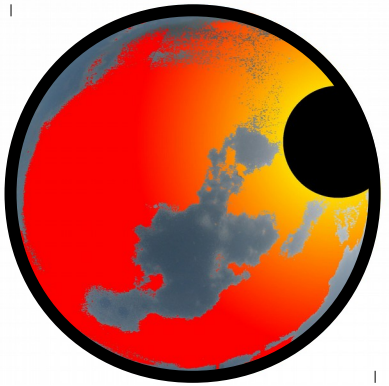
Barbados Cloud Observatory: All-Sky Cloud mask Algorithm (ASCA)



Marcus Klingebiel und Tobias Machnitzki

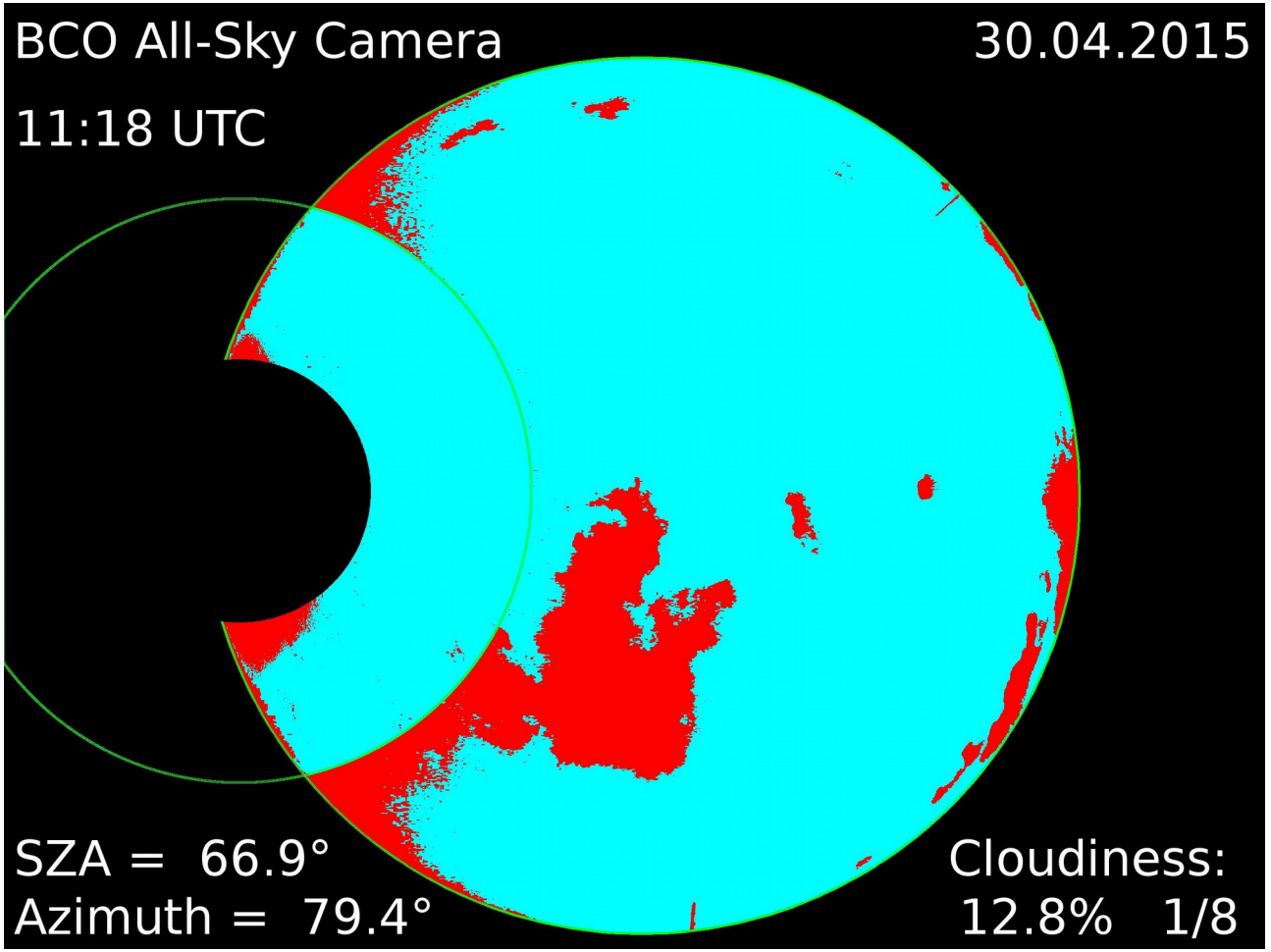
Max-Planck-Institut für Meteorologie
Hamburg

Purpose

- Improving computing time to < 1min
- Better handling of the glare (Don't recognize it as a cloud)
- Better recognition of clouds

So Far

Program works fine and produces results like this:



Ideas for the Improvement

So far:

- Calculating everything pixel by pixel
  → looping over 1944 x 2592 pixel causes long computing times

