Jennifer Studer
studerje@student.ethz.ch

To be determined

Master Thesis

Exoplanets and Habitability
Institute of Particle Physics and Astrophysics, D-PHYS
Swiss Federal Institute of Technology (ETH) Zurich

Supervision

Hans Martin Schmid
Christian Tschudi

December 16, 2021

Abstract

Contents

1 Introduction **3**

2 Aperture Photometry **3**
2.1 Aperture photometry and ghosts 4

3 Transformation to the r-φ plane **7**

4 Acknowledgments **13**

A **14**

1 Introduction

2 Aperture Photometry

In order to determine the flux of different objects in astrophysics, like stars and planets, aperture photometry can be used. This method sums the counts of the pixels inside a certain aperture around the star. In our case this aperture is usually a circle. In order to account for the background noise an annulus around the aperture is taken and the mean of the summed up pixels inside the annulus is subtracted from the apertures pixels. The flux of this aperture is then given by [2]

$$F_{ap} = F_{tot} - n_{px}\langle F_{bg}\rangle, \tag{1}$$

where F_{tot} is the total flux inside the aperture (sum up the pixel values inside the aperture), n_{px} is the number of pixels inside the aperture and $\langle F_{bg}\rangle$ is the mean background per pixel. This mean background per pixel is defined through the annulus and calculated from

$$\langle F_{bg}\rangle = \frac{1}{m}\sum_{i=1}^{m} c_i, \tag{2}$$

where m is the number of pixels in the annulus and c_i the respective pixel value. Figure 1 shows an example for a possible aperture and an annulus around a star, which can be used to do an aperture photometry.

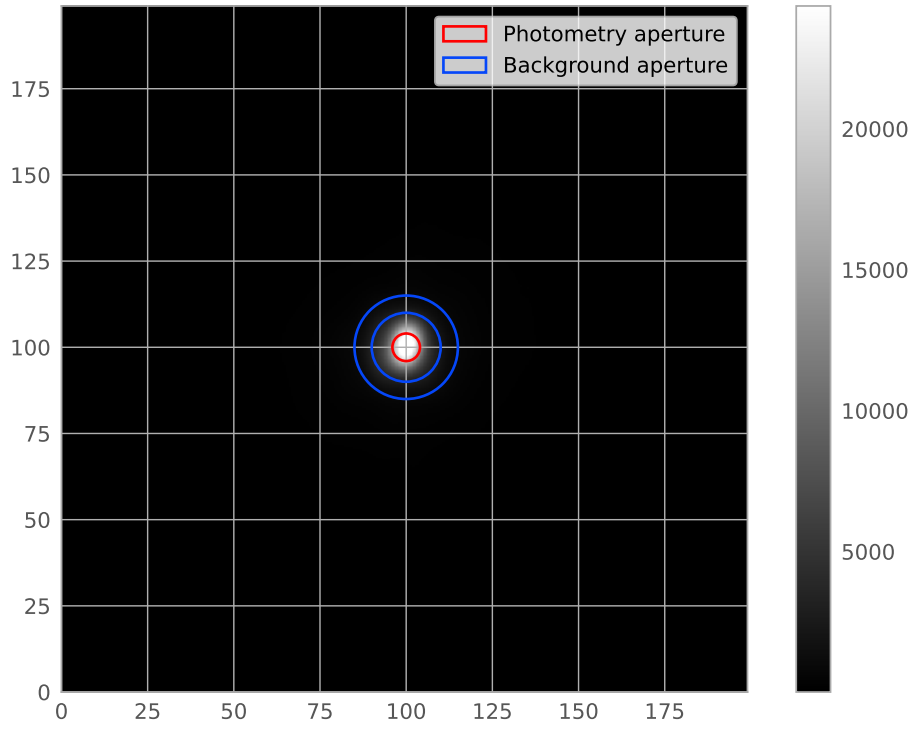


Figure 1: An aperture photometry for the star in the center, where the red circle indicates the aperture used and the two blue circles define the annulus used for the background subtraction.

