Analysis and Improvement of the Single-CCD Pointing-Image Simulation

Bachelorarbeit aus der Physik
vorgelegt von

Marco Egelkraut

Tag der Abgabe: 21. Juni 2019

Erlangen Centre for Astroparticle Physics
Physikalisches Institut
Friedrich-Alexander-Universität
Erlangen-Nürnberg

1. Gutachter: Prof. Dr. Christopher van Eldik
2. Gutachter: Prof. Dr. Stefan Funk

Abstract

The Cherenkov Telescope Array (CTA) is the next generation ground-based observatory for gamma-ray astronomy at very-high energies. With more than 100 telescopes located in the northern and southern hemispheres, CTA will be the world's largest and most sensitive high-energy gamma-ray observatory. CTA is currently in the Pre-Construction phase [1]. In order to determine its pointing, a single-CCD concept can be used. This CCD-camera pictures the night sky and the telescopes camera simultaneously. The observed star constellations are then used to determine the direction of the image.
This thesis investigates the quality of simulated single-CCD images. Therefore, real and simulated images were compared, using the results to improve the simulation. This thesis introduces a possibility to simulate images rotated around a given rotation point and it shows that using the focal length as specified by the manufacturer of the lens generates considerable distortions. This could be prevented by using a calculated value. Eventually, the simulated brightness of the background and of the stars can be successfully attuned to the real conditions.

Zusammenfassung

Das Cherenkov Telescope Array (CTA) ist ein erdgebundenes Experiment der nächsten Generation zur Beobachtung von hochenergetischer Gammastrahlung. Mit mehr als 100 Teleskopen, verteilt auf 2 Standorten, wird CTA das weltgrößte als auch empfindlichste Experiment seiner Art sein. Das Projekt ist aktuell in der Prototypenphase. Um die Beobachtungsrichtung ("Pointing") eines Teleskops zu bestimmen wird ein Single-CCD Konzept benutzt. Das heißt, dass die CCD-Kamera sowohl die Kamera des Teleskops als auch den Nachthimmel ablichtet. Die dabei fotografierten Sternenkonstellationen werden dann benutzt, um das Pointing zu bestimmen.
Diese Bachelorarbeit untersucht die Qualität von simulierten Single-CCD Bildern. Dazu wurden echte mit simulierten Bildern verglichen, wobei die Ergebnisse zur Verbesserung der Simulation genutzt werden. Diese Bachelorarbeit entwickelt die Möglichkeit, um einen gegeben Punkt rotierte Bilder zu simulieren. Des Weiteren wird gezeigt, dass die Nutzung der vom Hersteller des Objektivs angegegebenen Brennweite in der Bildsimulation zu nicht vernachlässigbaren Abweichungen führt. Diese können jedoch durch den Gebrauch eines berechneten Wertes verhindert werden. Schlussendlich kann die simulierte Helligkeit von Sternen und Hintergrund erfolgreich der Echten angepasst werden.

Contents

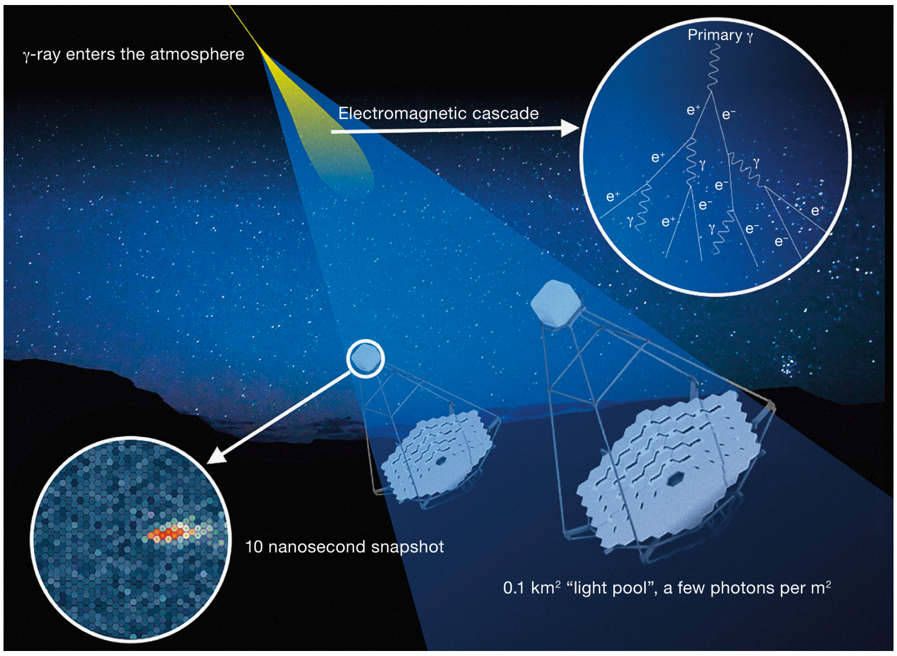
γ-ray enters the atmosphere
Electromagnetic cascade
Primary γ
e+
e−
γ
10 nanosecond snapshot
0.1 km² "light pool", a few photons per m²

Figure 1: This image shows a gamma ray entering the atmosphere, producing a Cherenkov light cone with a typical diameter of 250 m [1]. CTAO/ESO

1 Introduction

1.1 Gamma-Astronomy

Investigating the night sky has always been a fascinating task. At first, this was only done in the range of visible light, since suitable measuring instruments were not invented yet. As the understanding of electromagnetic waves has been improved, the idea arose to survey the sky with different wavelengths. As the atmosphere is not transparent for all wavelengths, this is mainly done with radio- and infrared-telescopes. In order to observe gamma-rays with Very High Energies (VHE) directly, a ground based approach is, because of the atmospheric absorption, not possible. One possibility is to use space telescopes, such as the Fermi Gamma-ray Space Telescope. Since events with high photon energies are increasingly rare, photons should be detected in an area as large as possible. Realising this using a space telescope would be very expensive or not even possible.

Therefore it's helpful to observe photons with very high energy indirectly, using the effects of atmospheric interaction with gamma-rays. When such a photon enters the atmosphere, it can initiate an electron-positron pair production, generating particles with very high energies. These particles undergo bremsstrahlung: They emit photons. Those photons can be such energetic, that they can initiate an electron-positron pair production again. This leads to a particle cascade, shown in Figure 1. This cycle continues until the energy

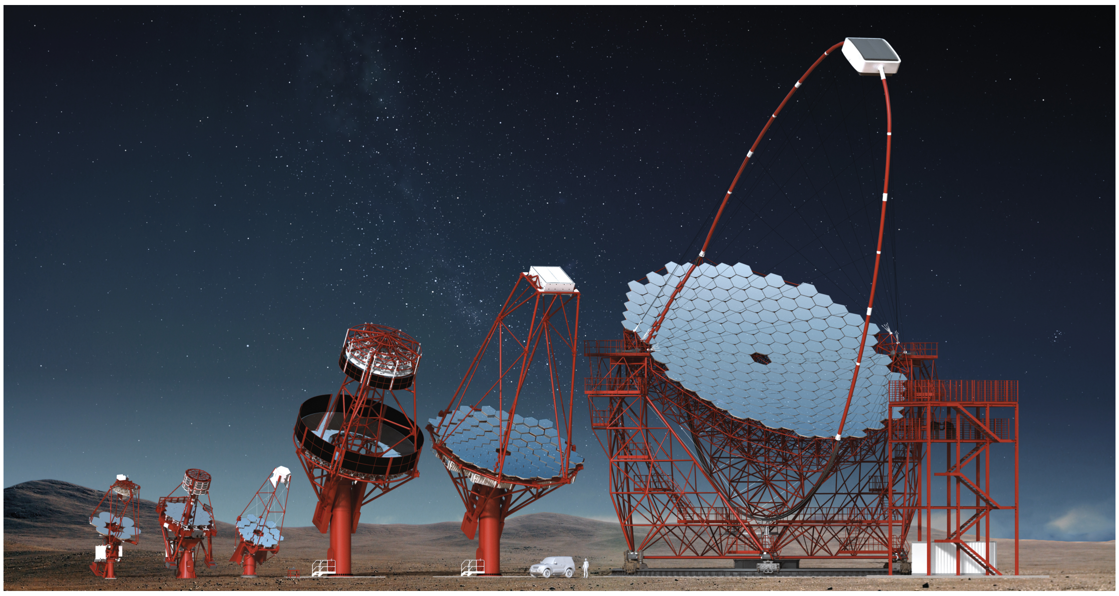

Figure 2: This is a rendered image of the different proposed CTA telescopes. SST (left), MST (centre) and LST (right). Gabriel Pérez Diaz, IAC

of the bremsstrahlung is no longer sufficient to produce electron-positron pairs. If these particles are travelling faster than the local speed of light, they emit Cherenkov radiation (a blueish light). Although this Cherenkov radiation only lasts a few nanoseconds, it can be detected by ground based Cherenkov Cameras [1]. This method of gamma-ray detection is called Imaging Atmospheric Cherenkov Technique (IACT).

One experiment using the IACT is the CTA, "the next generation ground-based observatory for gamma-ray astronomy at very-high energies. With more than 100 telescopes located in the northern and southern hemispheres, CTA will be the world's largest and most sensitive high-energy gamma-ray observatory." CTA is currently in the Pre-Construction phase. It contains three classes of telescopes: [1]

- Small-Sized Telescopes (SSTs) with energy range of 5 TeV to 300 TeV
- Medium-Sized Telescopes (MSTs) with energy range of 150 GeV to 5 TeV
- Large-Sized Telescopes (LSTs) with energy range of 20 GeV to 150 GeV

1.2 Pointing Solutions

In order to determine the origin of an observed gamma-ray, images from different cameras are combined. The light of an air shower produces an elliptical shape on each image. As the different cameras are located at different positions, these ellipses are typically orientated differently. Therefore it is possible to determine the gamma ray's origin by

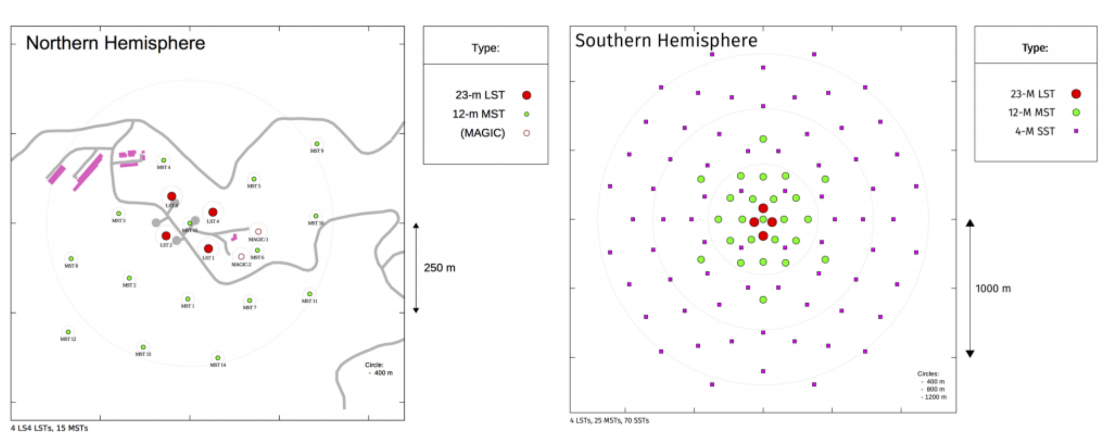

Figure 3: This map shows the planned location of the different telescopes at the two different sites. CTA Collaboration

estimating the intersection point of the ellipses' major axes. In order to match the origin's position on the image to a position on the night sky, it's necessary to know the observed direction of the telescopes (pointing) very precisely. The accuracy of the calculated origin of one source can be increased further by averaging over many events. This thesis concentrates on the pointing of MSTs. For these telescopes, the goal is to achieve a precision of $7''$ [2].