- For every operation the image itself is required

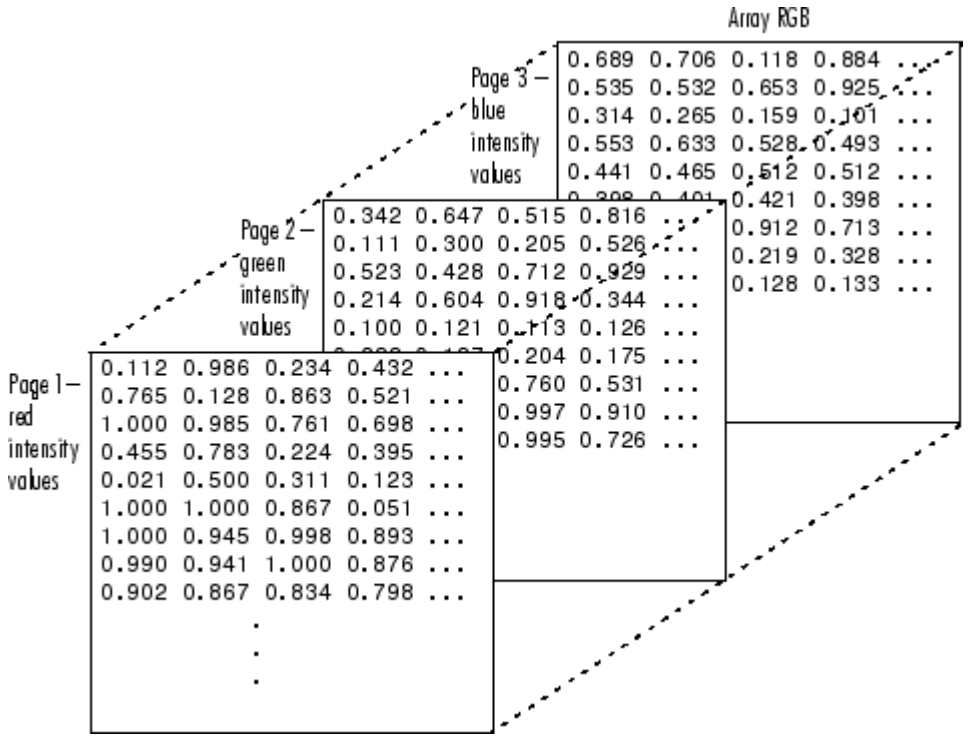
Solution:

- Converting the image to an array

- Using masks for getting rid of the loops

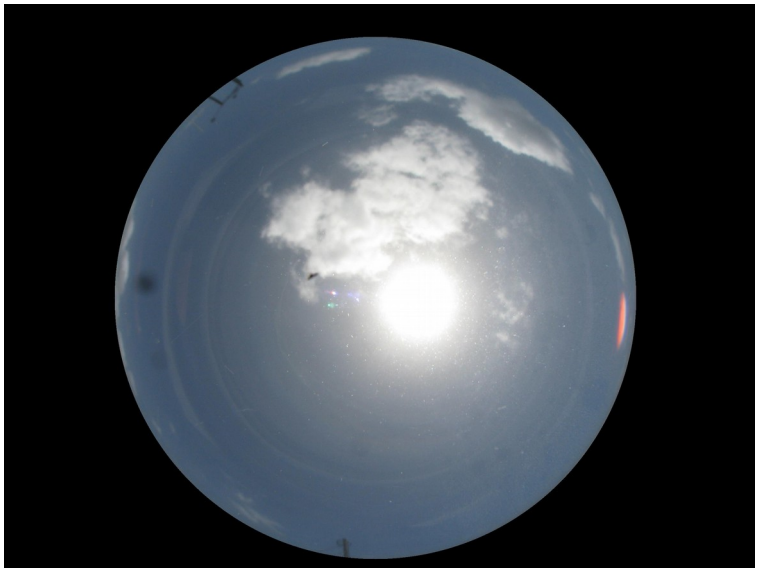
Structure of the array

- 3-Dimensional array as shown in the picture
  - First two dimensions for the location of every pixel
  - Third dimension has just 3 layers
    1)Red
    2)Green
    3)Blue



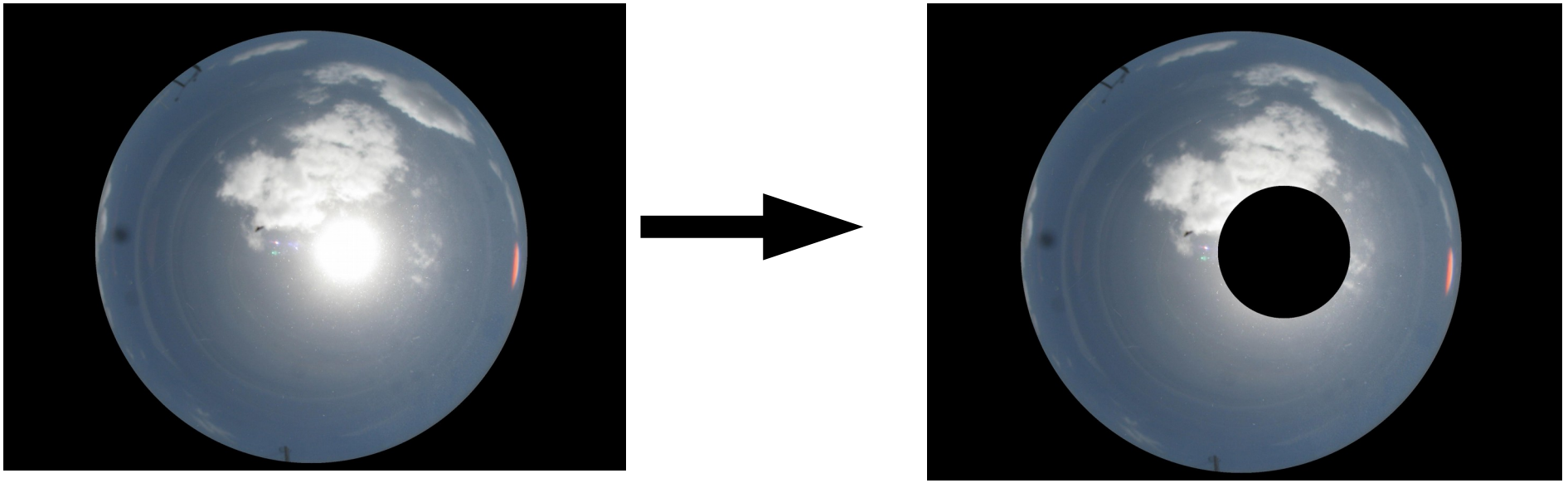
http://de.mathworks.com/help/matlab/math/multidimensional-arrays.html

Deleting unnecessary parts around the picture



- Mirror the image for easier calculation of sun position later
- Mask that covers just the part of the actual picture
  - Rest is colored black

Excluding the sun



- Mask that covers the sun
- Because clouds are indicated by th Brightness index and the Sky Index (SI), the sun has to be excluded. Else it would always be counted as a cloud