From the figure you can see that we chose the annulus not directly after the aperture, but this is just one way to do it. One could also choose the annulus directly after the aperture or choose a different distance between the annulus and the aperture. However the annulus should give a good

approximation for the background inside the aperture and therefore it should not be too far away from the aperture. We chose a small distance between the aperture and the annulus of 4 pixels, because we wanted that as little starlight (in this case) as possible is included in the annulus. If we plot the counts per pixels which are included in a certain radius around the star, as it is done in figure 2 we see that after around 10 pixels the increase is decreasing rapidly. That is where there is only little starlight left. This is also why we choose the annulus to go from a radius of 10 to 15 pixels.

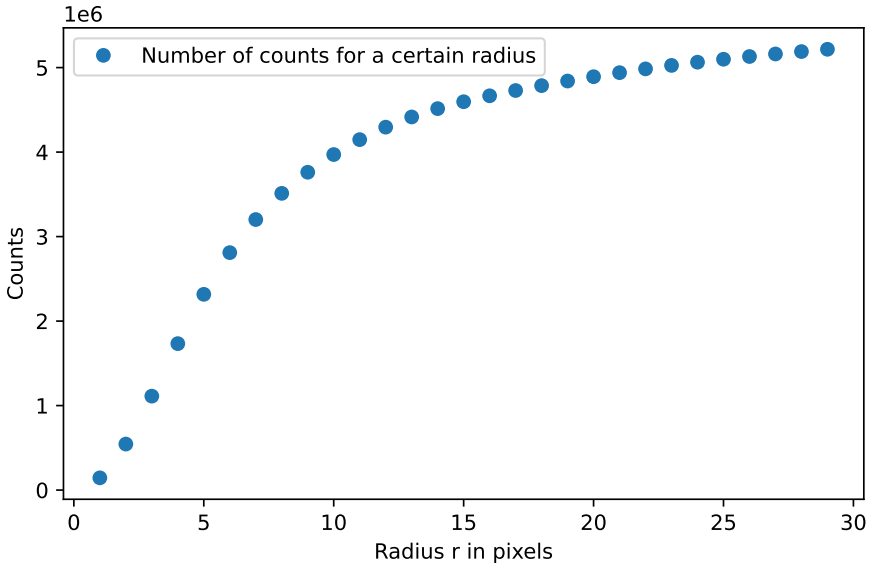


Figure 2: The total flux of the star is calculated for different radii and plotted. This shows that after a radius of 10 pixels the contribution from the star is almost gone.

Additionally we choose the radius of the aperture to be 6 pixels.

2.1 Aperture photometry and ghosts

In order to detect exoplanets one can use aperture photometry. In this subsection we are going to do these steps for the two ghosts which we have in our data from the circumstellar disk HD142527, in order to demonstrate how this could be done for an exoplanet and to learn more about ghosts. A ghost is a copy of the star, which is created by the back-reflection of the star on optical components of the telescope. Figure 3 is an image of HD142527, where the two ghosts are indicated. We see that the ghost on the top right (we will call this ghost 1) is brighter than the ghost on the bottom left (ghost 2).

