Simulation of Gravitational Lensing in the Roulette Formalism

Hans Georg Schaathun[1] ⟨georg@schaathun.net⟩
Ben David Normann ⟨ben.d.normann@ntnu.no⟩
Einar Leite Austnes ⟨eaustnes@gmail.com⟩
Simon Ingebrigtsen ⟨simon.ing.89@gmail.com⟩
Sondre Westbø Remøy ⟨sore@live.no⟩
Simon Nedreberg Runde ⟨simonrun@hotmail.com⟩

Department of ICT and Natural Sciences
Faculty of Information Technology and Electrical Engineering
N-6025 Ålesund, Norway

[1]Corresponding author

Keywords

Gravitational Lensing; Simulation; Dark Matter; Roulette formalism

Abstract

Gravitational lensing refers to the deflection of light by the gravity of celestial bodies, often predominantly composed of dark matter. Seen through a gravitational lens, the images of distant galaxies appear distorted. A range of mathematical frameworks exist to model the lensing effect, but reconstructing the dark lens mass remains a difficult problem, where different models give different insight. This paper considers the Roulette formalism due to Chris Clarkson, and develops a simulator visualising the lensing effect according to the formalism. The objective is to enhance our understanding of the formalism in order to understand its strengths and limitation with respect to lens-mass modelling.

I. Introduction

One of the big questions in astrophysics is the mapping of the Universe. Modern telescopes provide enormous amounts of images of the night sky, but about 85% of the mass is dark matter (DM). Emitting no light, this dark matter is not visible on the images. However, because of gravity, dark matter can distort the light from more remote objects. This is called a gravitational lens (GL) (e.g. Bertone & Tait, 2018), because it works analogously to an optical lens.

The shape and location of gravitational lenses can be inferred by studying images of galaxies which appear distorted from our viewpoint, but the calculations are complex and may amount to days of manual work for a single lens. Conventional techniques distinguish between weak and strong lensing, and different techniques are needed depending on the observed data. The Roulettes formalism (Clarkson, 2016a) is a relatively new technique, taking a weak lensing approach to strong lensing effects.

This paper presents the first computational implementation of the Roulette formalism. It simulates the lensing effect by taking a lens model and a source image producing a distorted image as it would be seen through the lens. This allows us to validate the formalism in a range of scenarioes. The graphical user interface allows the cosmologist quickly to test different hypotheses. We have designed the simulator as a flexible framework, which is not restricted to the Roulette formalism. A few other lensing models are supported and new ones may be added. The command-line tool allows efficient bulk generation of images. We hope that this can be used to generate training sets, so that we can invert the lensing function using machine learning, but this is still left as an open question.

II. Background on Gravitational Lensing

All matter, ordinary or dark alike, acts as a lens, distorting the images of distant galaxies. In 1919, the deflection of light by the sun was measured by Eddington during a solar eclipse, and shown to agree with Einstein's theory of general relativity. Since then, theoretical work on lensing has been extensively developed. The scarcity and low resolution of observations, have at times caused pessimism concerning the applicability of this tool for actual observation. But one step at a time, the cosmological community has found ways to observe the phenomenon, and at present, it is booming both with applications and observation. Indeed, GL has become one of the major tools for mining information from the night sky.

One of the clear applications of GL is to understand the nature of dark matter, which according to the present paradigm of cosmology is one of the main constituents in the universe. The other two are ordinary luminous matter ($\sim 5\%$) and the so-called dark energy ($\sim 68\%$). While the former is the stuff that makes up stars, planets and all the rest, dark energy is what causes the accelerated expansion of the late universe. Finally, dark matter ($\sim 27\%$) is the name given to mass indirectly observed in galaxies, but yet not seen. Its elusive nature has haunted cosmology since the 1930s. Although dark matter does not emit light, it must have mass, and thus it bends light like ordinary (so-called luminous) matter. This means that a study of lensing by a distant galaxy is implicitly a study of the dark matter in the lens. By studying lenses at different locations in the sky, one can thus create maps of the distribution of dark matter in the universe. This is important in order to understand the nature of dark matter.

Traditionally, the algebraic framework for GL is divided into two regimes; *weak* and *strong* GL, depending on the level of distortion. In both cases, the physical phenomenon is the same, and in practice, one would expect to see both weak and strong lensing around the same lens, particularly in the case of cluster lenses. For this reason, Clarkson (2016a, 2016b) extended the weak lensing framework also to cover strong lensing[1], resulting in what he called the *Roulette formalism*. In theory, it should be possible to use the Roulette formalism to reconstruct the lens mass from images of distant galaxies subject to GL. The usefulness of this formalism is seen in its ability to go beyond shear measurements (by incorporating higher-order effects to arbitrary order), thus incorporating effects that are typically considered in strong-lensing scenarios. The ability to do so could prove useful as data received from the night sky drastically increases in amount and accuracy, and automation of cluster lens-mass reconstruction could be implemented through machine learning.

