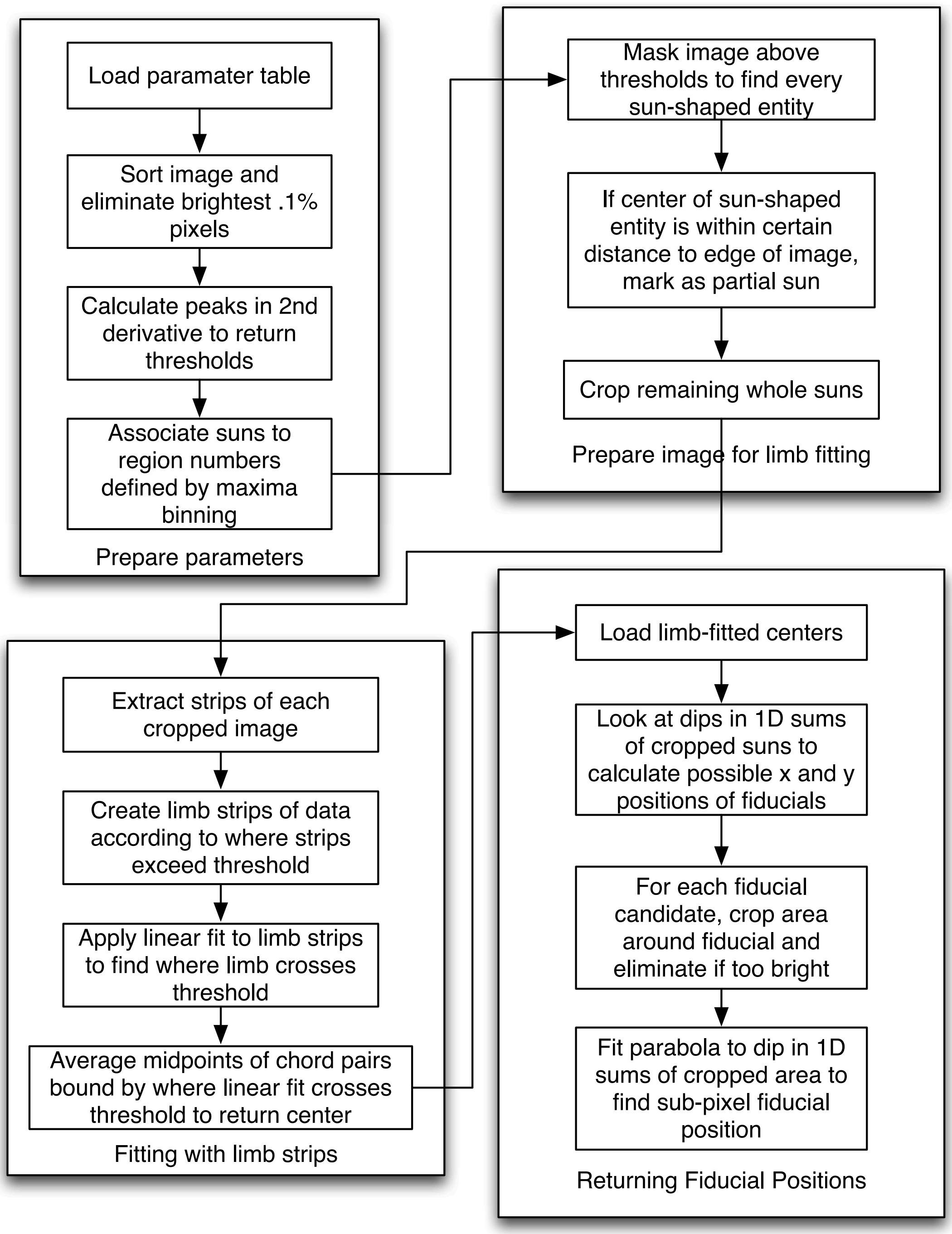


Figure 1

3 Setting Up Parameters

Before we analyze the solar image, we load a parameter table and assign values.

```
scan_width 10            ; Distance to next chord when picking chords to limb-fit
sundiam 70               ; Approx Solar diameter, deprecated
nstrips 5                ; Number of pairs of solar chords to limb-fit per direction
ministrip_length 4       ; Length of limb profile to linear fit
crop_box 120             ; Half-width of box used to find fiducials in
elim_perc 1              ; Percentage of highest pixels to eliminate when finding threshold
```