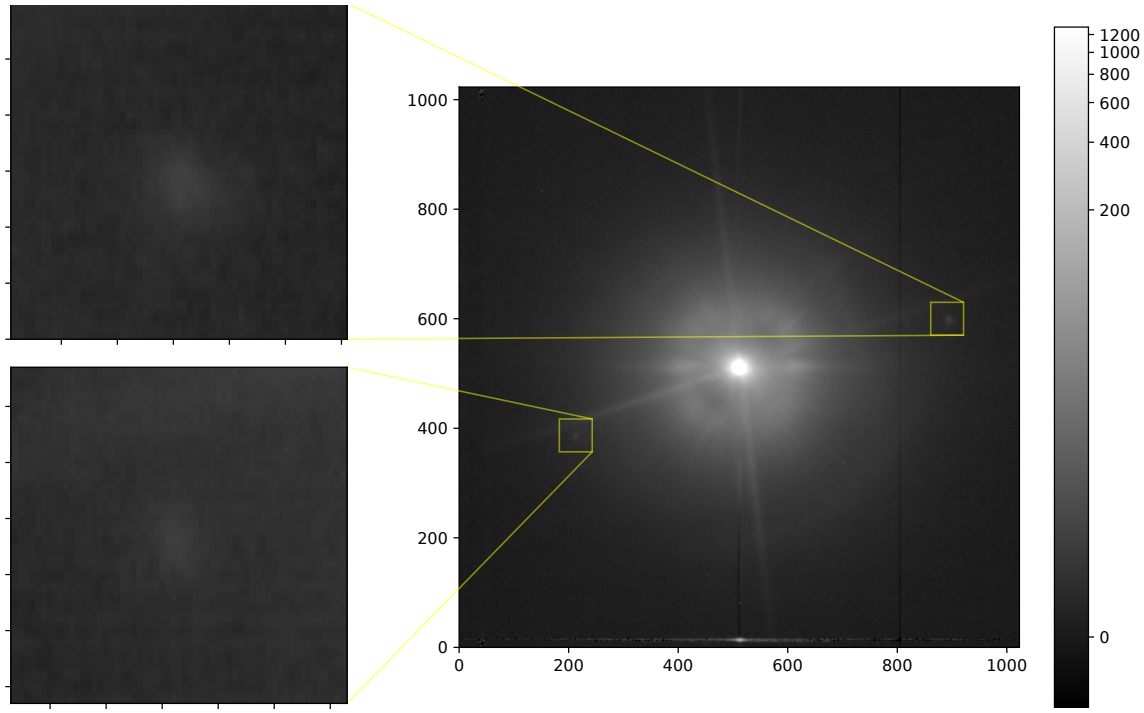


Figure 3: An image of the circumstellar disk HD142527, where the two ghosts are indicated.

If we want to confirm the signal from our ghost or also from other objects like exoplanets, which usually can not be seen by eye, we use the signal to noise ratio S/N. This means we do aperture photometry for several points around the star which are at the same separation from our star as the ghost (or exoplanet etc.), as in figure 4. From this we can calculate the standard deviation σ from the mean of all aperture fluxes and the signal to noise ratio. If the signal to noise ration is larger than the standard deviation, this means that we have a source (ghost, exoplanet) at this position.

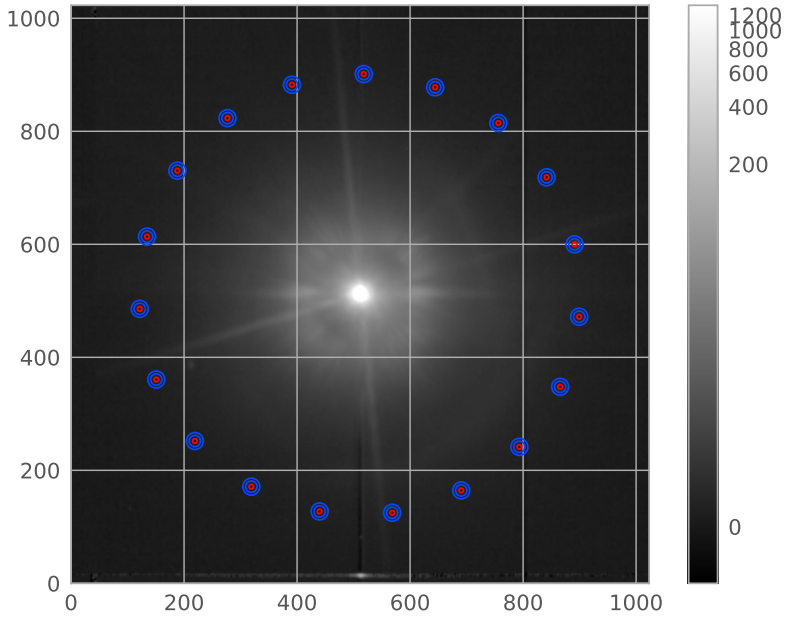


Figure 4: In order to confirm the signal from the ghost, we do aperture photometry for several points around the star which are at the same distance from the star as the ghost. We then calculate the standard deviation of all the aperture fluxes and from there find the signal to noise of the ghost's aperture. If the signal to noise is larger than the standard deviation, the position of the ghost is confirmed.