The well acquainted reader may wonder why one would use the Roulette formalism (expansion approach) for simulation when one could use the ray-trace equation instead. The purpose of the simulation, in this paper, is to understand, analyse, and further develop the Roulette formalism itself. The purpose of the Roulette formalism is to reconstruct the lens, using the algebraic framework it provides. It is a weak-lensing approach to strong lensing[2]. Thus, a 'Kaiser-Squire'-like inversion technique (Normann & Clarkson, 2020), will in principle provide information about arbitrary-order derivatives of the lensing potential at any point evaluated. It is plausible that this could provide an advantage in cluster lens-mass reconstruction.

In order to make use of the Roulette formalism to such ends, a number of questions should be answered. Firstly, since the Roulette formalism builds on a series expansion which has to be truncated, it is important to know if the region of convergence is large enough to give satisfactory images in practice. Secondly, is it possible to generate images at a reasonable speed? In this work we answer these questions by implementing numerical computation of the Roulettes formalism.

III. The Roulette formalism and its computation

The Roulette formalism was introduced by Clarkson (2016a), and Clarkson (2016b) provides complete details. Our presentation below will differ a little from conventional presentations in physics, in order to emphasise computable functions which can be used to calculate the distorted image. Readers who want a fuller

[1]Starting from the geodesic deviation equation.

[2]As opposed to (Fleury et al., 2017), in which a strong-lensing approach to weak lensing is proposed.

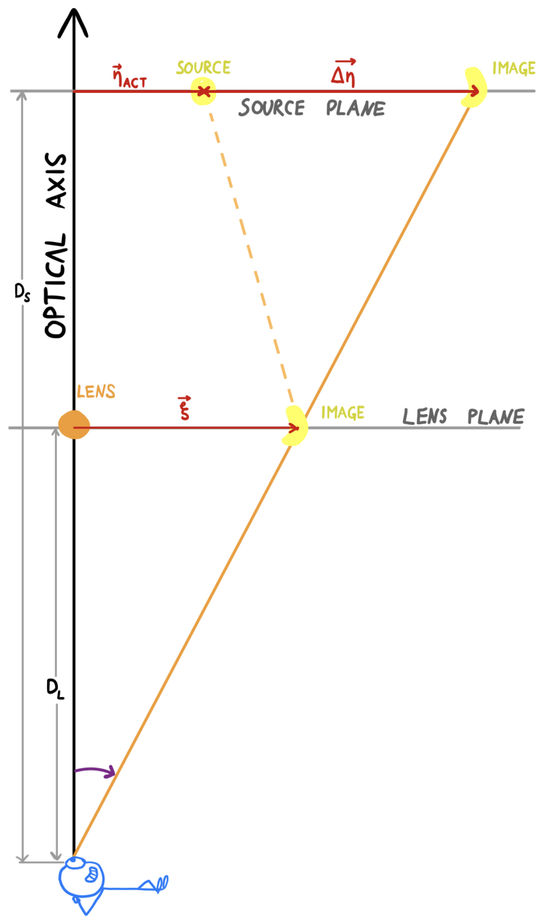


Fig. 1: The figure shows the set-up for the flat-sky approximation, with the source plane (the lens plane) a distance D_S (D_L) from the observer. Compare with Figure 2 for more details.

understanding of the algebraic model should consult Clarkson's original work.

We study two distant objects in the universe, namely the (gravitational) lens L at distance D_L from Earth and the (light) source S at distance D_S. Adopting the *thin-lens approximation*, we assume that the lens mass is concentrated in a plane orthogonal on the line of sight through its centre. The source image is considered only as the 2D projection (image) of its emitted light. With astronomical distances and a relatively small angle of view, we can assume planar projections; this is known as the *flat-sky approximation*. We consider two different images of the source. The source image is the ideal projection, as it would have been observed absent any obstructions. The distorted image is the image as it can be observed when light is deflected by the lens.

The observed lensing is decomposed into two steps, as shown in Figure 2. The first step is a translation (deflection), corresponding to the difference $\boldsymbol{\Delta\eta}$ between actual ($\boldsymbol{\eta}_{\rm act}$) and apparent ($\boldsymbol{\eta}_{\rm app}$) source-plane position. In the roulette formalism, this translational part of the lensing is given as

$$\boldsymbol{\Delta\eta} = \boldsymbol{\eta}_{\rm app} - \boldsymbol{\eta}_{\rm act} = -D_S \cdot (\alpha_1^0, \beta_1^0), \tag{5}$$