1.2.1 Pointing Runs

The nominal, adjusted pointing of the MST-telescope does not satisfy this precision requirement. The errors of the nominal pointing can be classified as reproducible or irreproducible errors, whereas reproducible ones, like offsets and bending of telescope components, can be corrected by a model [4]. This can be reached by so called pointing runs. Hereby the telescope is aligned to a bright star. Then the direction of the telescope will be slightly altered until the bright star is exactly in the centre of the Cherenkov-Camera. If this is done for multiple stars at different positions, the resulting pointing deviations can be used to create a pointing model. One disadvantage of this solution is, that the telescope is not available during pointing runs. Another problem are irreproducible errors, like deformation caused by wind pressure or obstacles on the drive rail, since they cannot be corrected by pointing runs [3].

1.2.2 Astrometry and CCD-Pointing

Another very common method is astrometry. Hereby the pointing is calculated using known positions of visible objects in the image. Astrometry can be done while the telescope is online and, in addition, can also correct irreproducible errors. Therefore the precision of astrometry towards pointing runs is in general higher.

However, the number of suitable gamma-band sources is far smaller compared to other regularly used wave bands. Therefore it is likely that an image does not provide enough known sources to calculate the pointing, especially since the MST covers only a field of view of less then 8°. Hence it is not possible to use this method for CTA directly.

Instead, an additional CCD-camera which is firmly connected to the whole telescope is used. This CCD-camera captures the position of visible stars and uses them as known sources to calculate the pointing of the image. There are two concepts using this idea: a single-CCD and a dual-CCD pointing. The dual-CCD pointing uses two CCD-cameras: The LidCCD is aligned to the CTA-camera whereas the SkyCCD is capturing the night sky. In comparison, the single-CCD concept uses only one camera which captures the CTA-Camera and the night sky simultaneously.

Since the dual-CCD system doesn't need to cover both the CTA-Camera as well as the sky, the field of view of those cameras can be, compared to the single-CCD camara, smaller. This means, that the optical aberrations of the dual-CCD system are reduced and the resolution per field of view is higher. Another advantage is, that both the CTA-Camera as well as the sky can be correctly focused. When using a single-CCD concept, this is not possible. The CCD must be focused on an imaginary plane, so both the CTA-Camera as well as the stars are equally blurred. Using a single-CCD concept is still reasonable, since it reduces the costs (maintenance and acquisition) and the pointing's complexity.

1.2.3 LibPointingMST and LibCCDCamera [5]

This thesis focuses on the single-CCD concept. The software used to calculate an image's pointing is **LibPointingMST**. Firstly, it uses a spot extraction algorithm to get the pixel position and the brightness of a spot. Then it tries to match this data set to a star catalogue, in order to determine the Ra-Dec Positions of the captured stars. With this information, the observed direction can be calculated. Furthermore, the rotation angle of the CCD-image can be determined.

In order to determine the precision of the single-CCD pointing, the library **LibCCD-Camera** was developed. It is able to simulate images of the sky, using the exact direction and time to rebuild the star constellations accurately.

For this reason, it is important that the simulated images and the real ones are as similar as possible. As a result, the assignment of this thesis is to compare simulated and real images, identify possible deviations and correct them if possible.

2 Implementation of GetStarID

In order to enable detailed comparisons between different images, it's necessary to track one star over several images. In LibPointingMST, each star is an object of the class PointingStar. At first, the attribute StarID, the methods getStarID() and setStarID() were added to this class. As StarID, the position of the star in the index file is used. In order to enforce that one ID only belongs to one star at maximum, the possibility to use multiple index files was deactivated.

This ID can be used by a C++ program named **getStarID**, which is derived from the existing **reconstruct-images** program. It takes an image as input and calculates its pointing. Afterwards, it writes for each star its ID, pixel position, magnitude and flux in an ASCI-file. In addition, it gives the original output of **reconstruct-images**, containing the pointing and the rotation of the image.

In order to calculate the Alt-Az direction of an image correctly, it's necessary to know the time at which the image was taken. This information can be gathered in the FITS-Header of the image. However, the simulated images do not save this parameters in their headers. Therefore the simulation of images was extended,so that the header of the simulated image contains now all relevant information, resembling the real images. The added flags were EXPSTART, EXPOSURE, TEMP and F-NUMBER, whereas the last 3 flags are needed to resimulate a simulated image.

3 Brightness of Spots and Background

3.1 Brightness of the Background

One important point of the simulation is the brightness of the spots and of the background. When the program **reconstruct-images** is used, it generates a background image, which models the background without the spots. In order to determine the brightness of a spot, the calculated background image is evaluated at the spot's pixel position and then subtracted from the measured brightness. Since the background affects the calculated brightness of the spots, it is important that the background of the simulated image matches the real one. To analyse the background, the program **getStarID** was extended. It evaluates now the brightness of the calculated background at each spot and saves this value in the StarID.txt file.

Figure 4 shows the calculated background of a real CCD-image taken with a CCD-Camera mounted on the MAGIC-Telescope. It's noticeable, that the calculated background is not uniformly distributed, but rather is the lower left corner brighter then the remaining parts of the image. This can be caused either by non-uniform efficiency of the CCD-Chip or by light pollution of the atmosphere. In comparison, the simulation models the background to be uniformly distributed. Since there are currently no images from CCD-cameras

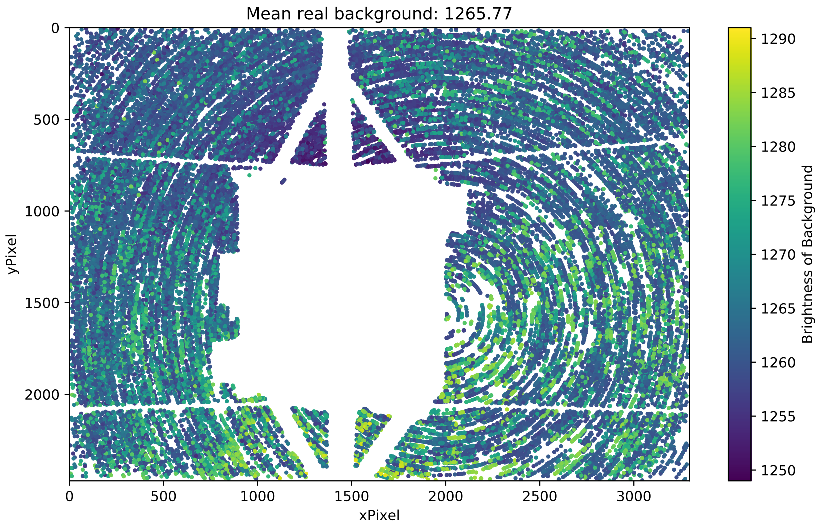

Figure 4: This figure shows the calculated background of real ccd-images taken at the MAGIC-Telescope. The calculated background was evaluated at each recognised spot. It's visible that the background's lower right corner is brighter. This can either be caused by a flawed CCD-Chip or by light pollution of the atmosphere.

mounted on MSTs available, it was decided to don't pursue this issue further at this point. In addition, this deviation is only around 3 % and can therefore be neglected at this state.

More important is, that the mean brightness of the real and simulated background coincides. In order to determine this, images of one single night were analysed. As Figure 5 shows, the simulation assumes a Gaussian distribution of the background brightness. The distribution of the real images is not as clearly identifiable, since it has a tail of higher brightness values, generating a broader distribution. The main difference between the simulations and the real images are, that the mean brightness of the simulations' background is 209 to high. This was fixed by subtracting this constant in the background of the simulation.

The result of this correction can be seen in Figure 6. This figure shows the brightness difference of the background at each spot between the simulation and the real images. For this comparison a random set of 98 images was analysed. As the mean value of the residual deviation is -2, the correction can be regarded as successful.

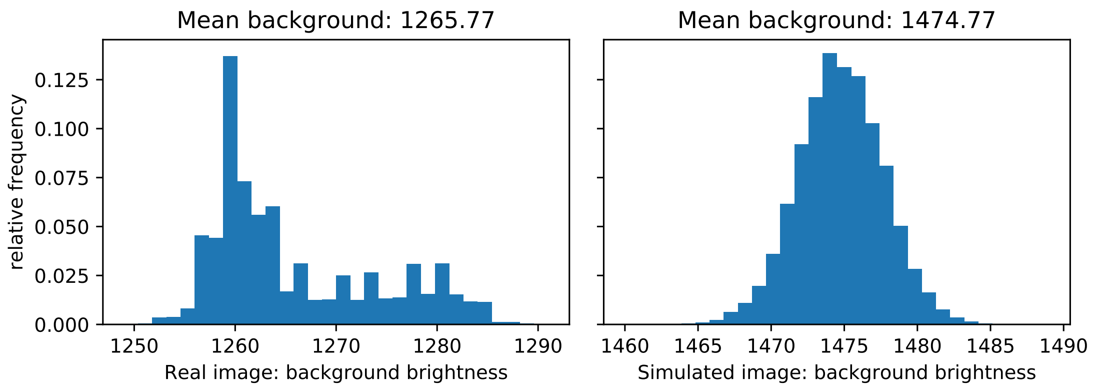

Figure 5: This figure shows the distribution of the brightness of the background at the pixel positions of the recognised spots. One can see clearly that the brightness distribution of the simulation is a Gaussian, whereas the real distribution is not as clearly identifiable. The real values are more widely distributed.

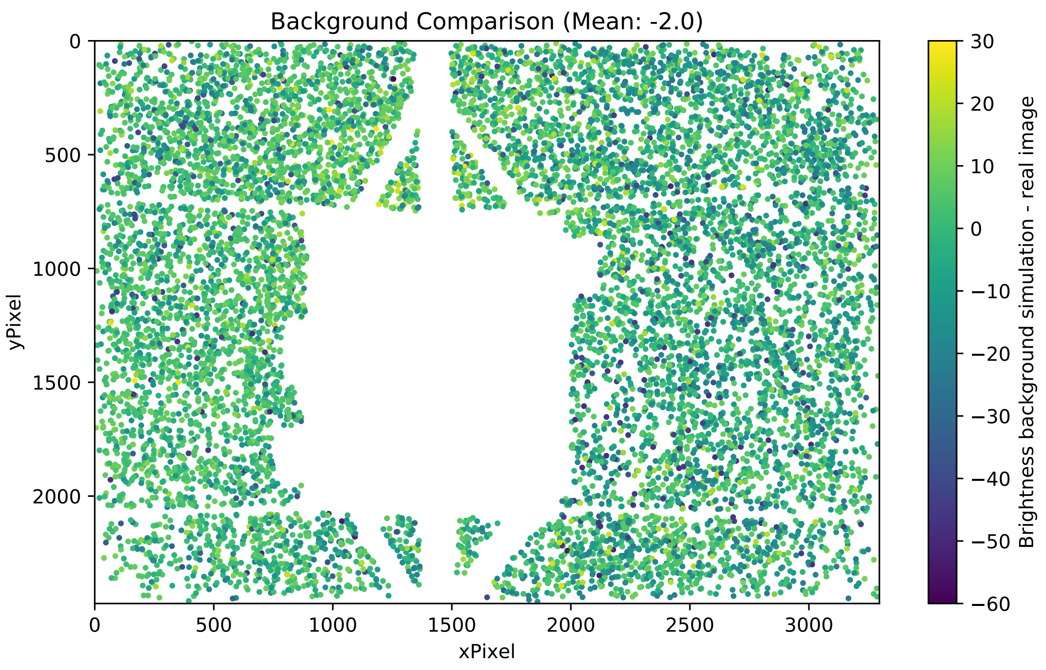

Figure 6: This figure shows for each spot the brightness difference of the background of the simulated to the real image. Since the mean difference is at -2, the background correction can be regarded as successful.