Calculation of the Sky Index

Sky Index:

$$SI = \frac{(DN_{Blue} - DN_{Red})}{(DN_{Blue} + DN_{Red})}$$

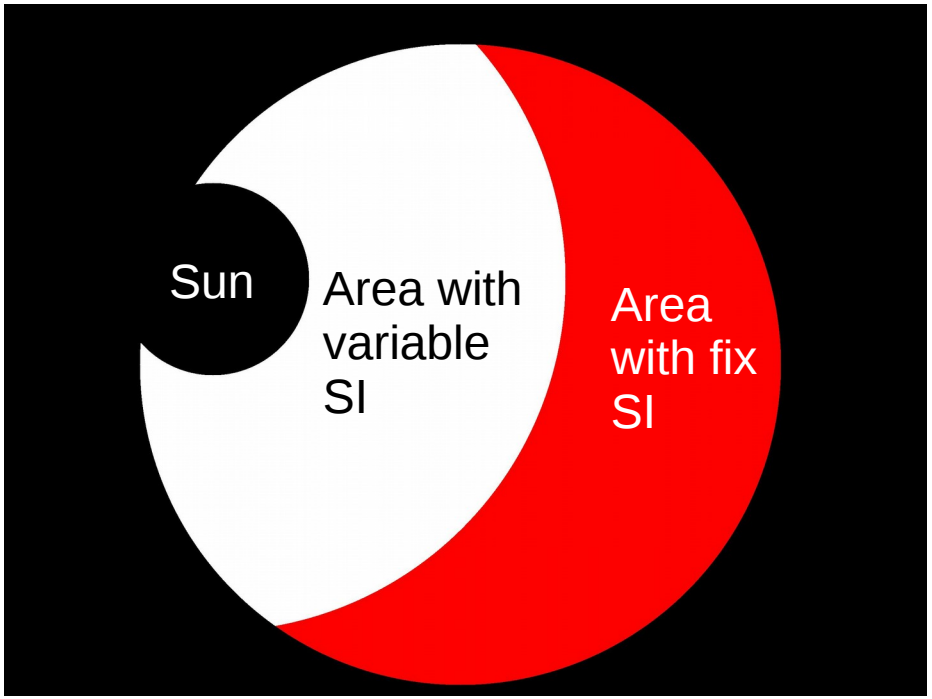
More cloudy

-1

More blue sky

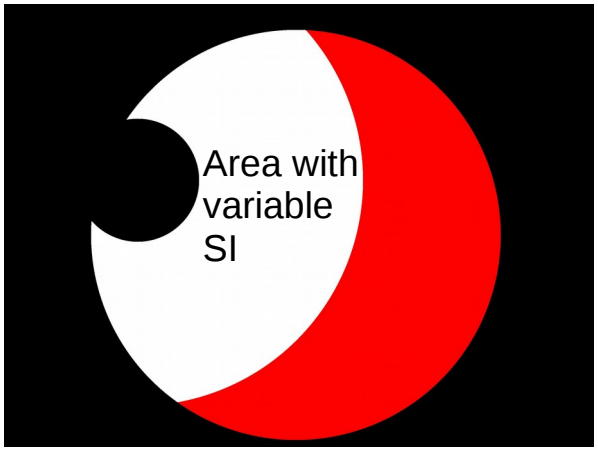
+1

Dividing image into two parts

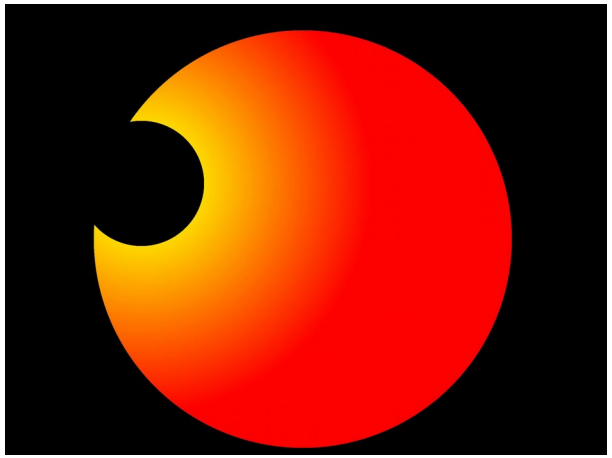


- Two Areas for different SI criterions
  - Area with fix criterion:
    - [SI < 0.1] is recognized as cloud
  - Area with variable criterion:
    - [SI < parameter] is recognized as cloud