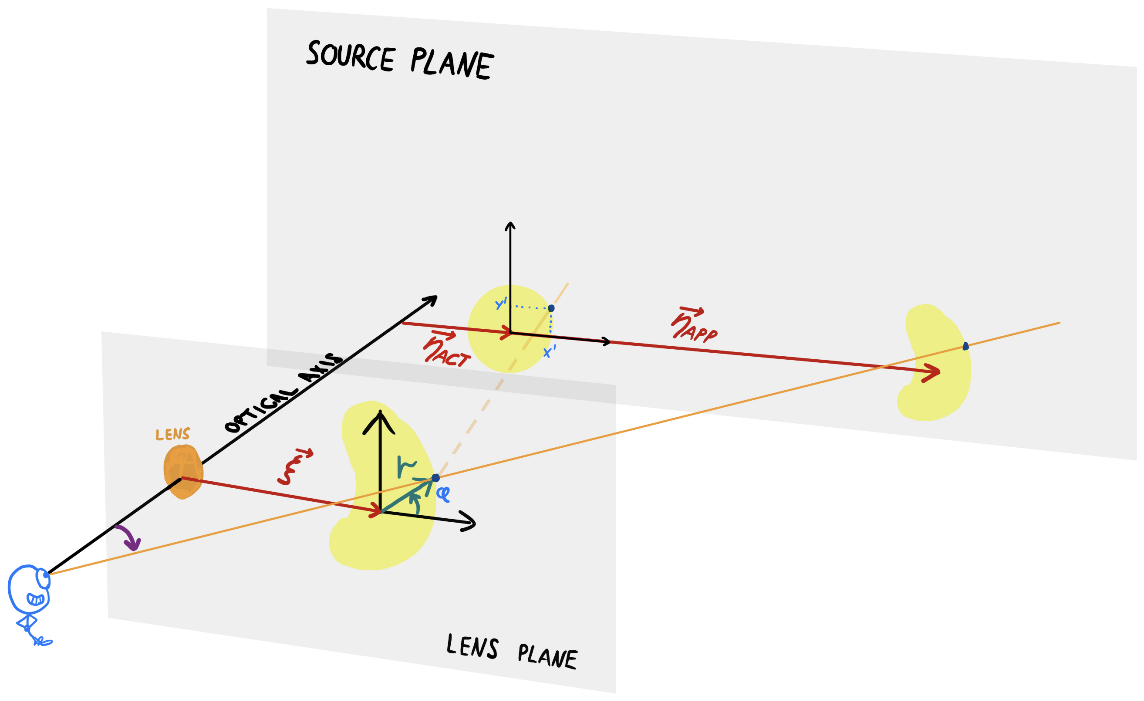


Fig. 2: The figure shows the set-up for the model used. In particular, the local coordinate systems used in the source plane and lens plane are shown. Compare with Figure 1.

$$\alpha_s^m = -\frac{1}{2^{\delta_{0s}}} D_{\mathrm{L}}^{m+1} \sum_{k=0}^{m} \binom{m}{k} \left(\mathcal{C}_s^{m(k)} \partial_{\xi_1} + \mathcal{C}_s^{m(k+1)} \partial_{\xi_2} \right) \partial_{\xi_1}^{m-k} \partial_{\xi_2}^{k} \psi, \tag{1}$$

$$\mathcal{C}_s^{m(k)} = \frac{1}{\pi} \int_{-\pi}^{\pi} \mathrm{d}\phi \sin^k \phi \cos^{m-k+1} \phi \cos s\phi, \tag{2}$$

$$\beta_s^m = -D_{\mathrm{L}}^{m+1} \sum_{k=0}^{m} \binom{m}{k} \left(\mathcal{S}_s^{m(k)} \partial_{\xi_1} + \mathcal{S}_s^{m(k+1)} \partial_{\xi_2} \right) \partial_{\xi_1}^{m-k} \partial_{\xi_2}^{k} \psi \tag{3}$$

$$\mathcal{S}_s^{m(k)} = \frac{1}{\pi} \int_{-\pi}^{\pi} \mathrm{d}\phi \sin^k \phi \cos^{m-k+1} \phi \sin s\phi. \tag{4}$$

TABLE I: Constitiuent definitions for the distortion function.

where (α_1^0, β_1^0) is a vector of roulette amplitudes, as defined in Table I. The second step is the actual, non-linear distortion. The distorted image is drawn in a local co-ordinate system in the lens plane, centred at $\boldsymbol{\xi} = (\xi_1, \xi_2)$, which corresponds to $\boldsymbol{\eta}_{\mathrm{app}}$ in the source plane. We write $\xi = |\boldsymbol{\xi}|$ for the distance between the distorted image and the lens in the lens plane. Since $\boldsymbol{\xi}$ and $\boldsymbol{\eta}_{\mathrm{app}}$ lie on the same line through the viewpoint (cf. Figure 1), we have

$$\xi = |\boldsymbol{\xi}| = \frac{D_{\mathrm{L}}}{D_{\mathrm{S}}} \cdot |\boldsymbol{\eta}_{\mathrm{app}}|.$$