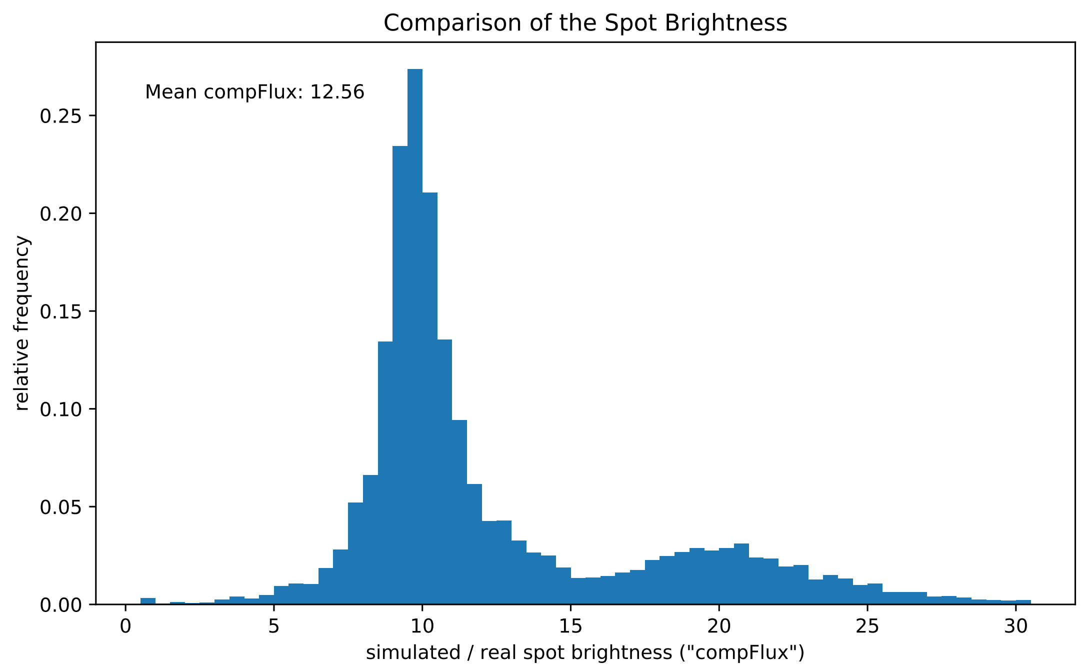

Figure 7: This figure shows the compared spot brightness of the real and simulated images. Therefore the brightness of a simulated star is divided by it's counterpart in the real image. This normalised value is called *compFlux*. The histogram shows that the simulation is in the mean 12.5 times to bright. However, the histogram is divided into two peaks, one at 10 and one at 20. This means, that most of the stars are simulated 10 times to bright. In addition, there is a small share of simulated spots which are 20 times to bright.

3.2 Brightness of the Spots

Therefore, the next step is to look at the correct brightness of the simulated spots. This property can be analysed by comparing the brightness value of a simulated spot with it's counterpart in the real image. This data can be accessed by **getStarID**. In order to compare the difference between simulated and real spot brightness consistently, it's necessary to normalise the brightness values. This is done by dividing the simulated brightness by it's real equivalent. This normalised value is henceforth called *compFlux*.

$$\text{compFlux} = \frac{\text{simulated spot's brightness}}{\text{real spot's brightness}} \tag{1}$$

As a first comparison, it is helpful to look at a histogram of the compFlux values. Therefore a random set of 100 images and their simulations where analysed, displayed in Figure 7. This histogram shows 2 peaks, one at compFlux = 10 and one at compFlux = 20. The

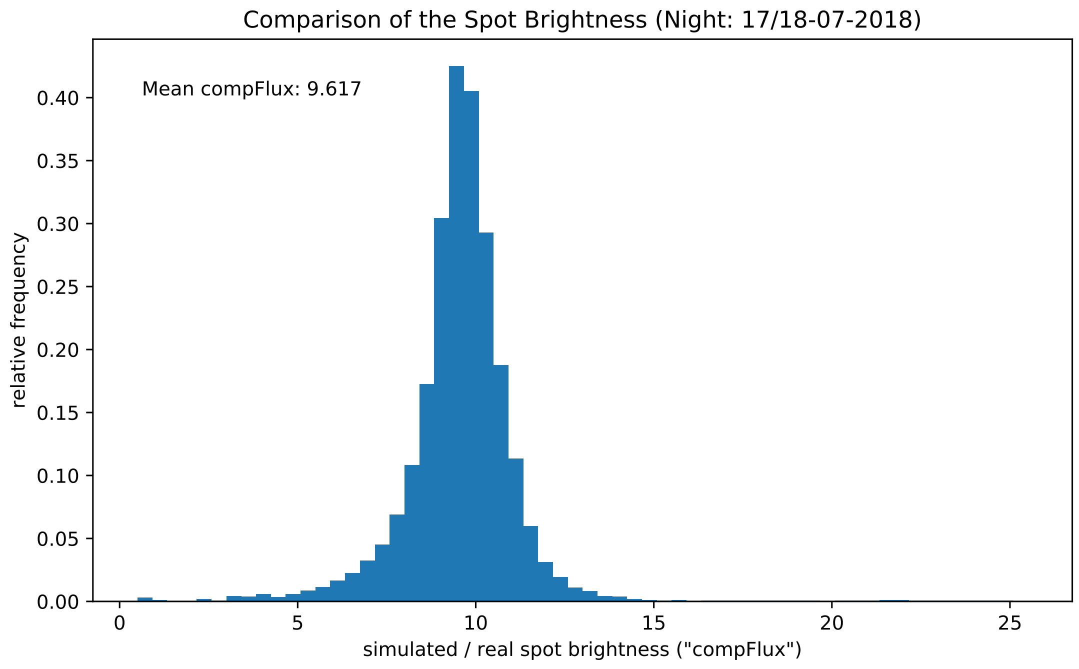

Figure 8: This figure shows a histogram of normalised brightness values of spots. This normalisation is done by dividing the simulated by the real spot's brightness. In contrast to Figure 7, this histogram has only one peak at compFlux = 10. This indicates, that the darker spots were indeed generated by weather conditions like clouds or fog.

main share of spots is ten times brighter than the simulation. The right "tail" of the histogram, containing the smaller peak, leads to the conclusion, that a small part of the stars are darker than others. This could be caused by clouds or fog. To investigate this further, a second set of images were analysed. This set contains only images from one night, hoping for constant weather conditions.

The analysis of the second set is shown in Figure 8. This histogram shows the distribution of the compFlux values, the normalised brightness of spots. In contrast to Figure 7, this distribution has got only one peak at compFlux = 10, whereas there are effectively no stars with compFlux = 20. This indicates, that a part of the random set of images used in Figure 7 was indeed subject to irreproducible errors like fog or clouds. Per definition, these errors cannot be reconstructed by the simulation.

In average, the simulation's spots are 9.62 times brighter than the real image's. In order to correct this, the parameter **efficiency** (**-E**) of **simulate-ccd-image** is used. This parameter describes the photon detection efficiency of the simulated CCD-Camera. However, this option does not affect the brightness of the background. By default, this value is currently $\frac{1}{1.88} = 0.532$. Therefore, the new value used for the option **-E** should be:

$$\text{Efficiency}_{\text{calculated}} = \frac{0.532}{9.617} = 0.0553 \qquad (2)$$

If this value is used, the new mean compFlux value is 0.543. This indicates, that the spot extraction for small flux values does not work properly. Since the spot extraction can only evaluate a confined area, pixels with small brightness values happen to be ignored, causing a distortion for small flux values. Therefore, the mean compFlux was estimated for multiple efficiency values. For this estimation random images of the night 17/18-July-2018 were used. The result can be seen in Table 1. In conclusion, the efficiency value of 8.75 % was implemented into the simulation.

efficiency CCD-Camera	mean compFlux value	number of analysed images
5.75 %	57.3 %	12
5.85 %	58.3 %	10
5.90 %	62.8 %	12
5.95 %	58.7 %	11
8.50 %	94.9 %	12
8.75 %	101.1 %	23
10.00 %	122.0 %	12

Table 1: This table shows simulated efficiency values of the CCD-Camera and their resulting mean compFlux value. In Addition, the last column contains the number of images which were to estimate the mean compFlux for the given efficiency.

In order to verify this implementation for a larger set of images, a histogram of compFlux values of all images in this night was generated. This is shown in Figure 9. As the mean compFlux value is 100.3 %, the new efficiency value can be seen as justified.

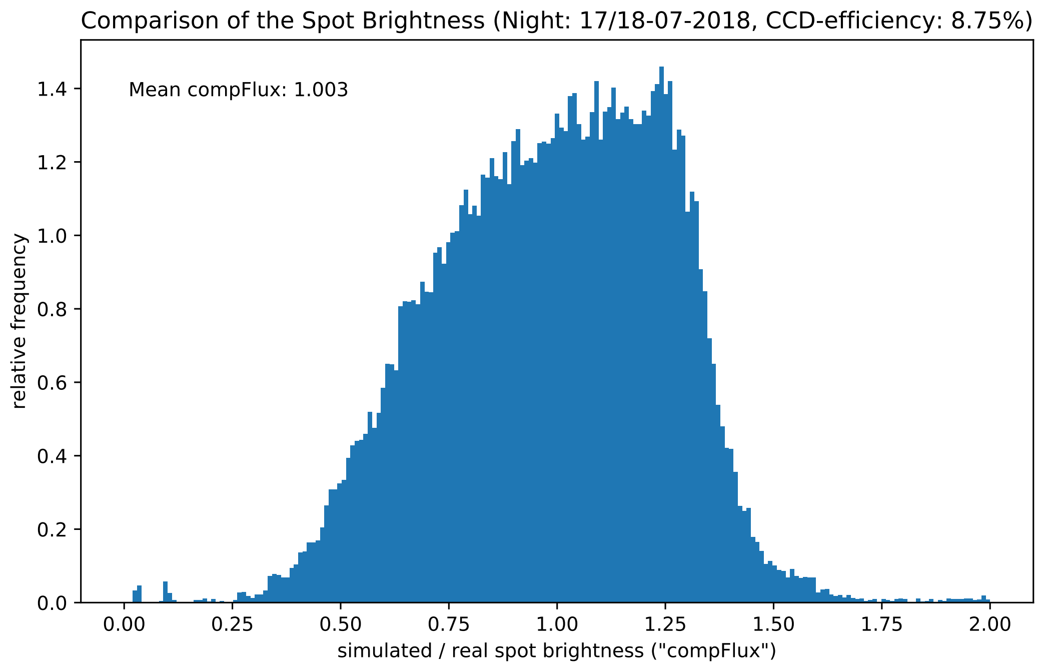

Figure 9: This figure shows a histogram of compFlux values. This data was collected from all images of one night. The mean compFlux value generated by the chosen CCD-efficiency of 8.75 % is 100.3 %, indicating that the chosen value is indeed valid. The histogram resembles a Gaussian distribution.