The standard deviation is calculated by

$$\sigma = \sqrt{\frac{1}{k-2}\sum_{ap=2}^{k}(F_{ap} - F_{mean})^2}, \tag{3}$$

where the aperture of the ghost is at $ap = 1$, k is the number of apertures and F_{mean} is the mean flux of the apertures (without the aperture of the ghost) given by

$$F_{mean} = \frac{\sum_{ap=2}^{k} F_{ap}}{k-1}. \tag{4}$$

From this we can find the signal to noise ratio as

$$S/N = \frac{F_1 - F_{mean}}{\sigma}. \tag{5}$$

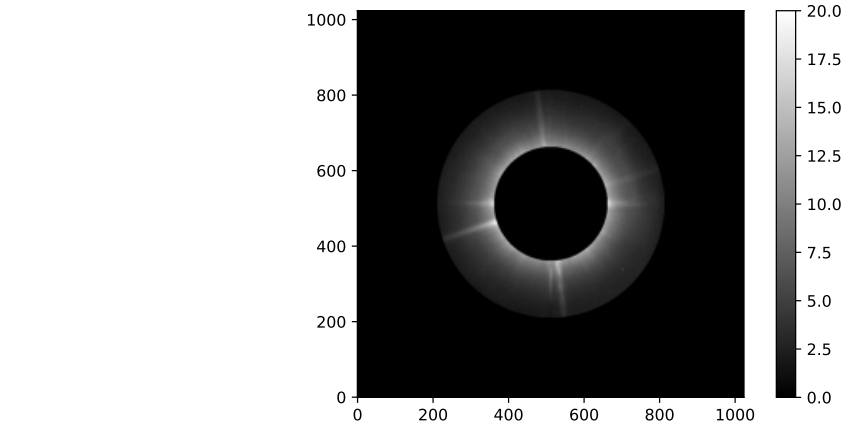
The results we get from our data about the ghosts are shown in table 1. It confirms that both ghosts are where we expected them to be, since for both the signal to noise is larger than the standard deviation.
We also want to compare the intensity of the ghosts to the intensity of the star. For this we calculate the ratio between the aperture flux of the ghost and the star. We find that the ghosts are about 10^{-4} times less bright then the star.

	Ghost 1	**Ghost 2**
S/N	33.9	16.5
σ	7.9	9.8
Ratio	$1.1 \cdot 10^{-4}$	$7 \cdot 10^{-5}$

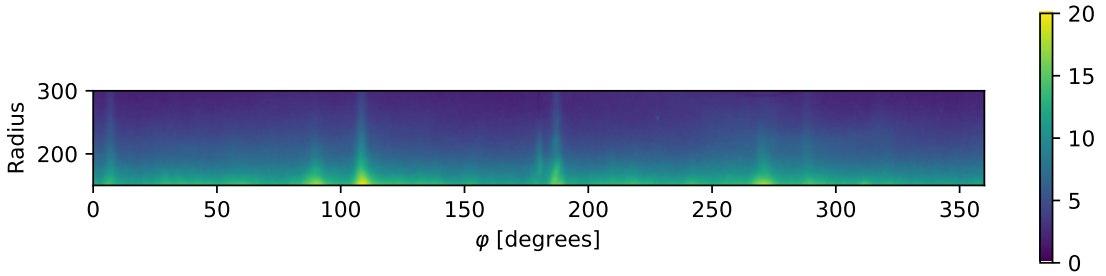
Table 1: Signal to Noise of the ghosts and their brightness with respect to the sun.