Following Clarkson, we use polar co-ordinates (r, ϕ) for the distorted image. The source image is described in Cartesian co-ordinates (x', y') centered at $\boldsymbol{\eta}_{\mathrm{act}}$ in the source plane. Thus the light observed at a position (pixel) (r, ϕ) is drawn from a different position (pixel) $(x', y') = \mathcal{D}(r, \phi)$ in the source image. From Eq. 48 in Clarkson (2016b) it is possible to show that the mapping $\mathcal{D}$ is given as

$$\frac{D_{\mathrm{L}}}{D_{\mathrm{S}}} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix} = r \cdot \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix} + \sum_{m=1}^{\infty} \frac{r^m}{m! \cdot D_{\mathrm{L}}^{m-1}} F_s^m \tag{6}$$

where

$$F_s^m = \sum_{s=0}^{m+1} c_{m+s} \left(\alpha_s^m \boldsymbol{A}_s + \beta_s^m \boldsymbol{B}_s \right) \begin{bmatrix} C^+ \\ C^- \end{bmatrix} \tag{7}$$

$$C^{\pm} = \pm \frac{s}{m+1}, \tag{8}$$

$$c_{m+s} = \frac{1-(-1)^{m+s}}{4} = \begin{cases} 0, & m+s \text{ is even,} \\ \frac{1}{2}, & m+s \text{ is odd,} \end{cases} \tag{9}$$

and

$$\boldsymbol{A}_s = \begin{bmatrix} \cos(s-1)\phi & \cos(s+1)\phi \\ -\sin(s-1)\phi & \sin(s+1)\phi \end{bmatrix}, \tag{10}$$

$$\boldsymbol{B}_s = \begin{bmatrix} \sin(s-1)\phi & \sin(s+1)\phi \\ \cos(s-1)\phi & -\cos(s+1)\phi \end{bmatrix}. \tag{11}$$

The coefficients α_m^s and β_m^s depend on the lens potential $\psi(\xi_1, \xi_2)$, from which one may derive the physical properties of the lens. The general formulae are shown in Table I. In practice the sum in (6) has to be truncated by limiting $m \leq m_0$ for some m_0.

A general implementation for arbitrary ψ would be intractible, but for many common lens models, it is possible to derive computationally tractible forms. The two simplest, but yet very popular, lens models are the point mass and singular isothermal sphere (SIS). Confer e.g. with Schneider et al., 1992, Sections 8.1.2 and 8.1.4 for more on the point-mass and SIS profiles, respectively. For the point mass, an exact algebraic solution on closed form exists, and we have implemented both this, and its Roulette approximation. For SIS, we have implemented the Roulette formalism together with the more customary point-wise application of the ray-trace equation.

A. *Point-mass lens*

Without loss of generality, one may assume that the centre of mass of the source is located on the positive x-axis. Using the general equations of Clarkson (2016b), it is straight forward to find the following formula for point-mass lenses as a special case:

$$\frac{D_\mathrm{L}}{D_\mathrm{S}}\begin{bmatrix}x'\\y'\end{bmatrix} = r\begin{bmatrix}\cos\phi\\\sin\phi\end{bmatrix} - \frac{R_\mathrm{E}^2}{\xi}\sum_{m=1}^{\infty}(-1)^m\left(\frac{r}{\xi}\right)^m\begin{bmatrix}\cos(m\phi)\\-\sin(m\phi)\end{bmatrix}. \tag{12}$$

In the above, R_E is the Einstein radius, which is determined by the gravity (or mass) of the point-mass lens, and thus determines the strength of the lensing effect. An approximation of the mapping can be calculated using the sum from $m = 1$ to some finite number m_0 with increasing accuracy as $m_0 \to \infty$. This model will be referred to as the *finite point-mass model.* Using analytic continuation, the infinite sum can be calculated and extended outside this region. Using geometric series, it can be written in closed form as follows (Clarkson, 2016b):

$$\frac{D_\mathrm{L}}{D_\mathrm{S}}\begin{bmatrix}x'\\y'\end{bmatrix} = r\begin{bmatrix}\cos\phi\\\sin\phi\end{bmatrix} + \frac{R_\mathrm{E}^2 r}{r^2+\xi^2+2r\xi\cos(\phi)}\begin{bmatrix}\frac{r}{\xi}+\cos(\phi)\\-\sin(\phi)\end{bmatrix}. \tag{13}$$

This is a standard result in the case of a point mass, and is not a result unique to the Roulette formalism. This model will be referred to as the *exact point-mass model.* The apparent position by the following well-known formula,

$$|\boldsymbol{\eta}_\mathrm{app}| = \frac{|\boldsymbol{\eta}_\mathrm{act}|}{2} + \sqrt{\frac{|\boldsymbol{\eta}_\mathrm{act}|^2}{4} + (\frac{D_\mathrm{S}R_\mathrm{E}}{D_\mathrm{L}})^2}. \tag{14}$$

B. *General recursive formulae*

A key element of the Roulettes formalism is recursive expressions for the amplitudes α_s^m and β_s^m. Proofs are given by Normann and Clarkson (2020). The base case is given as,

$$\alpha_1^0 = -D_\mathrm{L}\frac{\partial\psi}{\partial\xi_1} \quad\text{and}\quad \beta_1^0 = -D_\mathrm{L}\frac{\partial\psi}{\partial\xi_2}. \tag{15}$$