4 Comparison of the Stars' Position

4.1 First Comparison

Another most important property of a simulated image is, that the positions of the stars coincide in the simulated and the real image. Therefore, a random set of images was collected and analysed using **getStarID**, obtaining the pointing of these images. With this information the images were simulated accordingly using the program **simulate-ccd-image** in **libPointingMST**. Then, the program **getStarID** was again applied, this time on the simulations. As a result, one obtains the **getStarID** files of both the simulations and the real images, which enables the comparison of them.

To simplify this process, a python script "**resimulate**" was created. This script needs as input only the FITS image which should be resimulated. It automatically uses **reconstruct-images** to calculate the pointing of the image and searches the FITS header for other relevant information like the f-number or exposure. This gathered data is then used to replicate the image.

This is shown in Figure 10. This plot compares a real image with its replicated version. All spots identified by the spot extraction algorithm are marked with a cross, real spots are hereby green and simulated ones red. A circle instead of a cross indicates that the according spot could be matched by **libPointingMST**. If a star could be identified in the real as well as in the replicated image, they will be connected by an arrow. This arrow points always towards the simulated star. The CTA-Camera is located in the centre of the image and covers the night sky behind, which is also considered in the simulation. As result, there are no extracted spots in the centre of the image. Figure 10 is hereby characteristic for the whole test set.

This indicates, that the CCD-Camera is rotated through approximately 180° around the optical axis, which is neglected by the simulation. If this image is analysed with **reconstruct-images**, it calculates a rotation angle of 177.7°. The mean rotation of 4050 analysed images was $(176.6 \pm 2.3)°$. Hence it is necessary to implement a rotation around the optical axis into the simulation. This could be accomplished by using the internal coordinate transfer methods and classes. This enables the program **simulate-ccd-image** to take a rotation angle as input and then simulate the rotated image. As a first step it is assumed that the rotation is around the geometrical centre of the image. In addition, the script **resimulate** was extended: By setting the flag "**-r**" it automatically uses the rotation angle calculated by **reconstruct-images** for the simulation.

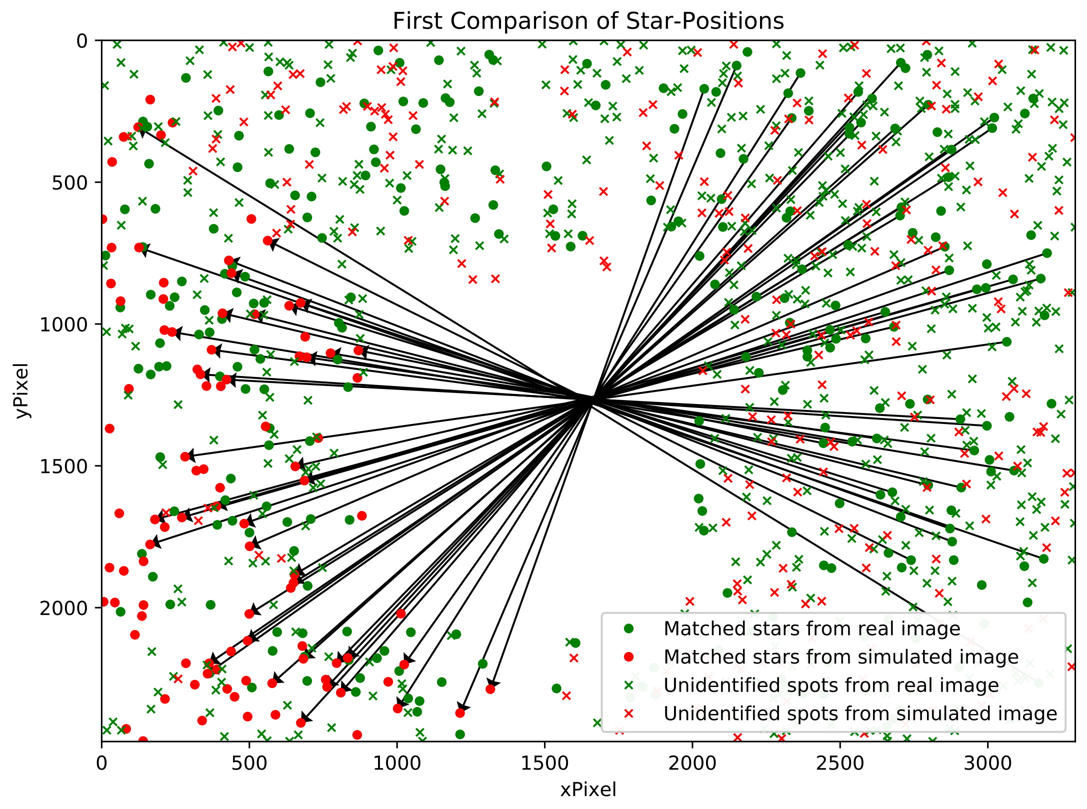

Figure 10: This figure compares a real image (green) with its replicated version (red). All found spots are plotted with a cross at their pixel position, whereas the circle indicates that the spot could be matched. If the same star is identified in the real as well as in the simulated one, they will be connected by an arrow. This arrow points toward the simulated star.

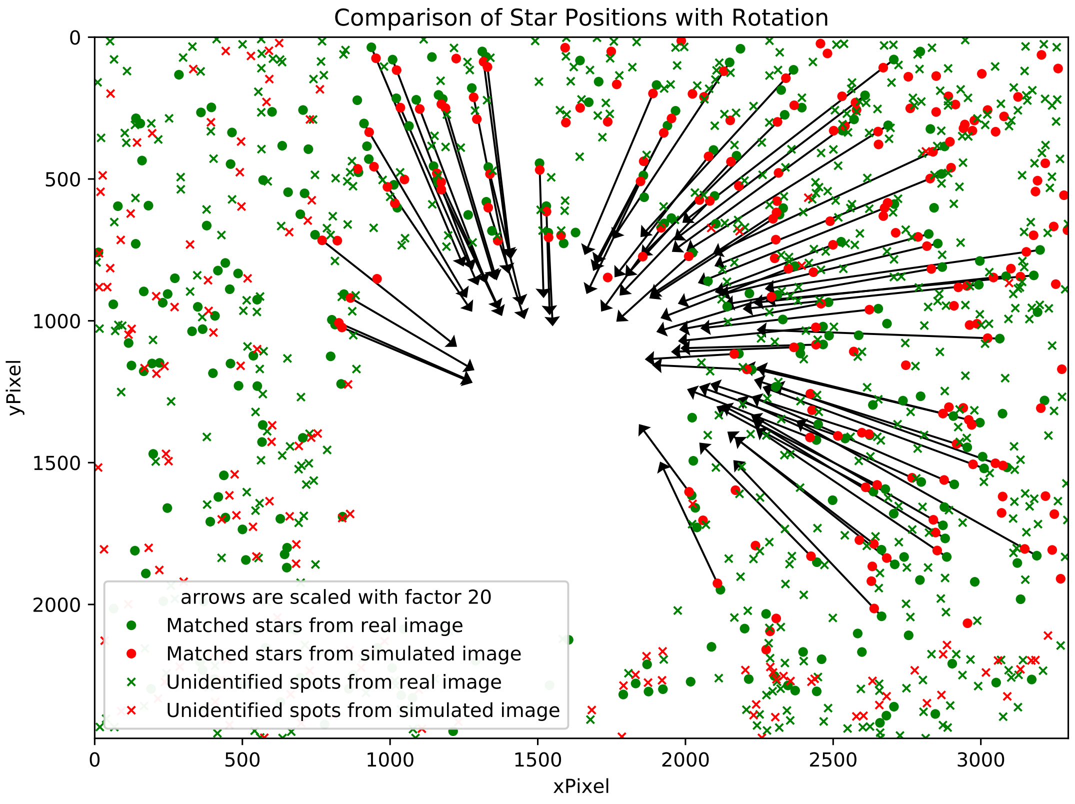

Figure 11: This image shows the same comparison as in Figure 10, but with corrected rotation around the optical axis. Now, all arrows are pointing towards the centre, which indicates a problem with the focal-length parameter of the simulation.

4.2 Analysing the Focal Length

Figure 11 shows the same real image and its replication, if the rotation around the geometrical centre of the image is corrected with the value calculated by **reconstruct-images**. In this plot, all arrows are scaled with the factor 20. Now the arrows point towards the centre, meaning that the simulated stars are positioned too centrally in the image. This leads to the conclusion that the simulation's field of view is oversized, compared to the real image.

Looking at the structure of a CCD-Camera (Figure 12), the angle of view α can be calculated as following:

$$\alpha = 2 \cdot \arctan\left(\frac{2}{2S_2}\right) \qquad (3)$$

Whereas S_2 is the distance between camera lens and CCD-Chip and d represents the

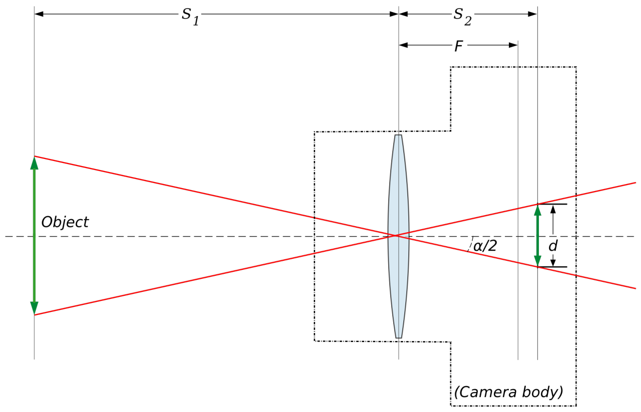

Figure 12: This figure shows the structure of a camera. S_2 is the distance between camera lens and CCD-Chip, d the height of the CCD-Chip and α the angle of view. If the image is in focus, S_2 is equal to the focal length.
"Lens angle of view“ by Moxfyre at English Wikipedia / CC BY-SA 3.0

physical dimensions of the CCD-Chip. If the image is in focus, S_2 is equal to the focal length F. Equation 3 leads to the conclusion, that the focal length used by the simulation is too small. Currently, the simulation uses a value of $F = 50\,\text{mm}$, as specified by the manufacturer of the applied lens.

In order to check this value, the program **reconstruct-images** can be used. This program calculates among others the focal-length of an image. Using a file containing the results of **reconstruct-images** of 4050 successfully matched images, the mean calculated focal length was computed.

$$F_{\text{calculated}} = 51.479\,\text{mm} \pm 6\,\mu\text{m} = 51.479\,\text{mm} \pm 0.011\,\% \tag{4}$$

This very small deviation leads to the conclusion, that either the provided focal length is false or that the CCD-Chip is slightly out of focus. In either case, this systematic error can be corrected by using the calculated focal length to simulate images, especially since the variance of F is small. Therefore the **resimulation** script was adapted: By setting the flag "**-f**", the simulation uses the calculated focal length of the original image for the simulation.

4.3 Analysing the residual rotation

Subsection 4.2 revealed that the simulation can be improved by using the calculated value for the focal length. Figure 13 shows the comparison between a real and such a simulated image. Since there are no arrows pointing towards (or away from) the centre, it shows that the simulation's angle of view is now far more realistic.