3 Transformation to the r-φ plane

We have now seen that the ghosts are about 10^{-4} times less bright then their star, but potential exoplanets are even less bright. A Jupiter like exoplanet would be 10^{-9} less bright (reflecting light) and an Earth like even only $2 \cdot 10^{-10}$. Therefore the data must be really sensitive, with a high contrast and a high spatial resolution. As we can see in figure 3 the star produces strong spiders and speckles which have similar brightness as the ghosts or are even brighter. Therefore exoplanets which are situated in this regimes cannot be detected, unless we are able to take out the signals of the spiders and speckles without taking away other signals, like the ones from exoplanets.
If we take a closer look at figure 3 we observe that the structure of the spiders and the speckles are radially oriented around the star. In order to get rid of this effects it might be a good idea to transform the image into the r-phi plane, where we define the star to be at radius zero. After the transformation the spiders and speckles are distributed along the phi axis and they become weaker along the r axis. The image before and after the warping is shown in figure 5, where we only warped the part of the image which is within radius 150 to 300 pixels.



(a)



(b)

Figure 5: We mask the region (radius=150-300 pixels) which will be warped to the r-φ plane (a). The resulting warped image (b), where we used spline interpolation.

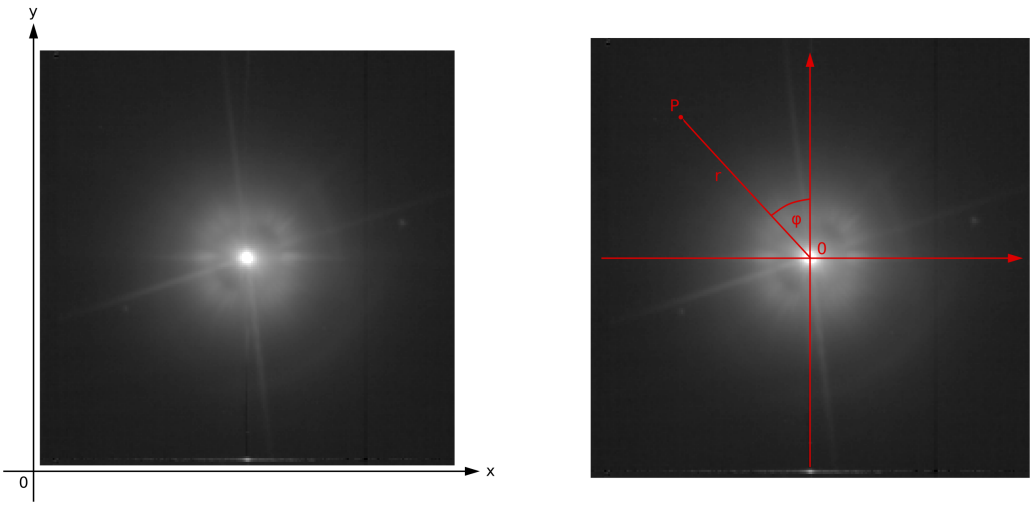
Lets have a look at how one can transform the image into the r-phi plane. This kind of transformation is called image warping in image processing. In our case the image warping is based on a specific transformation, namely the transformation from Cartesian to polar coordinates and is therefore also called polar-cartesian distortion.
In general an image warping is based on a transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$, such that

$$\vec{x} \mapsto \vec{u} = \begin{pmatrix} T_u(x, y) \\ T_v(x, y) \end{pmatrix}, \tag{6}$$

where $\vec{x} = \begin{pmatrix} x \\ y \end{pmatrix}$ and $\vec{u} = \begin{pmatrix} u \\ v \end{pmatrix}$. When we want to warp an image f into an image g we do the following calculation

$$g(\vec{u}) = g(T(\vec{x})) = f(\vec{x}). \tag{7}$$

This means that at pixel $\vec{u}$ the computed image g has the same intensity as the original image f at pixel $\vec{x}$. [1]
When we want to describe the position of a pixel we use Cartesian coordinates, the way it is shown in figure 6 (a), but there are also other ways to describe the location of the pixels. Figure 6 (b) shows the image with a polar coordinate system, where we chose the origin to be in the center (where the stars position is).