Table 1. Final data structure of solar region

Name	Type	Value	Notes
XPOS	FLOAT	210.522	Rough calculation using a simple masking method
YPOS	FLOAT	166.702	”
REG	INT	1	Region ID #: 1 is 100%, 2 is 50%, 3 is 25%
THRESH	FLOAT	106.000	Threshold calculated from sorting array and taking derivatives. Used in both finding rough X-Y center as well as the threshold for limb-fitting.
PARTIAL	FLOAT	0.	Flag that determines if the solar region is cut off on the edge or not. 0 means that it is not cut off
XSTRIPS	STRUCTURE	-> WHOLEXSTRIPS Array[5]	Strucutre containing the strips of whole solar data bound by a cropped region chosen by XPOS and YPOS
YSTRIPS	STRUCTURE	-> WHOLEYSTRIPS Array[5]	”
LIMBXSTRIPS	STRUCTURE	-> LIMBXSTRIPS Array[5]	LIMBSTRIPS contains a pair of arrays, ENDPOINTS and STARTPOINTS that mark the limbs of each strip of data from X/YSTRIPS
LIMBYSTRIPS	STRUCTURE	-> LIMBYSTRIPS Array[5]	”
LIMBXPOS	FLOAT	210.710	Center calculated from LIMBXSTRIPS
LIMBYPOS	FLOAT	167.172	”
NPIX	FLOAT	26680.0	Number of pixels above threshold

```
n_smooth 900              ; Elements to smooth by when finding threshold
soldiskr  60              ; Deprecated
border_pad 50             ; If solar center is within this value of border, marked as a partial sun
triangle_size  .25        ; Percentage of image height to use for triangle sides for making clipped-bottom-corner
    mask
fid_smooth_thresh -150    ; Threshold to determine row/column positions of fiducials
onedsumthresh 80          ; Once looking at fiducial candidates, look at 1D sum of smaller fiducial crop and
    threshold difference of smoothed array - original array by this
disk_brightness 15        ; Arbitrary pixel brightness to eliminate bright fiducial candidates which are on the
    solar disk but are not on a fiducial
fid_crop_box 15           ; Half-width of box used to analyze fiducials
fid_smooth_candidates 15  ; Smoothing paramater for 1D sums of fiducial candidates
```

In Figure 2 we dynamically set thresholds for our image. Using these thresholds, we mask our image and identify solar regions by their brightness. We can't simply order them by brightness because we may have a 100% and 50% brightness sun in the same image and in order to determine the difference between a 100%/50% sun combo and a 50%/25% sun combo we must look at the actual threshold values.

3.1 Possibility for Error

Herein lies the problem of determining solar regions based on brightness and threshold values. We don't have a clear way of determining whether or not a sun is being obscured pre-imaging and in result, becoming dimmer. If this happens, an image with a 100% and 50% sun may appear to be a 50% and 25% sun. We could do something with the time of day and make sure if the sun is obscured by anything we can predict, we account for it in our code.

4 Prepare Image for Limb Fitting

After we set our threshold and parameters, we iteratively find and crop suns in our image. To determine if the sun is partially cut off, we create a zone around the edge of our (usable) part of the image that if the center is within, it is counted as a fiducial. I emphasize *usable* because the image has two triangles cut out of the bottom corners, reducing the usable part of the image. Figure 3 gives and example of what our mask might look like. The gradient is unimportant, it's purpose is only to emphasize the shape of the mask.