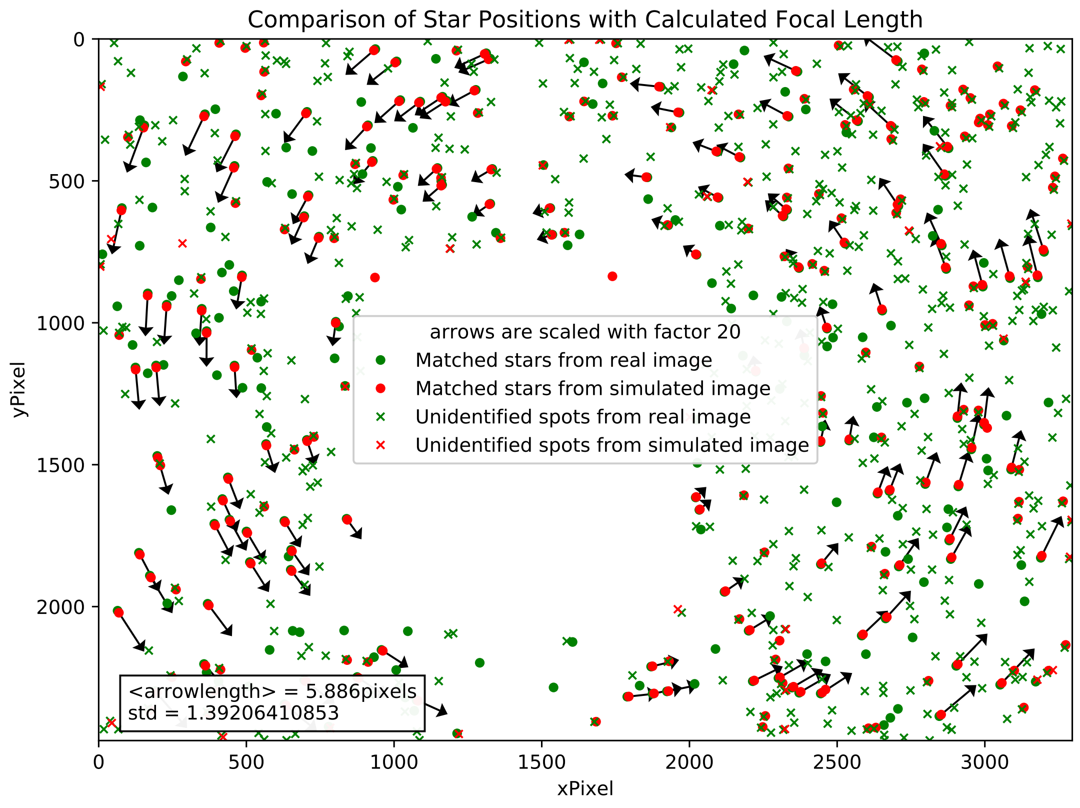

Figure 13: This figure shows the comparison between a real and a replicated image simulated with the calculated focal length. The angle of view of the simulated image is now far more realistic. The residual error indicates an additional unknown rotation.

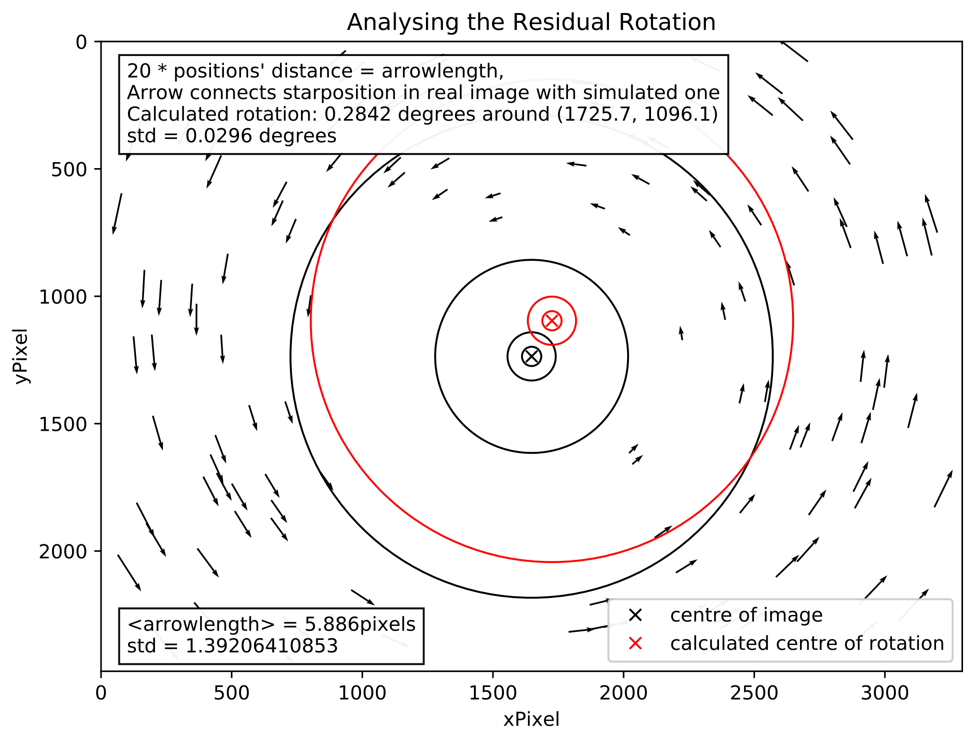

Figure 14: This figure shows the residual rotation of the simulated image. The arrow length is scaled with a factor of 20. The black cross/circles indicate the geometrical centre of the image, whereas the red one is located at the calculated rotation point. The positions of them do not coincide.

The residual arrows indicate a by now unknown rotation. To improve the understanding of this phenomenon, the centre of the rotation was reconstructed. This is shown in Figure 14. Figure 14 shows again the deviation of a stars' pixel position in the real and the simulated image. The arrows are scaled with a factor of 20. To calculate the centre of the image, a payoff function f was introduced, which calculated a payoff value for a given rotation centre. A small value indicates, that the arrows can be described by a rotation around the given centre point. The function f understands an arrow as a chord of a circle around the centre point. This circle goes through the base point and the arrowhead. Then an auxiliary line is calculated, which is rectangular to the tangent of the circle at the base point of the arrow. The payoff function sums up the distance from all this auxiliary lines to the centre point and returns this value as payoff value. This payoff function is then minimised. As a result, it is visible that the constructed rotation centre does not coincide with the geometrical one.

4.4 Considering a Shifted Rotation Point

As Figure 13 shows, there remains a residual rotation of the stars' positions. This could be caused by approximating the rotation centre to the geometrical centre of the image. If this is the true, there would be two possibilities to decrease the residual deviation:

1. Fixing the rotation point in the reconstruction to the geometrical centre of the image. This generates a deviating value of the calculated rotation angle of a real image. This deviating value should then improve the description of the star positions.
2. Using the calculated position of the rotation centre. Since this method does not use an approximation, the results should be better than those of method one. However, it would be beneficial to assume a constant position of the shifted rotation centre, in order to simulate images without a real source image. As of yet, it's unclear if this assumption is valid.

Since it is not clear to which extend the residual rotation is caused by the used approximation, it was decided to firstly pursue method 2. This method should theoretically cause the best results. Afterwards, it can be determined, if the residual deviations are indeed caused by approximating the rotation point to the geometrical centre of the image.

In order to implement this, the option **-p <xPixelPosition> <yPixelPosition>** was added to the program **simulate-ccd-image**. If this option is set, the simulated image will be parallel shifted, so that the rotation centre will be at the given pixel coordinates. This coordinates can be determined by **reconstruct-image**. The resimulation of an image with a shifted rotation centre can be done automatically by using **resimulate** with the option **-p**.

The program **reconstruct-image** calculates two pointing directions: The pointing and the centre direction. Until now, the centre direction was used. This direction points toward the geometrical centre of an image. Since the centre of the simulated image (with option **-p**) is shifted towards the pixel position of the given rotation point, the centre pointing direction of the simulated image would now be incorrect. To avoid this, the option **-p** of **simulate-ccd-image** should only be used with the pointing direction. This direction points toward the rotation centre of an image, because the optical axis of the telescope is directed towards this position. This leads to a correct pointing of the simulated image.

Figure 15 shows the residual deviations after simulating the image with the shifted rotation point (option **-p**). Firstly, it's evident that the mean arrow length decreased from 5.89 to 0.96. This indicates, that approximating the rotation point to the geometrical centre was indeed the main problem. However, it is surprising that the stars on the right side are more accurately simulated than their left counterparts. At this point, there are three possible explanations:

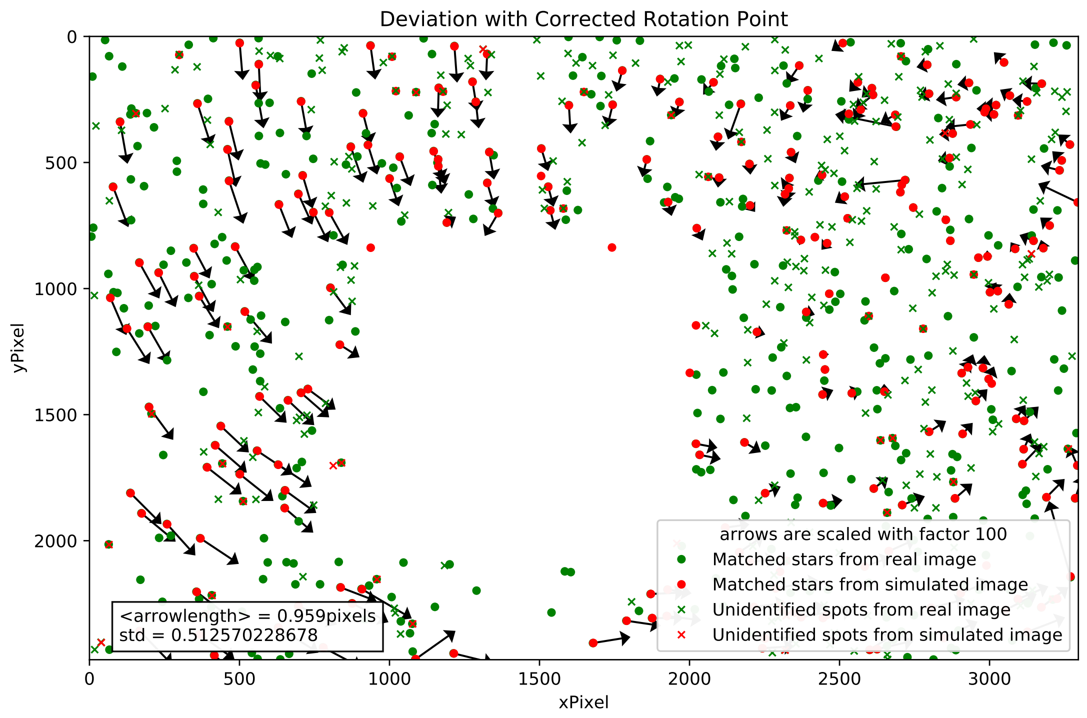

Figure 15: This figure shows the deviations between the position of stars in the real and the simulated image, whereas the actual rotation point (option **-p**) rather than the geometrical approximation was used. This measure decreased the mean residual arrow length from 5.89 to 0.96 pixels.