(a) (b)

Figure 6: To describe the location of pixels in an image one usually uses Cartesian coordinates (a), but we can also use polar coordinates (b).

In order to change from Cartesian coordinates $x, y \in \mathbb{R}$ to polar coordinates $r \in [0, \infty)$, $\varphi \in [0, 2\pi)$ one needs to do the following calculations:

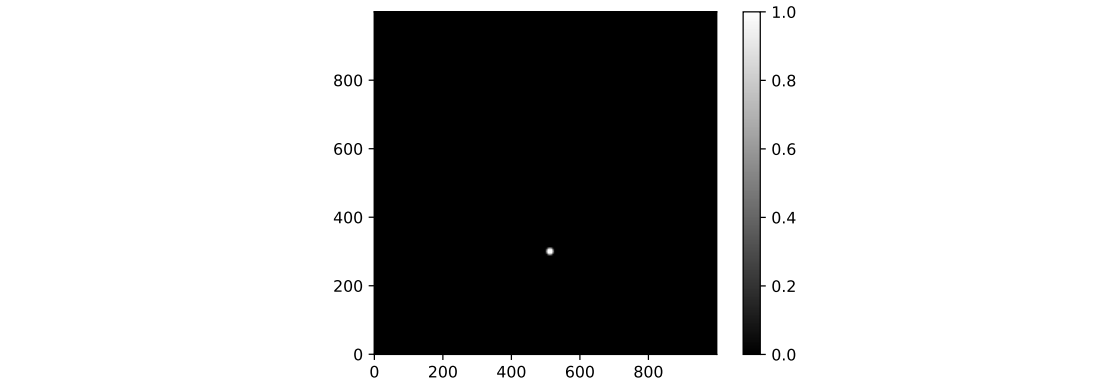
$$r = \sqrt{x^2 + y^2} \qquad \varphi = \arctan\left(\frac{-x}{y}\right). \tag{8}$$

For the back transformation from polar to Cartesian coordinates we compute

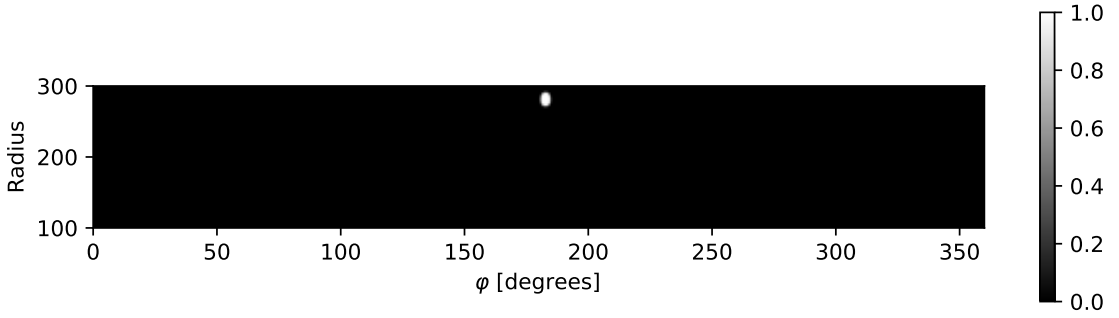
$$x = -r\sin(\varphi) \qquad y = r\cos(\varphi). \tag{9}$$