Calculating the parameter



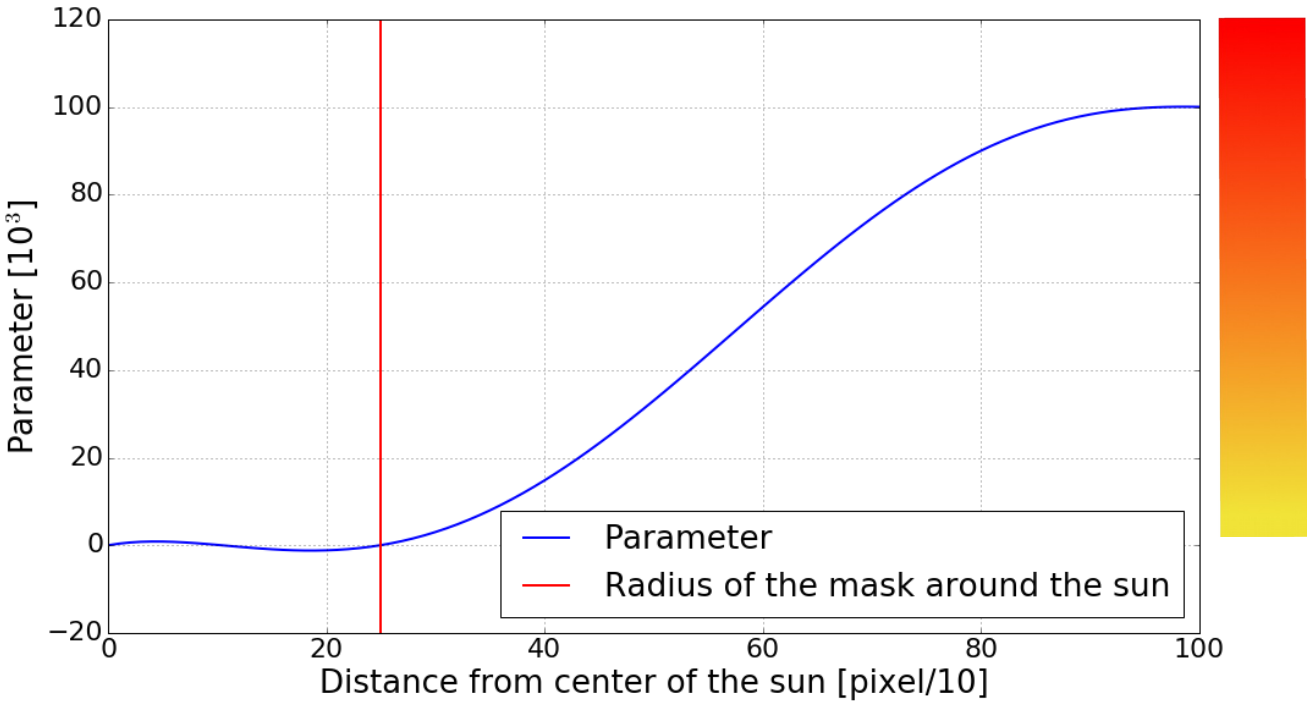
- Parameter has to be proportional to the distance to the sun
  - closer to the sun, the parameter has to be smaller (best: parameter = 0)
  - Further away from the sun the parameter has to be bigger (best, parameter = 0.1)



Calculating the parameter

- Fit a nonlinear curve on some guessed values for the parameter
  - Guesses: (0/0),(100/0,001),(270/0,01),(500/0,033),(800/0,09),(1000/0,1)
    → parameter:

$$f(x) = (0.4424283716980435\,x - 0.06676211439554262\,x^2 + 0.0026358061791573453\,x^3)$$
$$(-0.00002941713087331177\,x^4 + 1.029285214959394 * 10^{-7}\,x^5) * 0{,}001$$



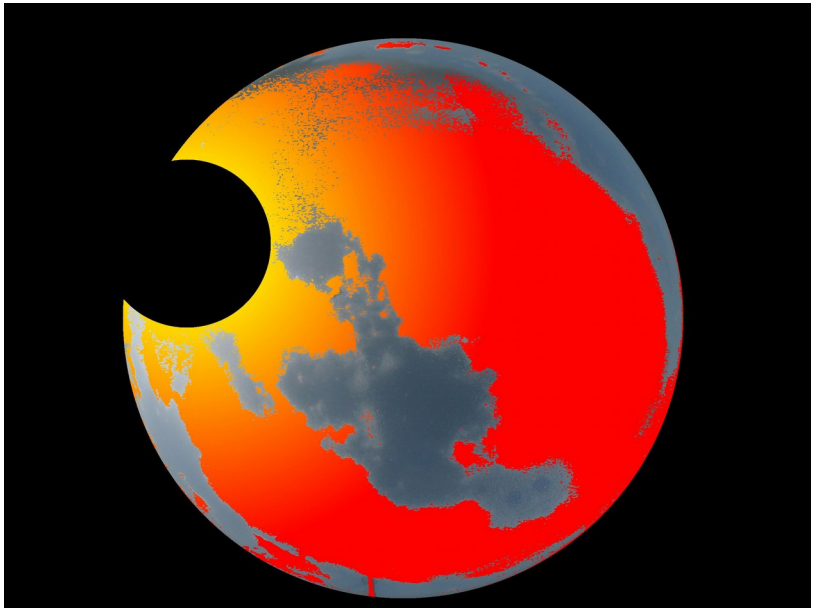
Explanation of the Guesses

- (270/0.01): This is where the mask of the sun ends, so where the detection of clouds start. Because of the high brightness in this part of the picture the parameter needs to be really small.

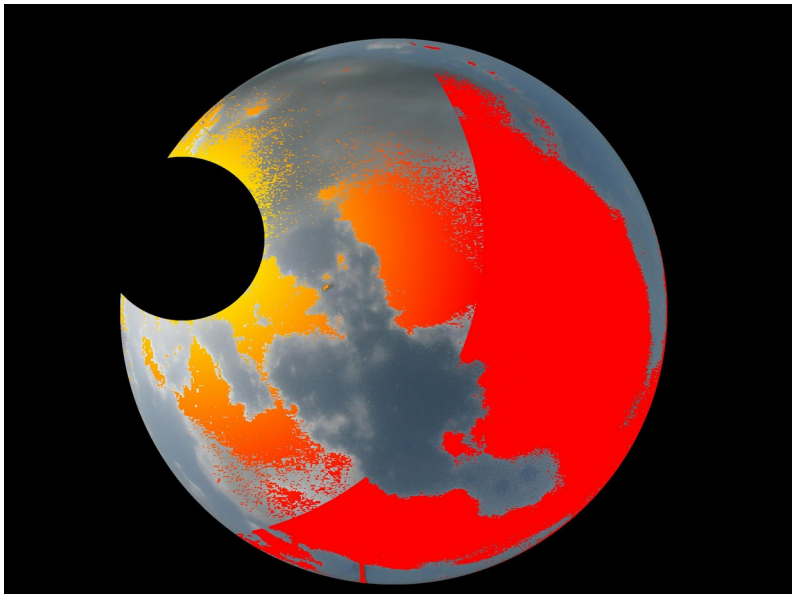
- (1000/0.1): After the passage of the Area with variable SI-parameter, the parameter will be fix at 0.1. So to get a fluent passage, the last value of the Area with the variable parameter needs to be 0.1, as well.