1. There is a problem with the matching algorithm: That means, that mainly (or solely) stars from the right side of the image are used to calculate the pointing of the real image, which causes the left side to be less accurate.
2. There could be a problem with the time stamp of the real image, which is used to simulate it's replicate. If the simulation uses a faulty time, it's possible that the pixel positions of the stars are shifted by the earths rotation.
3. There could be a problem with the simulation itself: It is possible that the simulation is not able to model the reality correctly.

4.5 Comparing the Simulation with the Calculated Stars' Positions

In order to remove unconsidered errors of the real image, it is helpful to transform the star's catalogue' position into the expected pixel position in the image, using the determined pointing. This functionality was added to the program **getStarID**, which writes this calculated pixel position into the StarID.txt file. This catalogue position (calculated with the pointing of the real image) is then compared with the simulated spot's position. This is shown in Figure 16. The arrows point towards the simulated pixel position.

Since the comparison of the catalogue positions with the real star positions shows approximately the same deviations as previously observed, it can be ruled out that possibility 1 is the source of this problem. In this case, the arrows in Figure 16 would vanish, since both positions are using the same pointing.

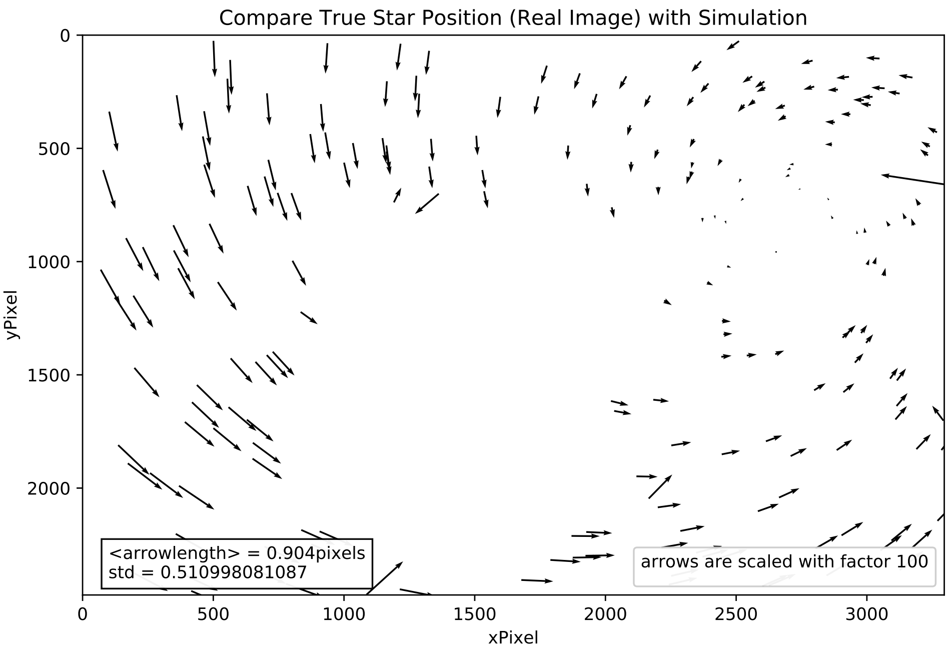

Figure 16: This figure shows the comparison between the star catalogue pixel positions calculated with the pointing of the real image and the pixel positions of the simulated stars. The arrows point towards the simulation's positions. This figure shows roughly the same deviations as Figure 15. Since both positions use the same pointing, any error caused by possibility 1 would disappear. Therefore this possible explanation can be ruled out.

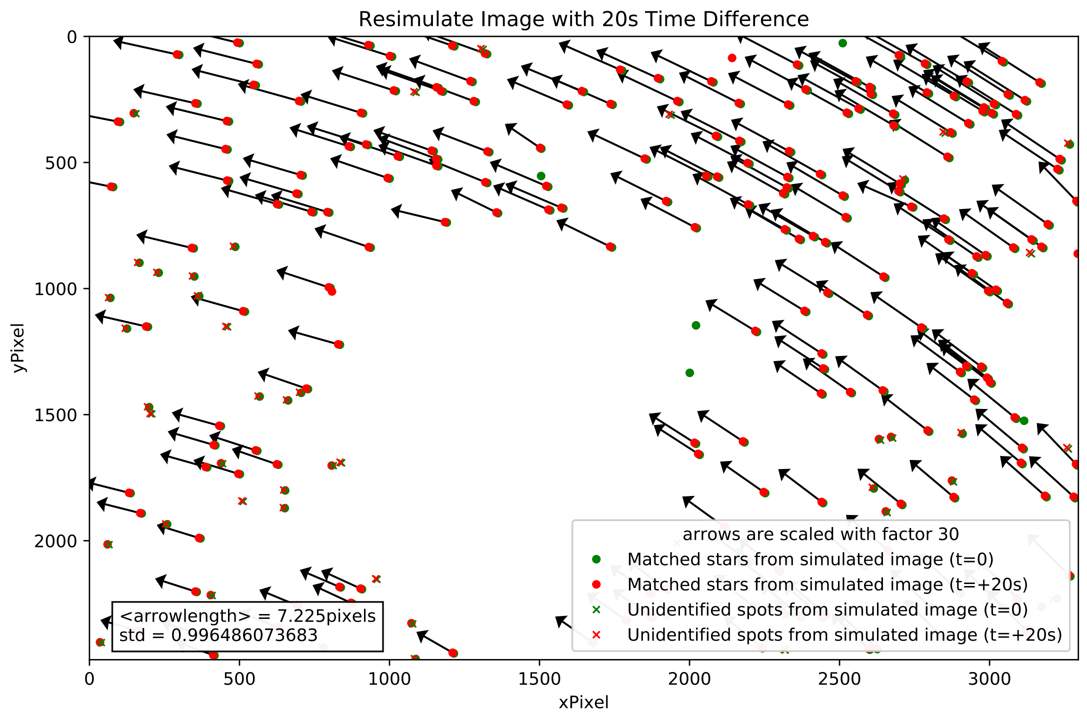

Figure 17: This image shows the comparison between two simulated images, both with the same pointing. The sole difference is, that the second image (Red) simulates the sky 20 s later than the first one (Green). It can be seen that the second image is, compared to the first one, slightly rotated around a rotation point located outside of the image.

4.6 Considering False Time Stamps

4.6.1 Understanding the Consequences of a False Time Stamp

The above mentioned second possibility is, that the time stamp of the real image is flawed. In order to understand the consequences of this flaw, it is helpful to look at the deviations caused by such a time discrepancy. This is shown in Figure 17. This figure shows the deviations caused by a time discrepancy of 20 s, where as the arrows are pointing towards the subsequent image. Both images are simulated.

As Figure 17 shows, the deviations caused by a time shift of 20 s can be described as a rotation with a rotation point located outside of the image. As a first approximation, it can be described as a simple shift. If an algorithm tries to calculate the pointing of an image with such a flawed time stamp, it will try to minimise the residual deviations. This can be done by using this approximation.

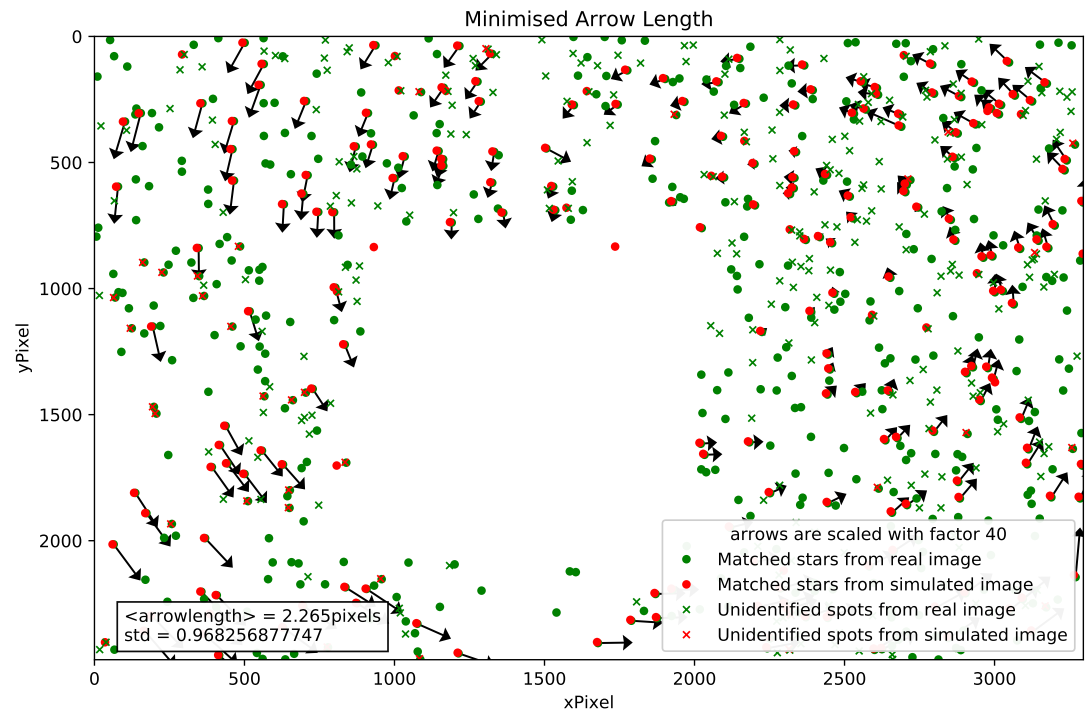

Figure 18: This figure shows the expected behaviour when analysing an image with a faulty time stamp. (Here: Figure 17.) The resulting deviation approximately resembles the current deviations of the simulation in comparison to real images (Figure 15).

In order to model this behaviour, the shift vector was determined by minimising the residual deviations of the star positions. The mean residual arrow length could be reduced by this procedure from 7.225 to 2.265 pixels. The result of this shifting is shown in Figure 18. It is noticeable, that the resulting arrows approximately resemble the current deviations between simulation and real image, as shown in Figure 15. This shows that faulty time declarations are able to cause the observed deviations.

4.6.2 Resimulation of Simulated Images

Since Section 4.6.1 could show that faulty time stamps can indeed cause deviations like displayed in Figure 15, it's reasonable to analyse this topic further. However, there is no possibility to verify the logged time of the real images. Therefore it is necessary to work only with simulated ones, as they provide correct time stamps.

We therefore consider a simulated image A and the resimulation of A named B. Since the simulation does not show irreproducible effects, A and B should be very similar.

Figure 19 shows the deviations of these images. However, these deviations are roughly the same as previously observed. This leads to the conclusion, that FITS-headers containing faulty time stamps are not the primary error causing the observed residual deviations.

This directs to the above mentioned third possibility, that the spot simulation cannot describe the reality properly. From this it follows that the effects of this possibility should vanish, when two simulations are compared. Since Figure 19 shows that the comparison between to simulated images is still showing the observed deviations, it can be ruled out that this effect causes these deviations.