The recursive relations are given as

$$\alpha_{s+1}^{m+1} = (C_+^+)_{s+1}^{m+1}(\frac{\partial\alpha_s^m}{\partial\xi_1} - \frac{\partial\beta_s^m}{\partial\xi_2}) \tag{16}$$

$$\beta_{s+1}^{m+1} = (C_+^+)_{s+1}^{m+1}(\frac{\partial\beta_s^m}{\partial\xi_1} + \frac{\partial\alpha_s^m}{\partial\xi_2}) \tag{17}$$

$$\alpha_{s-1}^{m+1} = (C_-^+)_{s-1}^{m+1}(\frac{\partial\alpha_s^m}{\partial\xi_1} + \frac{\partial\beta_s^m}{\partial\xi_2}) \tag{18}$$

$$\beta_{s-1}^{m+1} = (C_-^+)_{s-1}^{m+1}(\frac{\partial\beta_s^m}{\partial\xi_1} - \frac{\partial\alpha_s^m}{\partial\xi_2}) \tag{19}$$

with

$$(C_+^+)_s^m = 2^{\delta_{0(s-1)}}\frac{m+1}{m+1+s}D_\mathrm{L} \tag{20}$$

$$(C_-^+)_s^m = 2^{-\delta_{0s}}\frac{m+1}{m+1-s}D_\mathrm{L} \tag{21}$$

The astute reader may notice that amplitudes for even sums $s + m$ cannot be found through these relations. However, the contribution from these terms are equal to zero, because of the factor c_{m+s} in Equation (6). In other words, one can calculate all the amplitudes needed from the aforementioned relations.

C. *The Singular Isothermal Sphere (SIS)*

The SIS model is somewhat similar to the point-mass model as they both have circular symmetry. The SIS-model however, assumes that the mass of the GL is distributed in a spherically symmetric shape rather than concentrated at a single point. This means that the final simplifications that were used to get the simple equations (12) and (13) cannot be used for the SIS model. However, we can use the recursive formulae from the previous subsection, with the lens potential given as

$$\psi_\mathrm{SIS}(\xi) = \frac{R_\mathrm{E}}{D_\mathrm{L}^2}\xi. \tag{22}$$

In general it is determined by the mass distribution of the lens.

Remark 1: Readers who inspect the source code will note that we have omitted the factor D_L in ψ, α_s^m, and β_s^m ($C_\pm^+$), and the right hand side of (6). The reason for this is that they all cancel out. Verifying this is tedious but straight forward.

The formula for the apparent position is also different. From Eq. (5) it follows that

$$|\boldsymbol{\eta}_\mathrm{app}| = |\boldsymbol{\eta}_\mathrm{act}| + \frac{D_\mathrm{S}R_\mathrm{E}}{D_\mathrm{L}}, \tag{23}$$

and consequently

$$\xi = \frac{D_\mathrm{L}}{D_\mathrm{S}}\cdot|\boldsymbol{\eta}_\mathrm{app}| = \frac{D_\mathrm{L}}{D_\mathrm{S}}\cdot|\boldsymbol{\eta}_\mathrm{act}| + R_\mathrm{E}.$$

IV. The simulator software

The simulator works with pixmap representations of the source image and the distorted image. The Roulettes model maps Cartesian co-ordinates in the lens plane to polar co-ordinates in the source plane. Hence it is trivial to generate the distorted image pixel by pixel, by simply looking up the corresponding pixel (light ray) in the source image. Fractional pixel co-ordinates may be interpolated, but if high-resolution images are used, this is not necessary. Even though the distorted image is calculated in the lens plane according to the Roulettes formalism, we project it back into the source plane, so that the scale (size) is comparable to the source image.

The simulator is implemented as a C++ library, using OpenCV for image manipulation. Front-end tools are implemted in Python, using Pybind11 to wrap the C++ library. There is a GUI tool, as shown in Figure 3, and a command line tool to generate images in bulk. The software is available in Open Source on github[3]

The simulator is a very simple object-oriented structure, where new lens and source models can easily be added. The abstract *Source* class represents the source image, with concrete subclasses for spherical and ellipsoid lenses. These classes store the source image which is generated upon instantiation. The abstract *LensModel* class represents the gravitational lens with subclasses for point mass and spherical (SIS) lenses. These classes implement the distortion function $\mathcal{D}(r, \phi)$, and store the distorted image as well as a reference to the source object, An update method computes the distorted image, which can be retrieved with a getter function.

The python wrapper does not expose the object model. The *CosmoSim* class has setters for types of lens and source models as well as all the relevant parameters. It exposes the Lens Model's update method and getters for the distorted and the actual image. This reduces code size and simplifies maintenance, since we do not have to keep wrapper classes for all the classes in the C++ library. Still it gives complete access to all the features of the simulator.

A critical step in the SIS model is to calculate all the amplitudes α_s^m and β_s^m. We use Python to pre-generate expressions for each (m, s) pair up to some maximum truncation limit m_0, using the sympy module to differentiate ψ. The resulting algebraic expressions are loaded by the C++ code from a text file and evaluated numerically using the symengine library.