The transformation of the pixels from Cartesian to polar coordinates results in pixels which are not squares. However for our resulting image in the r-φ plane, we need square pixels. This pixel distortion happens because the number of pixels included at a certain radius away from the center increases proportionally to the radius. This means that there are twice as many pixels at a radius of 100 (pixels) than at radius 50. We now have two options after we have defined the grid of our r-φ plane. Either we choose the grid such that at $r = r_{max}$ every pixel in the grid corresponds to a pixel in the original image and for smaller radii we have pixel which are empty. Or we choose the size of the grid differently and use interpolation to assign an appropriate value to each pixel in the r-φ plane. We decided to use interpolation, since we cannot work with the data if there are empty pixels. In figure 5 we have already seen an example for this transformation using spline interpolation.
Now lets think about which length the grid should have. Since our main interest is to find round objects like exoplanets, it would be beneficial if round objects are still round after the transformation. In order for this to be satisfied, the number of pixels at the radius position of the object has to stay unchanged. We therefore decided to choose the radius range (the φ angle goes from 0 to 360 degrees and thus covers the whole circle), such that the object we want to examine is in the middle and the grid size corresponds to the length of the radius range, i.e. for $r_{min} = 100$ and $r_{max} = 300$ the length of the grid the radius direction is $r_{len} = r_{min} - r_{max} = 200$. The resulting grid size in φ direction depends then on the chosen radius range. Namely such that: $\varphi_{len} = 2\pi \cdot (r_{min} + \frac{r_{len}}{2})$. Figure 7 illustrates the effect of this transformation onto circles in the original image (a). In the center of the warped image at $r = 200$ the number of pixels is unchanged by the transformation, this means no interpolation nor averaging was needed. Therefore circles stay circles, if they are placed in the middle of the warped image, see figure 7 (c). If we go to larger radii the number of pixels per radius (in original image) increases. Since in the warped image all radii have the same

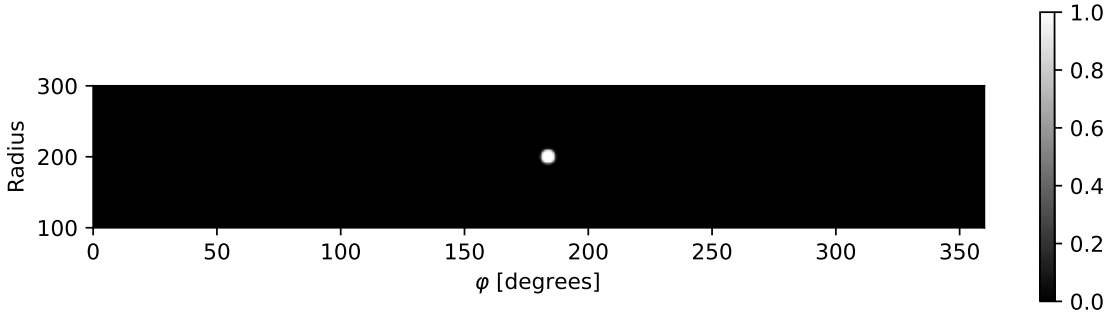
number of pixels r_{len} the transformation averages the information in the original pixels into fewer pixels and the circle becomes elliptic like in its shown in figure 7 (b). The opposite effect happens if we go to smaller radii where the number of pixels per radii (in original image) decreases and interpolation is needed to distribute the original information to an enlarged number of pixels. This results also in an elliptic shape, but with the semi-major axis now in the φ direction, see figure 7 (d).



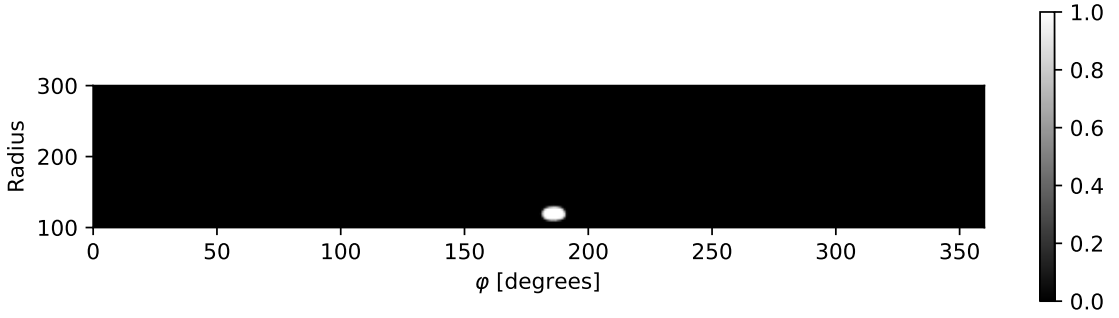
(a)