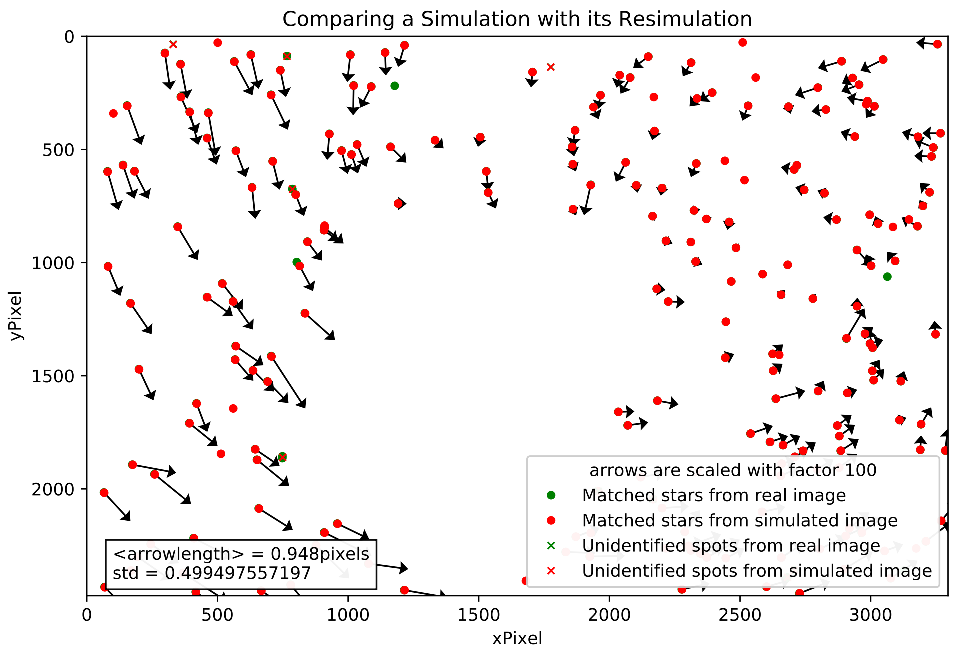

Figure 19: This figure shows the deviations between a simulated image and it's resimulation. It is visible, that this figure shows approximately the same deviations as observed in subsection 4.4. This indicates, that a faulty time stamp is not the primary cause of the residual deviations.

4.6.3 Analysing the spot simulation

In order to investigate this phenomenon further, it's helpful to eliminate the effects of varying calculated pointing directions. This can be achieved by comparing the transformed star catalogue positions of A with the extracted spot position of B. As both are calculated using the same pointing, they should coincide.

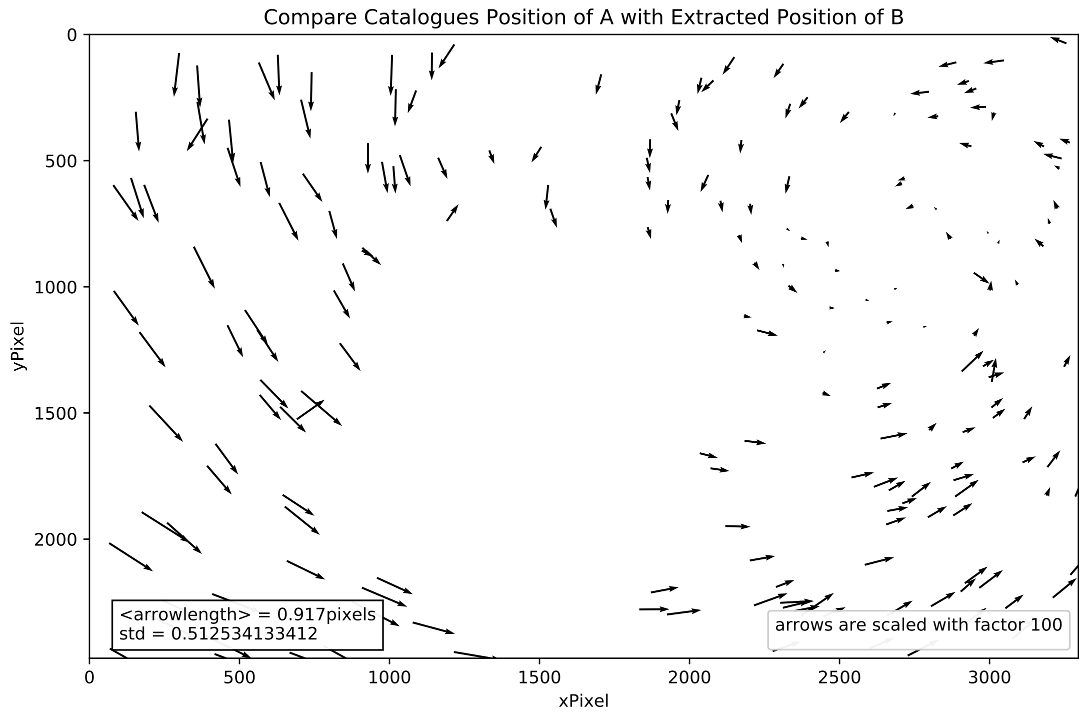

Figure 20: This figure shows a comparison between the star catalogues positions of A transformed to pixel coordinates and the extracted spot positions of B. Since the observed deviations are unchanged, a different calculated pointing direction cannot be the reason of the deviations.

This is shown in Figure 20. One can see, that the resulting deviations are equal to the previous observed ones. This indicates, that different calculated pointing directions aren't the source of the deviations.

4.7 Reduction of the Simulation's Complexity by Setting the Exposure near Zero

As all previous attempts to explain the residual deviations have failed, it is reasonable to decrease the complexity of the simulation. Since Section 4.6.1 showed that a false timing can cause comparable deviations, it was decided to examine the exposure of the simulation further. As an example, the images showed above had an exposure of 10 s. The simulation handles a duration of exposure by splitting the time frame into many time steps, at which static images are being simulated. These static images are combined afterwards. Therefore, it is useful to analyse such a single static image.

This was implemented by setting the duration of exposure to 1 ms. As the brightness depends on the exposure, this decreased the brightness of the simulated spots. This was compensated by multiplying the simulated efficiency of the CCD-Camera by the factor 1000. In Addition, the spot extraction's quality check was deactivated, since it would filter spots which are too small. As a consequence, the simulated background was removed, in order to prevent false extracted spots. As a result, the simulation generates images without blurring. This should improve the accuracy of the spot extraction.

Comparing a Static Simulation with its Static Resimulation

As a first step, it was decided to compare a simulated static image with it's static resimulation, recreating the situation discussed in Section 4.6.2. This can be seen in Figure 21. It is visible, that the observed residual deviations are now different. Whereas Section 4.6.2 observed approximately a rotation, the static images are showing a rather constant shift of 0.67 pixels.

One possible explanation is, that this shift is caused by an inaccuracy of the calculated pointing. This can be tested by recreating the comparison of stars' catalogue positions transformed to pixel coordinates with the extracted positions of their resimulated spots, as described in subsection 4.5. As both positions are calculated using the same pointing, varying pointings can not have any effect in this comparison.

Figure 22 shows this comparison. It can be seen that most of the arrows are still constantly shifted, comparable to Figure 21. The longer arrows could be caused by mismatched stars. As the residual deviations of the previous Figure are still present, it can be excluded that they are caused by faulty pointings.

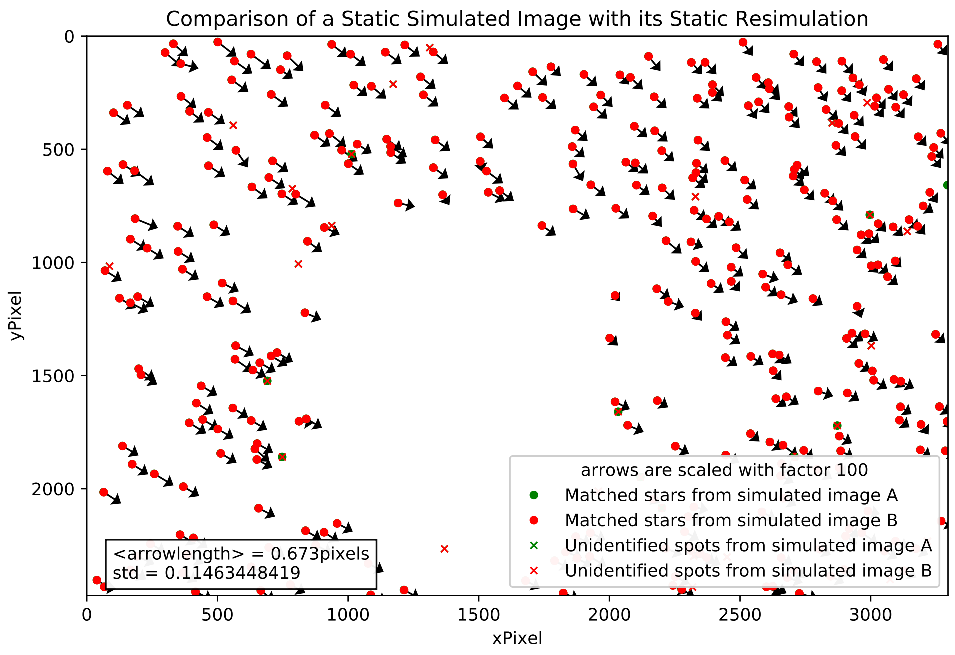

Figure 21: This figure shows a comparison of a static simulated image with its static resimulation. Static means, that the exposure was set to 1 ms, whereas the decreased spot brightness is compensated by increasing the simulated efficiency of the CCD-Camera. It can be seen that this Figure shows a constant shift rather than the previous observed rotation.

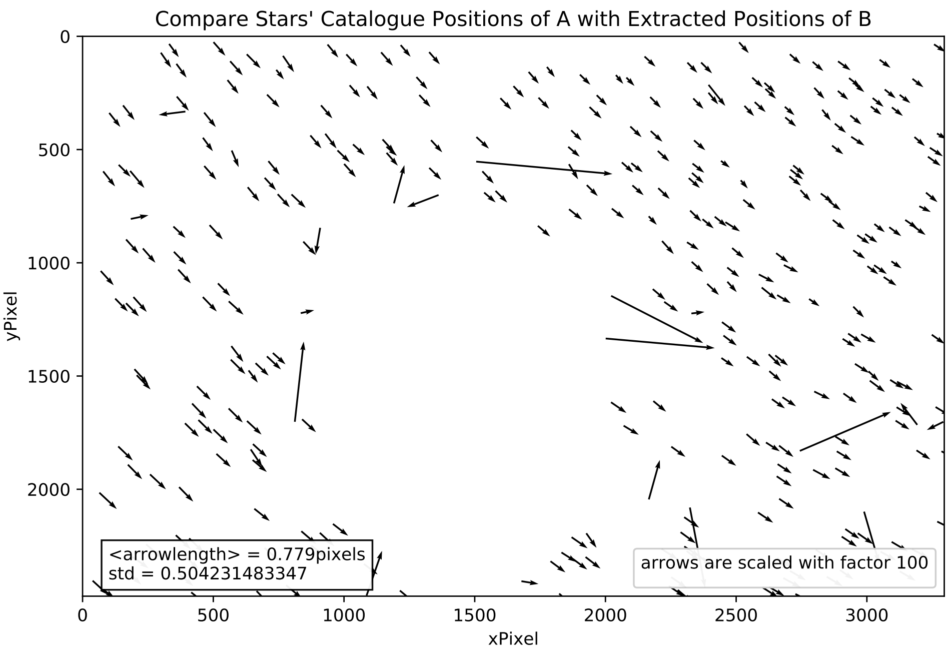

Figure 22: This figure shows the comparison of the transformed stars' catalogue positions of A with the extracted positions of its resimulation B. It can be seen that most of the deviations are a constant shift, whereas the longer arrows could be caused by mismatches. It can be excluded that the shift is caused by inaccuracies of the estimated pointing, as both positions were calculated using the same pointing.