Explanation of the Guesses

A fluent passage between the two areas is recommended, as otherwise you will get 'ugly' edges:



Right



Wrong

Explanation of the Guesses

- (270/0.01): This is where the mask of the sun ends, so where the detection of clouds start. Because of the high brightness in this part of the picture the parameter needs to be really small.

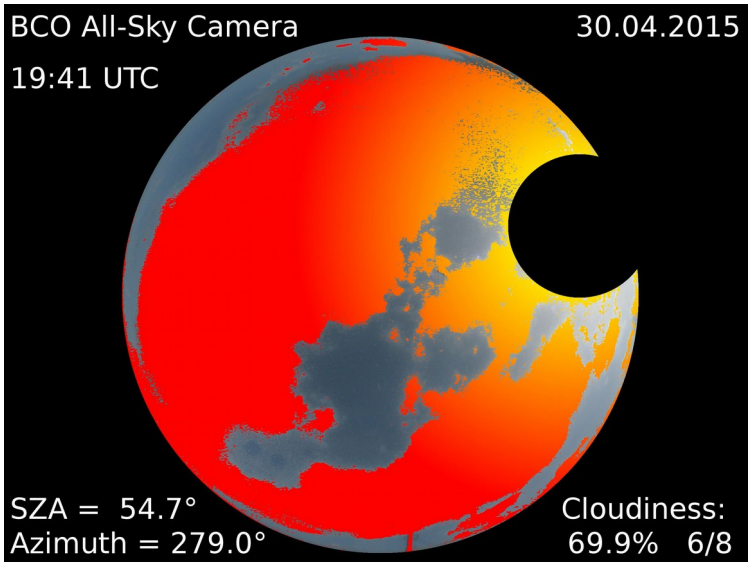
- (1000/0.1): After the passage of the Area with variable SI-parameter, the parameter will be fix at 0.1. So to get a fluent passage, the last value of the Area with the variable parameter needs to be 0.1, as well.

- All other tuples found out by trial and error, so that the glare gets minimized and the cloud recognition gets maximized.

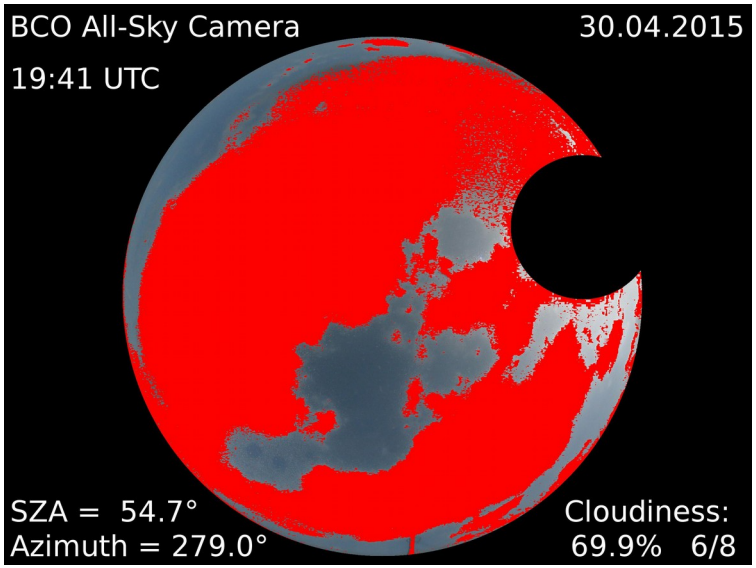
Calculating the Cloudiness

1. Making 2 arrays out of the picture
   a) The one for printing the image, where the SI-parameter is colored
   b) One for calculating, where everything getting recognized as cloud is colored red (255,0,0)
2. Use array b) and make it 2-dimensional by summarizing all colors (sum(r,g,b))
3. Counting how many values of that counting-array are exactly 255

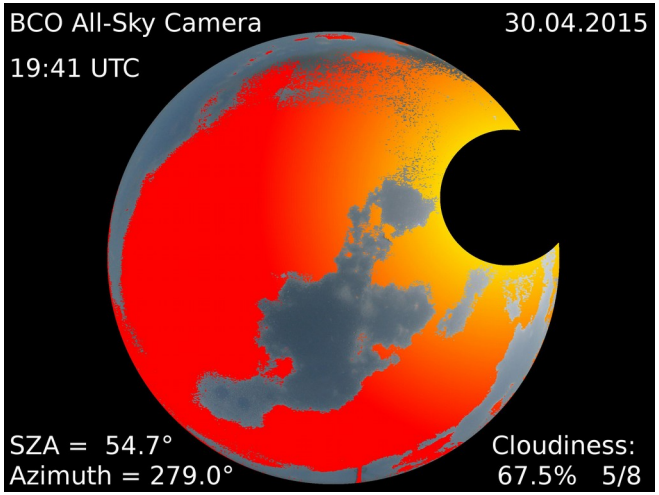
a)