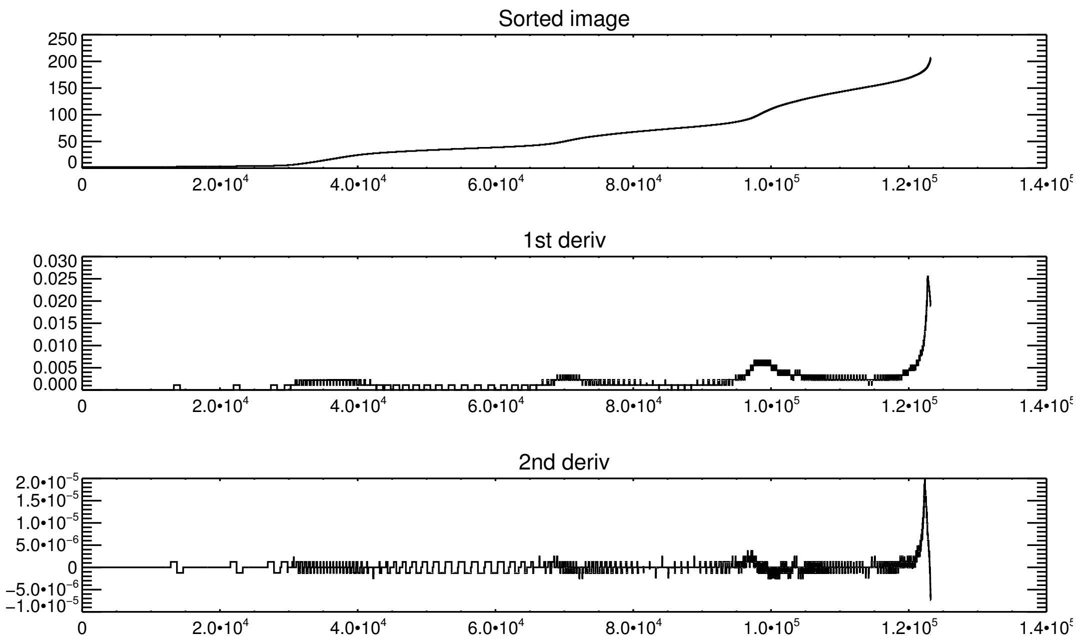
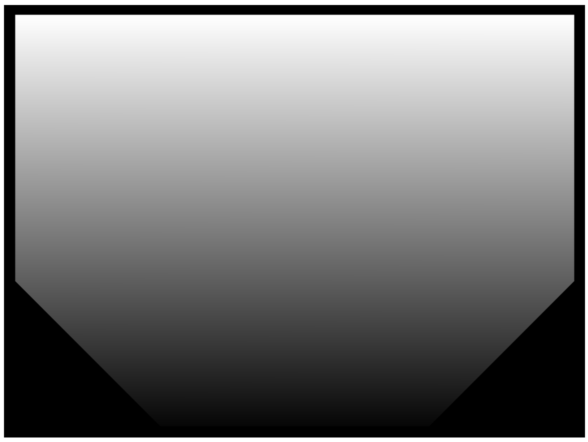


Figure 2 We look at the second deriv because it consistently quantifies the thresholds to identify each solar region by. Unfortunately (or for better), we repeat this process per image we analyze. In terms of total time spent from starting the program to returning limb-fitted centers and fiducials, setting thresholds takes up about 25%.

(a) What our mask should look like - side of black triangle is 1/4 of image width



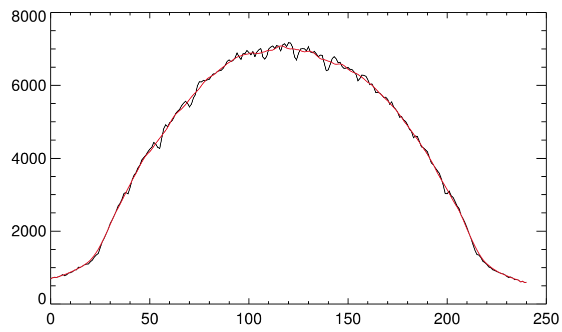
(b) A proposed mask that looks within a certain distance form the border.

Figure 3 The bottom corners will never see any data; the border mask must take into account the distance from the hypotenuse of the bottom corners still.

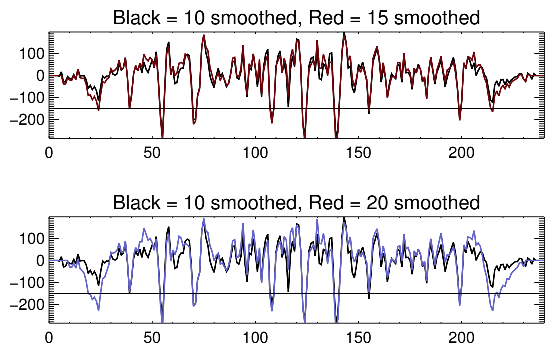
5 Fitting Limb Strips

6 Finding Fiducials

For each position below a certain threshold (see Figure 4b) a row/column is returned. Once we have an array of possible fiducial row and column positions, they are matched against each other using a method that iteratively checks to see if a pair of coordinates is a fiducial. Figure 5 is what each fiducial candidate



(a) This is the 1D sum of a solar image. Small dips are seen in the profile corresponding to fiducials. The line in red is the sum smoothed by 10 pixels.