V. Results

The GUI interface (Figure 3) allows the user quickly to experiment with different parameter settings, and visually review resulting distorted images. For the cosmologist on the team, this has proved an invaluable tool, particularly to develop intuition and develop a deeper understanding of both the phenomenon (GL) and the model (Roulettes). A particular point where it proved useful was in understanding the convergence ring and the spurious images which we discuss below. Moreover, it has allowed us to verify the theory.

The spurious images is a model artifact. Calculating the distorted images in the Roulettes formalism with an even truncation threshold m_0 produces $m_0 + 1$ spurious images in a ring roughly centred on the local origin $\boldsymbol{\xi}$. The model is exact at the origin, and a good approximation in a neighhood around it. This is clearly seen in the comparison of the exact point mass model and the Roulettes approximations in Figure 4. On one hand, these simulations show how well the Roulettes formalism matches the exact solution, something which can also be verified quantitatively by computing difference images. On the other hand it illustrates the convergence ring, outside of which the model is meaningless, with the spurious images as a blatant example.

Asymptotically, when the number of terms m_0 tends to infinity, it can be shown that this ring has radius ξ centred on $\boldsymbol{\xi}$, and that it approaches the limit from the outside. This result is provided by Clarkson (2016b) and is called the ring of convergence. We can also see in Figure 4 how the spurious images are smaller for large m_0. When the number of images tends to infinity, the size of each one will tend to zero.

Figure 5 shows an example of the behaviour for different degrees on lensing. When the distance η_{act} between the lens (optical axis) and the source image is larger, compared to the Einstein radius R_{E}, the lensing effect is weaker. If it is sufficiently large, the image fits well inside the convergence ring and is a good representation of the physical behaviour. For stronger lensing effects (Figure 5d), we can see how the image is drawn out towards the spurious image. Thus we have demonstrated a limit for when the Roulettes formalism is satisfactory. It should be noted that we are not limited to calculating the roulette model in the centre of the image, at $\boldsymbol{\eta}_{\text{app}}$ or $\boldsymbol{\xi}$. We can choose any point in the lens plane as the origin and calculate the corresponding point in the source plane using ray tracing. This has been verified both algebraically and experimentally.

Knowing the exact location and size of the convergence ring, we can speed up calculation by omitting pixels outside the ring. The distortion equation (6) is computationally expensive (depending on image size and m_0). This masking is made optional in the tools. Without the masking, the GUI is usable around $m_0 = 16$ for 512×512 image size, but it quickly gets irresponsive for $m_0 \geq 20$. With masking the GUI is acceptably responsive up to at least $m_0 = 50$. These tests have used a desktop computer with an AMD Ryzen 9 5900X 12-Core Processor at 2195.8MHz. The image size of 512×512 is, of course, a lot higher than typical empirical images, but the high resolution may be important for the testing of the theory.

For a more objective performance test, we have done bulk generation of images, using the same desktop computer. Generating 1000 images at 400×400 resolution took 35½s walltime and 10 minutes 5 seconds CPU time for $m_0 = 16$. For $m_0 = 50$, it took 4 minutes 38

[3]The experiments reported here have used v2.0.2 at https://github.com/CosmoAI-AES/CosmoSim/releases/tag/v2.0.2.