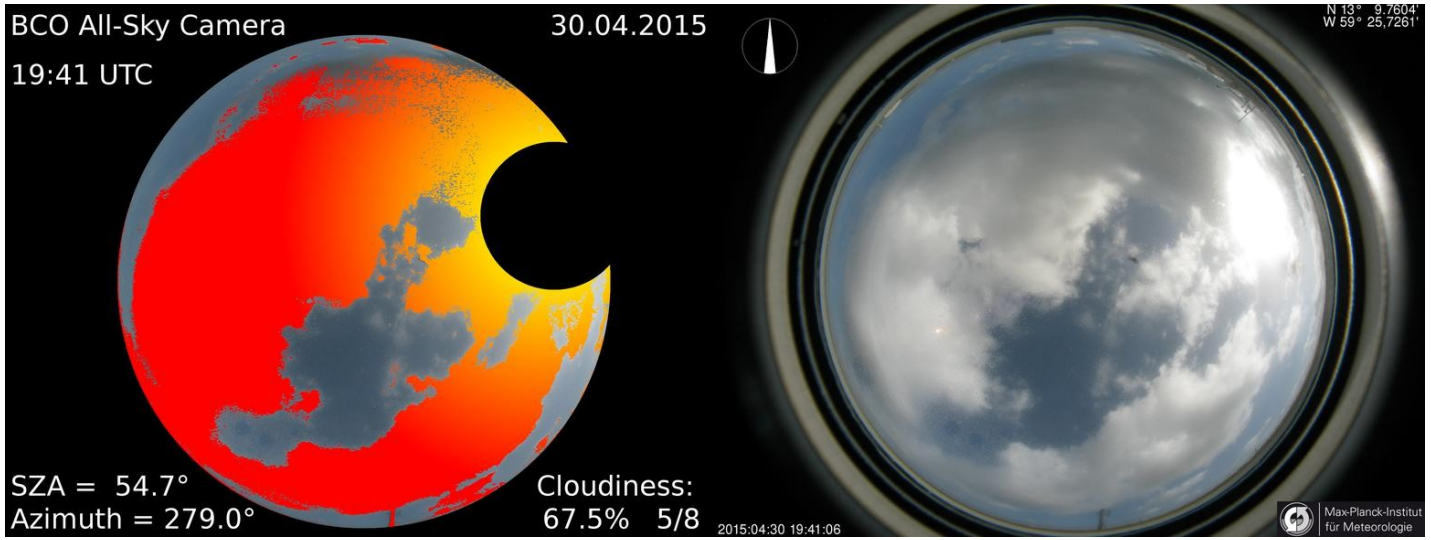
b)



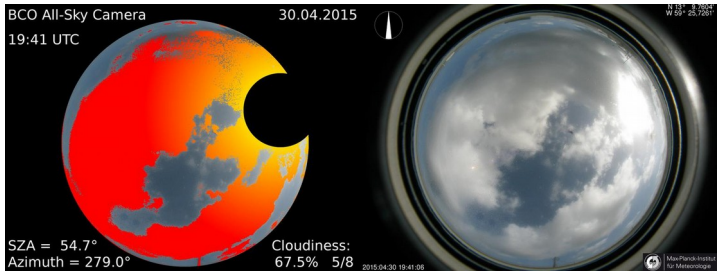
Saving the Image (and Downscaling)



+

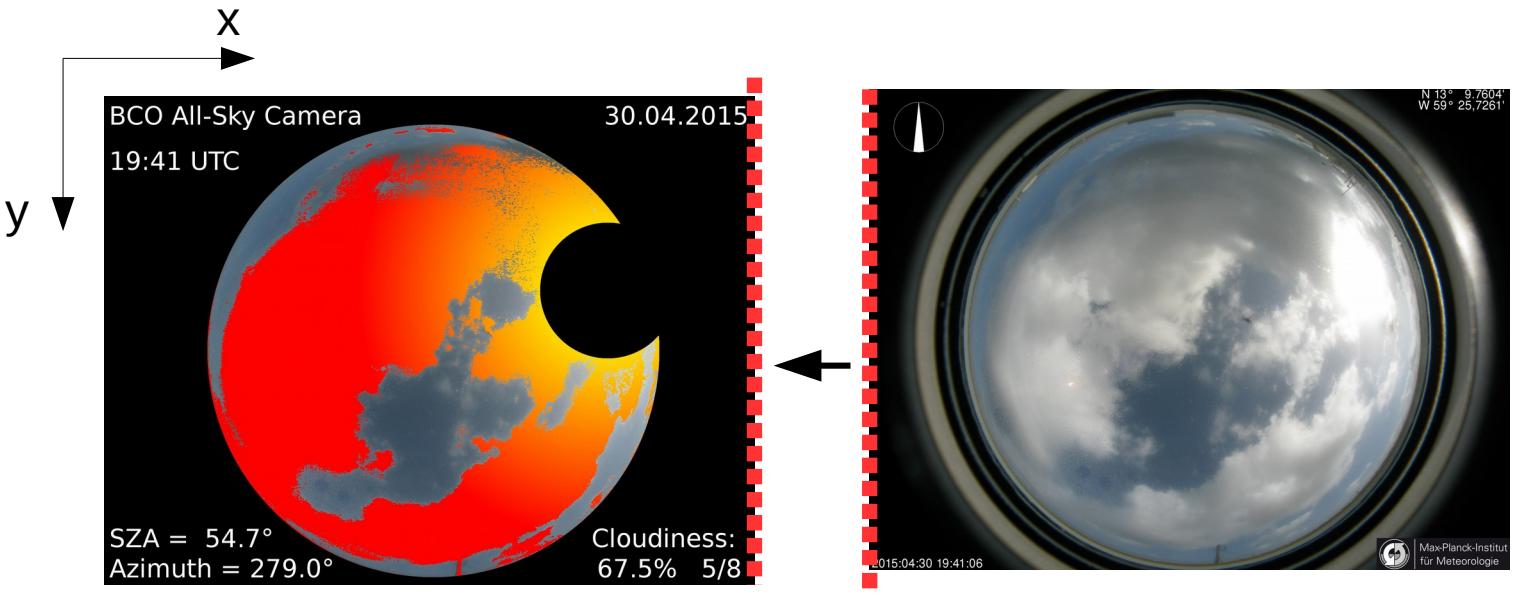


Saving the Image (and Downscaling)



For saving both images as one:

- Using the arrays of the original image and the on of the processed image
- Simply append one Array to the other at the second axis (y-axis)



Purpose

- Improving computing time to < 1 min
  - Is now < 10 sec

- Better handling of the glare (Don't recognize it as a cloud)
  - Especially around noon the glare makes now less than 2% of the cloudiness

- Better recognition of clouds

Further Ideas

Fehlerquellen

- Automatic corrections by the camera
  - Whitebalance → SI sometimes better sometimes worse, depending on the setting for each picture
  - Autofocus:
    - Same for SI as Whitebalance
    - Blurred pictures have worse contours
      → Edges of the clouds don't get recognized