(b) We subtract the smoothed profile from the raw data to emphasize dips. Comparisons of the smooth amount change the width and number of the dips, although not really the depth.

Figure 4

looks like for a certain sun. The position of plots in Figure 6 corresponds to the same regions in Figure 5.

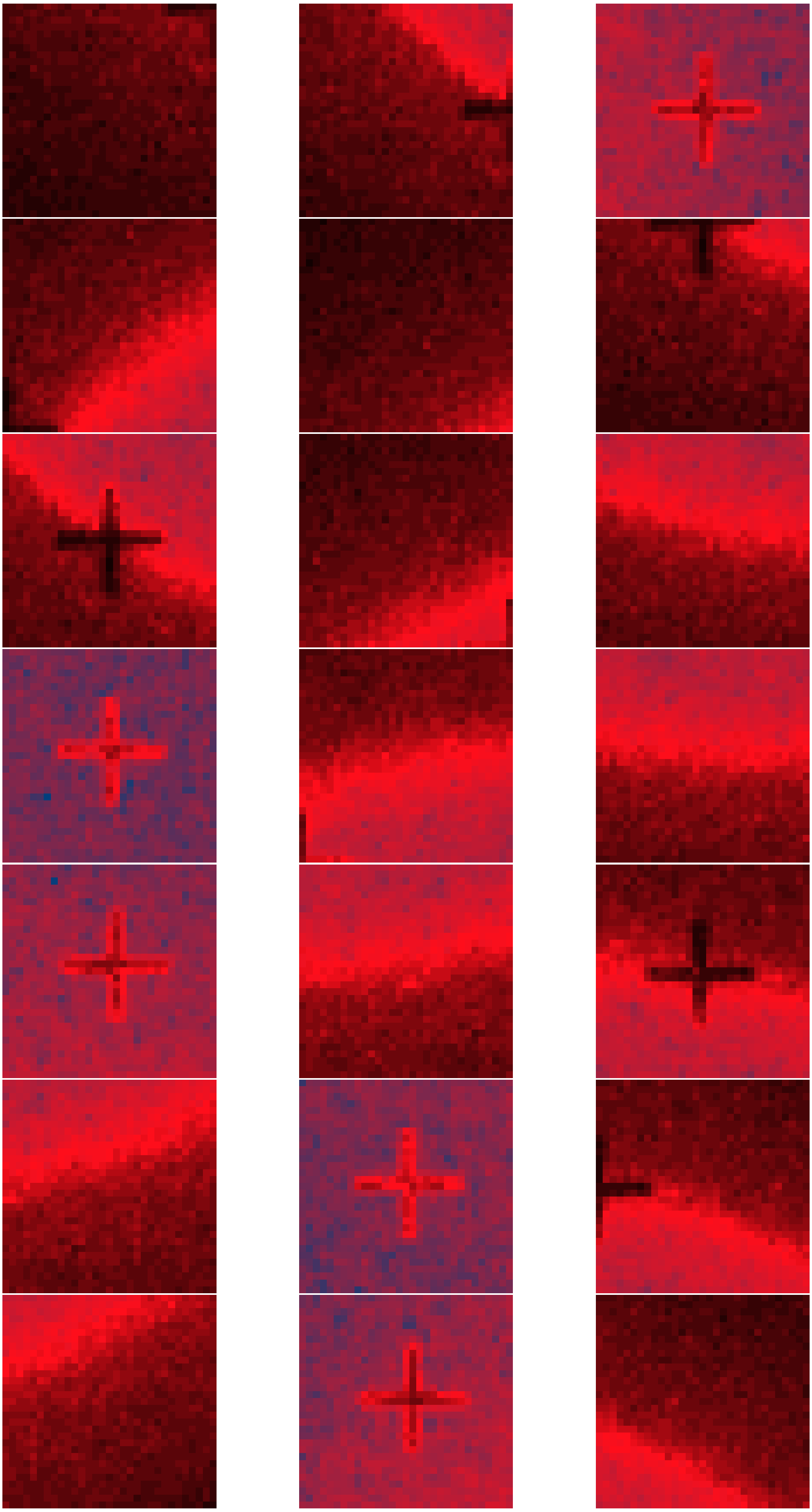