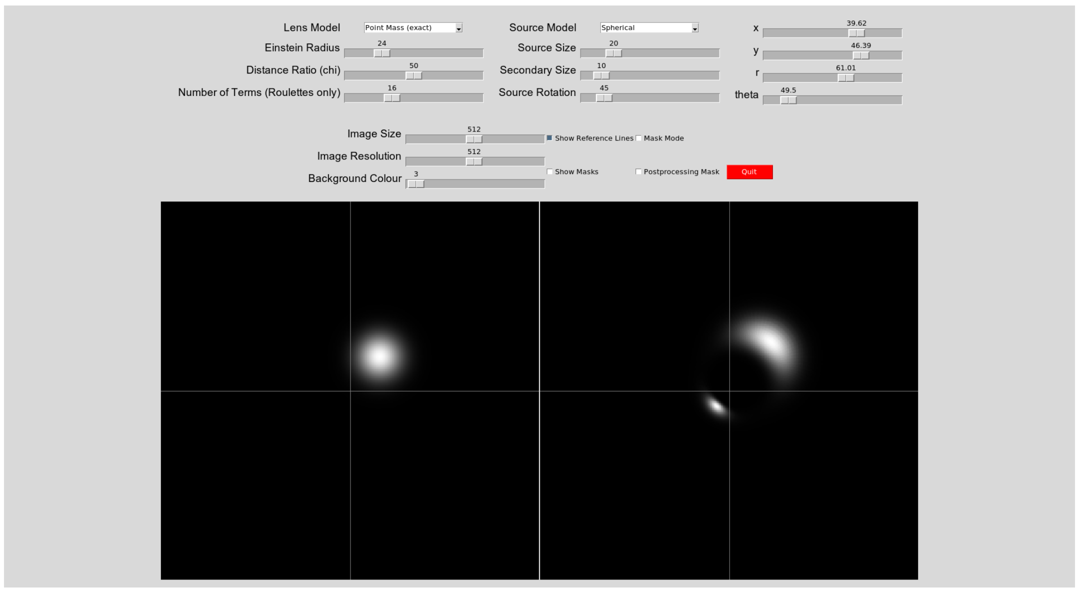

Lens Model: Point Mass (exact)
Einstein Radius: 24
Distance Ratio (chi): 50
Number of Terms (Roulettes only): 16
Source Model: Spherical
Source Size: 20
Secondary Size: 10
Source Rotation: 45
x: 39.62
y: 46.39
r: 61.01
theta: 49.5
Image Size: 512
Image Resolution: 512
Background Colour: 3
[✓ Show Reference Lines] Mask Mode
Show Masks
Postprocessing Mask

Fig. 3: The GUI for the Simulator.

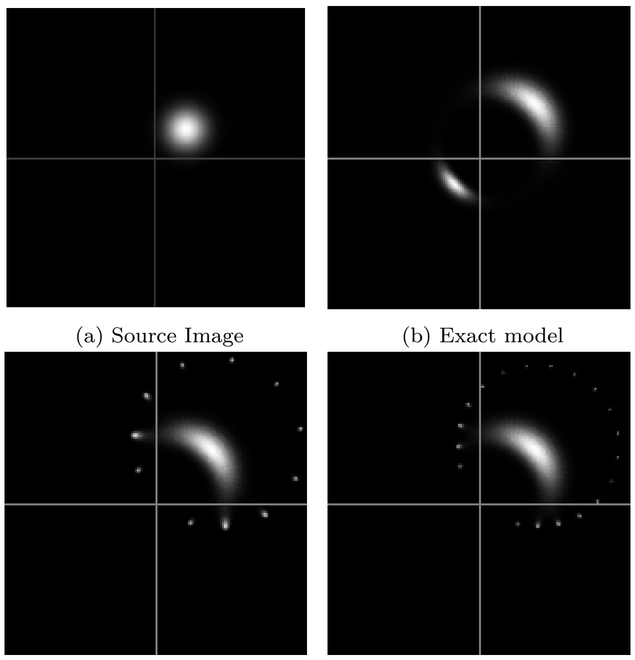

(a) Source Image
(b) Exact model
(c) Roulettes with 10 terms
(d) Roulettes with 20 terms

Fig. 4: Examples with a spherical source and point mass lens; $D_{\mathrm{L}}/D_{\mathrm{S}} = 50\%$, $\xi = 22$, $\theta = 45°$, $R_{\mathrm{E}} = 14$, $\sigma = 7$.

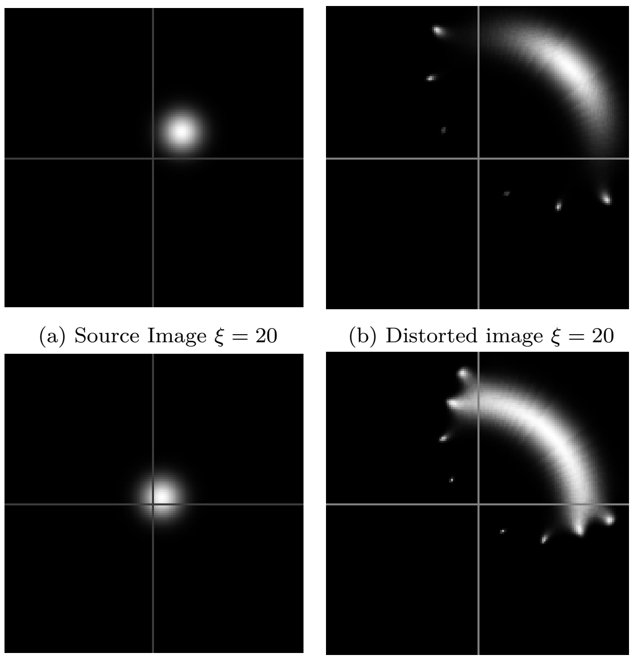
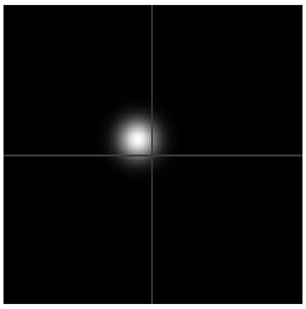

(a) Source Image $\xi = 20$
(b) Distorted image $\xi = 20$
(c) Source Image $\xi = 5$
(d) Distorted image $\xi = 5$

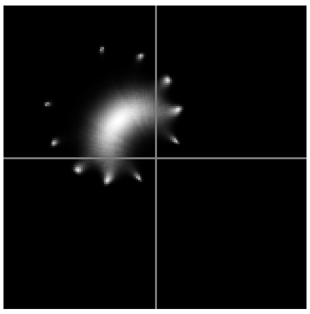
Fig. 5: Examples with a spherical source and SIS lens, with different source positions; $D_{\mathrm{L}}/D_{\mathrm{S}} = 50\%$, $\theta = 45°$, $\xi = 24$, $\sigma = 7$.