(b)



(c)



(d)

Figure 7: The warping is defined such that a point in the original image (a) is left unchanged, if it is in the middle of the radius range of the transformed image (c). Otherwise the point will become an ellipse (b) and (d).

In order to perform the transformation to the r-φ plane, we need to define the new shape of the warped image, as discussed before. From this we then define the new pixel grid in polar coordinates and then assign the corresponding Cartesian coordinate values. With this information it is already possible to map the values of the pixels in the original image to the warped one. In our program (shown below [1]) we used the python package scipy.ndimage.map_coordinates() which uses cubic spline interpolation to do the mapping.
As we already mentioned interpolation is used to assign a value to every pixel in the new frame by using the information from the old frame. Or in other words the wholes (where we have no information) are closed by cleverly inventing new values for these wholes through the observation of the information in the neighborhood of the whole. The used spline interpolation fits polynomials to the known values (in our case third order polynomials) in the neighborhood and takes then the values given by the polynomials. We chose this method, because we received the best results with it, but one can also use different interpolation methods like nearest neighbor interpolation or the bilinear interpolation.

```
def to_rphi_plane(image, im_shape, r_min, r_max):
    """
    Warping to r-phi plane.

    Parameters
    ----------
    image : float32, np.array
        An intensity image.
    im_shape : (int, int)
        Shape of image f.
    r_min : int
        Inner radius.
    r_max : int
        Outer radius.

    Returns
    -------
    warped : float32, np.array
        r-phi plane image.

    """
    # Define the shape of the resulting warped image
    r_len = r_max - r_min
    phi_len = int(2*np.pi*(r_min + r_len/2))

    # Define the new grid
    rs, phis = np.meshgrid(np.linspace(r_min, r_max, r_len),
                           np.linspace(0, 2*np.pi, phi_len), sparse=True)

    # Assign the corresponding Cartesian coordinates to the new grid
    xs, ys = rphi_to_xy(rs, phis)
    xs, ys = xs + im_shape[0]/2 -  1, ys + im_shape[1]/2 - 1
    xs, ys = xs.reshape(-1), ys.reshape(-1)
    coords = np.vstack((ys, xs))

    # Create the warped image with spline interpolation 3th order
    warped = scipy.ndimage.map_coordinates(image, coords, order=3)
    warped = g.reshape(phi_len, r_len)

    return warped
```

It is also possible to transform the image back to the Cartesian coordinate map. By simply doing a warping of the image in the r-φ plane back to the original x-y plane.
Lets go back to our example with the circle and have a look if the aperture flux of the circle is preserved trough the warping of the image. We find that the circle has an aperture flux of 306, after the transformation to the r-φ plane this aperture flux is 302. So the aperture flux does not change much through the transformation. After the back transformation we get an aperture flux of 306

which is the same as in the original image. Still we also need to check for this in our data, where we have smooth transitions. We do so by using ghost 1 which is at a radius of 390 pixels in the data of HD142527. For the flux of the aperture in the image we get 231.5. After a transformation to the r-φ plane where we chose $r_{min} = 290$ and $r_{max} = 490$ the flux of the aperture was 232.1. So we can say that the transformation conserves the aperture flux which is a really good and important property.

4 Acknowledgments

A

References

[1] Christian Bauckhage. Numpy / scipy recipes for image processing: General image warping. 01 2019.

[2] Gisin Dominique. HR 8799 imaged in the long I band. 2021.