Figure 5 Each possible fiducial candidate

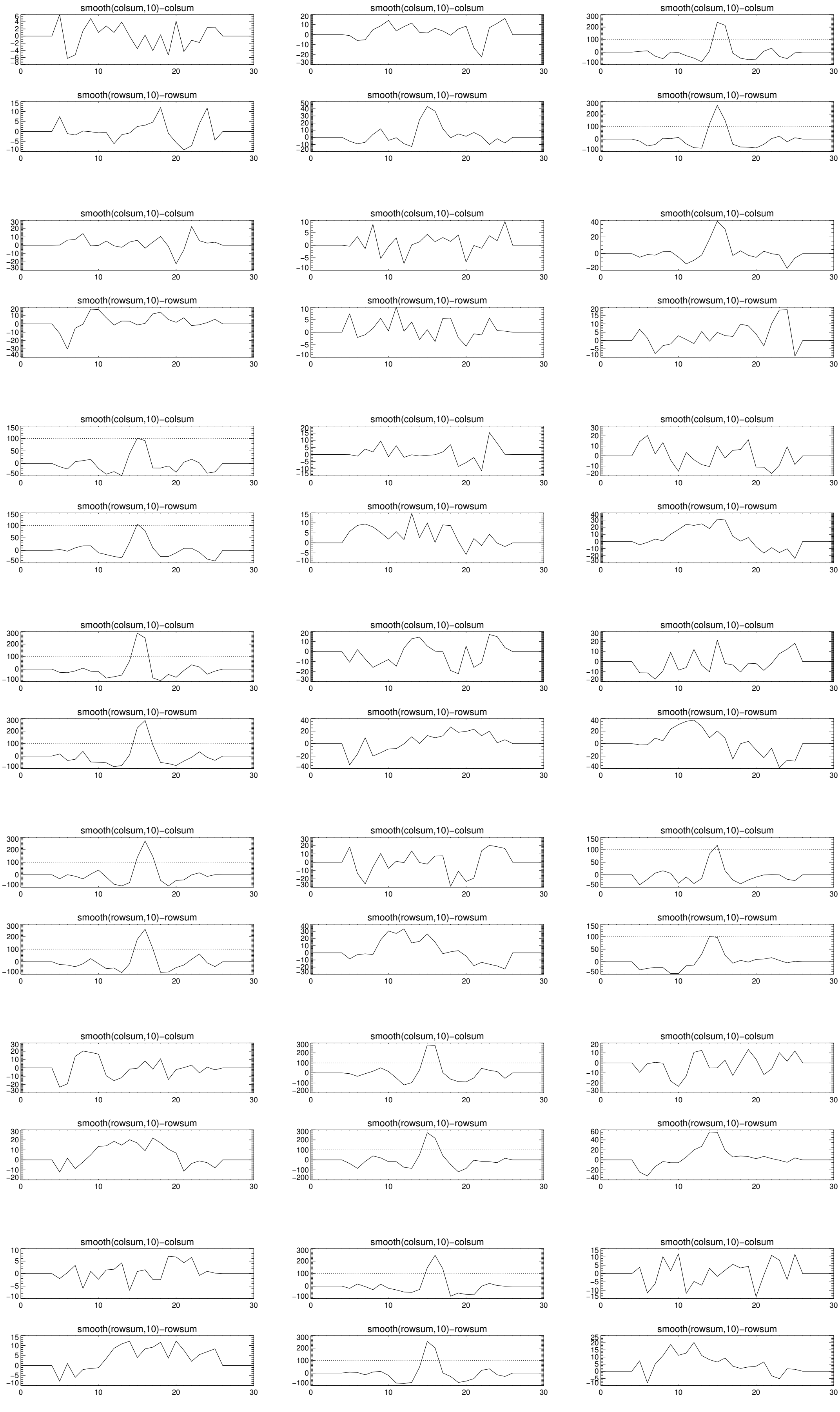


Figure 6 The horizontal line is at 100; if there are any elements of the array above 100 for both a column 1D sum and a row 1D sum, then the cropped area is identified to have a fiducial in it. A parabolic fit is applied to 3 consecutive pixels with the center pixel at the peak of the array.