Further Ideas

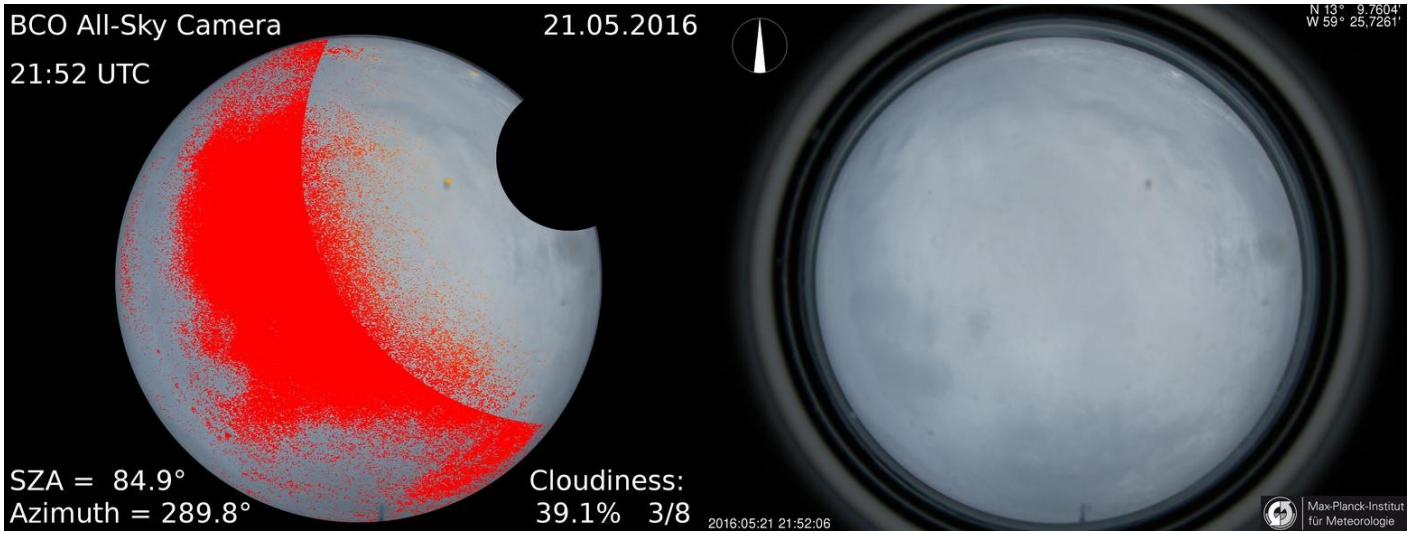
Further Ideas

- Den SI als “Legende” am Rand angeben

- Windvektoren mittels korrelation und Verschiebung des Bildes in alle Richtungen
  - Dazu muss das Bild aber erst mal entzerrt werden
  - (Idee von Yanns Batchelorarbeit)

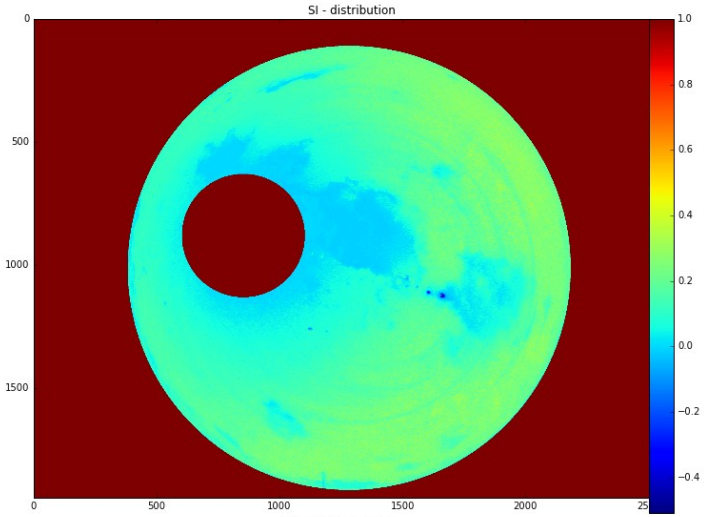
Further Ideas

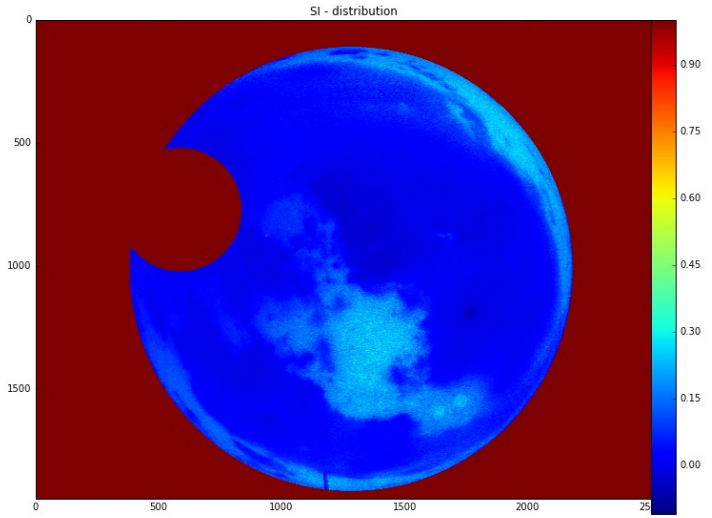
- Changing the SI criterion with the waehter and time:
  - Cloudiness doesn't change from 1/8 to 6/8 over one minute
    → Using the last picture for validating the actual picture could solve some underestimation at really cloudy weather conditions
    - → The Program would get a statistic part
  - making the SI criterion a function of the time could prevent mistakes like this:

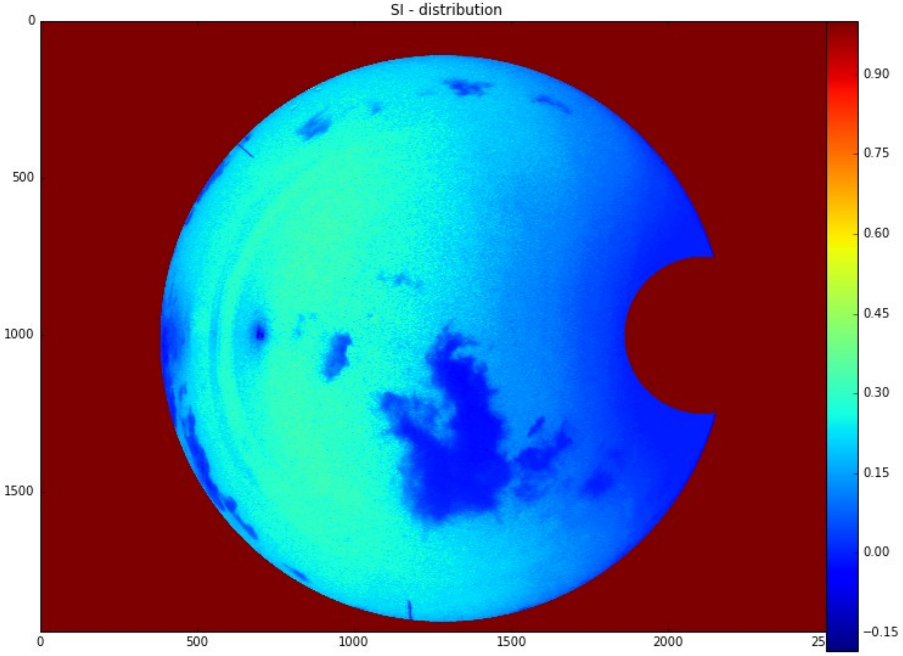


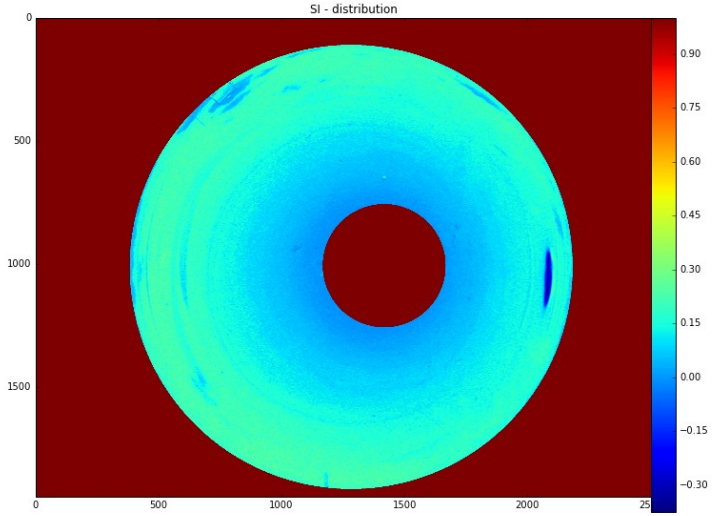
Last shot on the 21.05.2016 before sunset. Its easy to see, that the cloudiness is rather 8/8 than 3/8.

Further Ideas



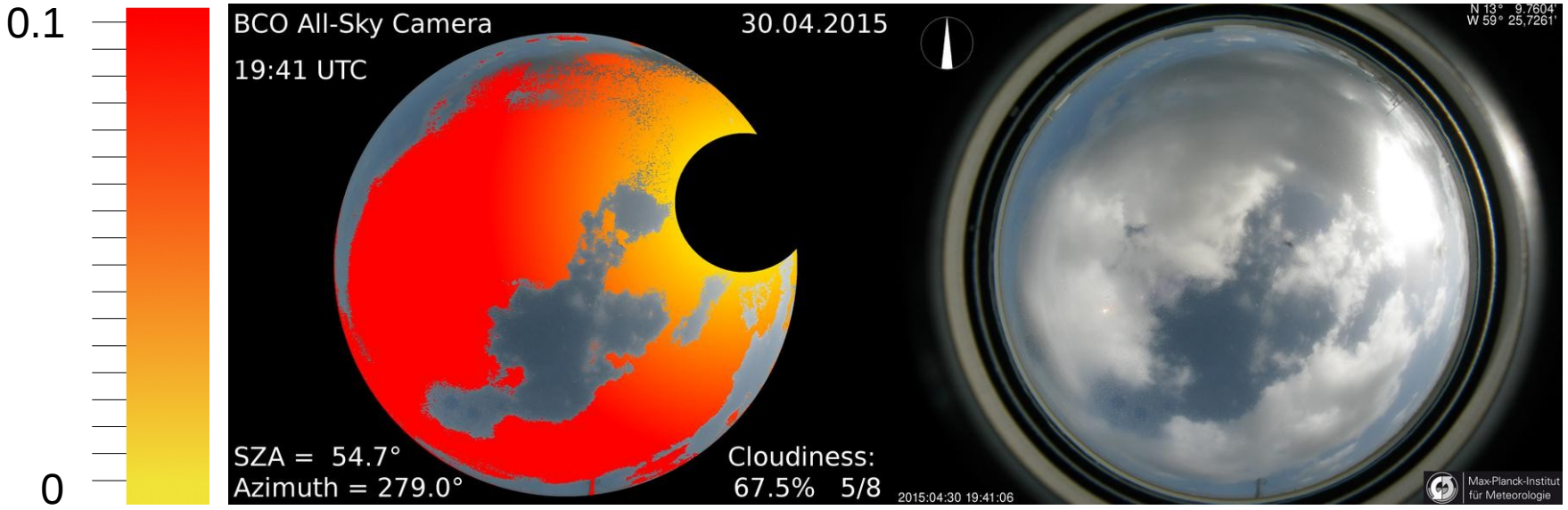






Further Ideas

- Presenting the SI as a Legend of the Side of each picture like this:



Further Ideas

- Calculating the wind at the hight of the clouds by moving the actual image at all positions over the last picture and find out where the correlation is the biggest
    - Therefore a ceilometer is needed
    - The distortion of the fish eye objective on the camera needs to be straightened out