4.8 Possible Reasons for the Observed Residual Deviations

4.8.1 Explanations for the Constant Shift of Static Images

Since the previous paragraph showed that imprecise pointings are not the cause for the residual shift, it's necessary to research this topic further.

One possible explanation for this effect could be a problem with the spot extraction. Since all spots are approximately shifted equally, it is possible that the spot extraction uses, compared to the simulation, a shifted pixel coordinate system. One likely example is, that both coordinate axis are shifted by 0.5 pixels. This is a common operation, used to define the location of a pixel as the centre rather than the pixel's corner. This would result in a diagonal shift with an arrow length of $\sqrt{0.5^2 + 0.5^2} = \sqrt{0.5} \simeq 0.707$ pixels. This matches the observed deviations, which is a diagonal shift with a mean arrow length of 0.673 pixels.

4.8.2 Explanations for the Rotated Deviations of Dynamic Images

As a problem with the spot extraction would also affect the images with normal exposure (here: 10 s, "dynamic images"), it is reasonable to attempt to decouple this effect. Therefore one pointing was used to simulate two images A_1 and A_2, whereas A_1 is static and A_2 is dynamic. The time used to simulated A_2 was shifted, so that A_1 is in the middle of the exposure. This comparison is shown in Figure 23. This Figure shows a rotation with a small angle around a rotation point located outside of the image. This resembles Figure 17, which displayed the difference of spots position after a time difference of 20 s.

On the one hand, this could indicate a problem with the time handling of the simulation. One problem of this explanation is, that the rotation centre of this image is in the direction of the lower left, whereas this of Figure 17 is in the upper right direction. And that regardless of the fact, that both images are resimulations of the same real image. On the other hand, this can also be ascribed to a problem of the spot extraction. In theory, a star which is photographed over a given time period, should be depicted as a "banana shape". It is possible that the extraction algorithm cannot handle this shape correctly and returns a distorted spot position.

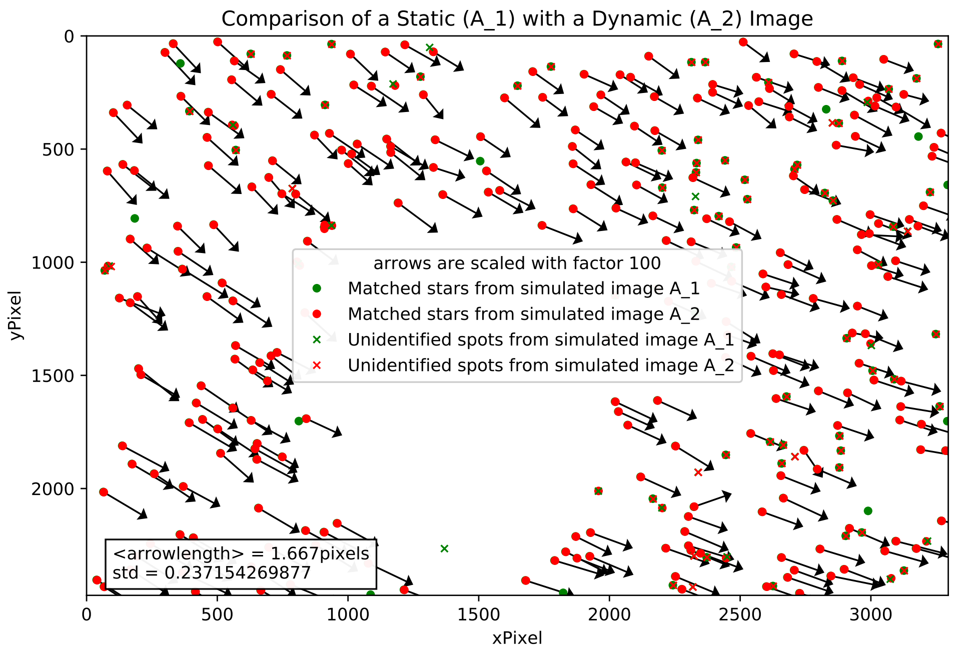

Figure 23: This figure shows a comparison of static A_1 and a dynamic image A_2 generated with the same pointing. The time of A_2 was adusted in this way, that the middle of the exposure time frame is the time of A_2. This figure shows a rotation (small angle) around a rotation point located outside of the image.

5 Summary and Outlook

This thesis investigates the quality of the single-CCD pointing image simulation based on LibCCDCamera and LibPointingMST. In order to get suitable real images, a CCD-Camera mounted on the MAGIC Telescope in La Palma was used. At First, a program was introduced which was able to match the spots of an image to real stars. This was done by introducing a distinct StarID. With this data, a first comparison of a real image with its simulation was possible.
This comparison concentrated on two rather independent topics: The brightness of spots and background and the correct pixel positions of the stars. At first, the background of the simulation was analysed. It was found out that the simulation overestimates the background, which could be compensated by implementing a constant, which handles the background at the telescope's site correctly. Secondly, the brightness of the simulated spots was compared with their real equivalents. It was revealed, that the simulation's spot are 9.61 times too bright. The spot brightness can be controlled by the efficiency of the simulated CCD-Camera. However, the attempt to correct this value in a linear way (as theoretically expected) failed. This was ascribed to problems of estimating the brightness of dark stars. Therefore a suitable value was estimated by trial and error, which lead to the new photon efficiency of 8.75 %. This generates simulated spots which are 0.3 % brighter than their real counterparts. Therefore, this adjustment can be regarded as successful.

The other topic was the correct positioning of the simulated spots in the image. A comparison of real images with their simulations found out, that real images are rotated by approximately 177°. This could be compensated by introducing a rotation angle to the simulation. Initially the real rotation centre was approximated to the geometrical centre of the image. The rotation angle was provided by **reconstruct-images**. As the next step, the comparison revealed that using the focal length as specified by the manufacturer of the lens is not sufficient, since the simulated images were not correctly scaled. This made it necessary to use calculated values for the focal length, which was also provided by **reconstruct-images**. As a result, a new comparison of real and simulated images showed, that a residual rotation can be observed.

This residual rotation was affiliated to approximating the rotation centre to the geometric centre of the image. Therefore, the program **reconstruct-images** was adapted, so that it also outputs the position of this rotation point. Furthermore, the simulation (**simulate-ccd-image**) was extended, enabling it to simulate a given arbitrary rotation centre. This features led to another smaller residual rotation. By now, the cause of this residual deviation is unknown. Following possible explanations were researched:

1. A false pointing is causing these effects. This could be excluded by comparing the star catalogue position (transformed to pixel coordinates) with the extracted position of its simulated counterpart. This comparison showed also the residual rotation. Since both the transformation and the simulation used the same calculated pointing, the explanation "false pointing" could be excluded.
2. The pointing cannot model the sky correctly. Since a comparison of the simulation A and its resimulation B also showed the questionable deviations, this explanation could be excluded.
3. Since the deviations are also visible when comparing two simulations, it can be excluded that a faulty time declaration is causing these effects. However, it was showed, that resimulating an image rotated with a small angle around a rotation point located outside of the image (resembling a time inaccuracy) could indeed cause deviations resembling the observed ones.

In Addition, it was found that comparing static simulated images reduces the residual rotation to a constant shift. A static simulated image is an image with an exposure of 1 ms, whereas a dynamic image has typically an exposure of 10 s. The decreased spot brightness is compensated by a higher simulated CCD-efficiency. Currently, the most likely assumption is, that this constant shift is caused by the spot extraction. The observed shift resembles a 0.5 pixel shift in x- as well as in y-direction. This is done when exchanging the nominal pixel position between its corner and centre.

The conclusion is, that the observed residual deviations must be generally further researched. The most promising step is to begin with a check of the spot extraction. Furthermore, the reason for the different deviations of static and dynamic images must be investigated.

6 Appendix: Useful scripts and programs

In order to write this thesis, some scripts/programs where developed/adapted. The two most important ones are **resimulate** and **getStarID**:

resimulate: To simplify the simulation of a real image, the python script "resimulate" was created. It uses a given CCD-image as input. It automatically collects relevant information from the FITS-Header and from **reconstruct-images** in order to start the existing script **simulate-ccd-image**. In Addition, it can be specified if the calculated value of focal-length, rotation angle and rotation point should be used.

getStarID: This program saves information for each spot:

StarID	This is distinct ID, used to identify a star over several images
x/yPixel	This is the extracted pixel position of a star
skyMag	The magnitude of the star [mag]
flux	Extracted brightness for spot
recoAz/Alt	Reconstructed AzAlt position of spot
Az/Alt	Star catalogue position of a star transformed into AzAlt
Ra/Dec	Star catalogue position of a star transformed into RADec
Star(X/Y)Pixel	Star catalogue position of a star transformed into pixel coordinates
backgroundValue	The brightness of the fitted background evaluated at the spots pixel coordinates

7 Index of abbreviations

CTA	Cherenkov Telescope Array
FITS	Flexible Image Transport System
H.E.S.S.	High Energy Stereoscopic System
IACT	Imaging Atmospheric Cherenkov Technique
LST	Large-Sized Telescope
MAGIC	Major Atmospheric Gamma-Ray Imaging Cherenkov Telescopes
MST	Medium-Sized Telescope
SST	Small-Sized Telescope
VHE	Very High Energy

8 References

[1] CTA-Website. `https://www.cta-observatory.org/`. Online, accessed 14-May-2019.

[2] Markus Gaug, David Berge, Michael Daniel, Michele Doro, Andreas Förster, Werner Hofmann, Maria Conetta Maccarone, Dan Parsons, Raquel de los Reyes Lopez, and Cristopher van Eldik. *Calibration strategies for the Cherenkov Telescope Array*, volume 9149. 2014.

[3] S. Gillessen and H.E.S.S. Collaboration. Arcsecond Level Pointing of the H.E.S.S. Telescopes. *International Cosmic Ray Conference*, 5:2899, July 2003.

[4] Dirk Lennarz. *A Study of Transient Very-High-Energy Gamma-Ray Emission from Gamma-Ray Bursts and Supernovae with H.E.S.S.* PhD thesis, Ruperto-Carola University of Heidelberg, 2012.

[5] Domenico Tiziani. Investigations towards a single-ccd pointing-solution for the medium-sized telescopes of the cherenkov telescope array. Master's thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, 2015.

Danksagung

An dieser Stelle möchte ich mich bei Allen bedanken, die mich bei dieser Bachelorarbeit unterstützt haben. Insbesondere sind zu nennen:

- Prof. Dr. Christopher van Eldik für die Vergabe des Themas und für die zahlreichen Diskussionen, unter anderem in den wöchentlichen Meetings
- Prof. Dr. Stefan Funk für die Übernahme des Zweitgutachtens für diese Bachelorarbeit
- Die gesamte Arbeitsgruppe für die allgemeine Hilfsbereitschaft und für die gute Arbeitsatmosphäre
- Die Teilnehmenden der montaglichen Pointing-Meetings für die vielen Ideen
- Domenico Tiziani für die Beantwortung meiner zahlreichen Fragen und für das Korrekturlesen
- Ronja für die moralische Unterstützung und für das Korrekturlesen

Der größte Dank gilt meinen Eltern.

Erklärung

Hiermit bestätige ich, dass ich diese Arbeit selbstständig und nur unter Verwendung der angegebenen Hilfsmittel angefertigt habe.

Erlangen, den 21. Juni 2019

Marco Egelkraut