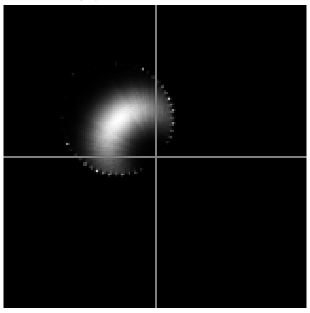
seconds walltime and 81 minutes 24 seconds CPU time, and for $m_0 = 150$, 44 minutes 41 seconds walltime and 11h19 CPU time. This is very acceptable, although interactive applications may not be able to go much above $m_0 = 50$. For the purpose of machine learning, the training set generation is negligible compared to the training time, as it should be.

VI. Impact and Conclusion

Our simulation model provides a computational representation of the algebraic Roulettes formalism (Clarkson, 2016a). An important motivation has been to bridge the gap between computer scientists and physicists, by developing a model which is meaningful in both domains. This is a necessary first step to open up this important research field from cosmology for a wider community, most importantly for machine learning which may be able to invert the distortion function.


(a) Source Image


(b) 10 terms


(c) 50 terms

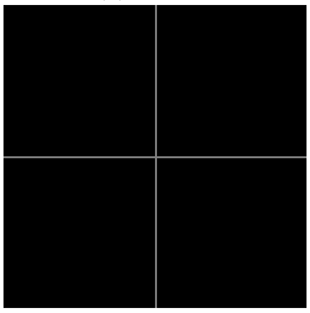

(d) 150 terms

Fig. 6: Examples of the spurious images for various numbers of terms; $D_\mathrm{L}/D_\mathrm{S} = 50\%$, $\theta = 135°$, $\xi = 12$, $R_\mathrm{E} = 8$, $\sigma = 7$.

While a range of simulators exist, this is the first one to visualise the roulette formalism, and the use of the visual interface has helped clarifying several aspects of the formalism, such as the convergence ring. Bulk generation of images is done more efficiently using the ray-tracing equation directly. This has also been implemented (v2.1.0) to be readily available open-source. We do not compete with simulators like Lenstronomy (Birrer & Amara, 2018) and PyAutoLens (Nightingale et al., 2021), which provide comprehensive modelling and model-fitting for strong lensing systems. Instead, our intention is to provide a tool to test and develop the roulette formalism further, and eventually use it to develop new and more efficient techniques for lens mass reconstruction in complex cluster lenses.

We have not given any results on machine learning. The first rudimentary tests are promising, but more work is needed before it is ready for discussion. In the future, we hope to train models to predict the Roulette amplitudes (α_s^m, β_s^m) which provide a local, high-order description of the lensing potential ψ in a selected point, and in turn use this to postulate expressions for ψ. Meanwhile, the simulator has other uses, as a visual tool for testing and exploring hypotheses in cosmology. Somewhat unanticipated, our simulations have revealed problems and limitations in the Roulettes formalism, and thus identified needs for further research.

This work is a mere starting point, leaving several interesting open problems. Development of machine learning models to reconstruct the lens profile and possibly the source image has already been mentioned. To achieve this, we will also have to adapt our system to simulate the noisy, low-resolution data in real images of the night sky. An independent line of research is computational models for a broader range of lens models including cluster lenses.

Acknowledgements

The first prototype of the simulator was a final year project (BSc) by four of the authors (Ingebrigtsen et al., 2022), and the work would have been impossible without their initial work. The models and prototypes have since been extended and improved by the other two authors. Also, we would like to thank Chris Clarkson at QMUL for his various inputs.

References

Bertone, G., & Tait, T. M. P. (2018). A new era in the search for dark matter. *Nature*, *562*(7725), 51–56. https://doi.org/10.1038/s41586-018-0542-z

Birrer, S., & Amara, A. (2018). Lenstronomy: Multi-purpose gravitational lens modelling software package. *Physics of the Dark Universe*, *22*, 189–201. https://doi.org/10.1016/j.dark.2018.11.002

Clarkson, C. (2016a). Roulettes: A weak lensing formalism for strong lensing: I. overview. *Classical and Quantum Gravity*, *33*(16). https://doi.org/Artn16lt0110.1088/0264-9381/33/16/16lt01

Clarkson, C. (2016b). Roulettes: A weak lensing formalism for strong lensing: II. derivation and analysis. *Classical and Quantum Gravity*, *33*(24), 245003. https://doi.org/10.1088/0264-9381/33/24/245003

Fleury, P., Larena, J., & Uzan, J. P. (2017). Weak gravitational lensing of finite beams. *Physical Review Letters*, *119*(19). https://doi.org/ARTN19110110.1103/PhysRevLett.119.191101

Ingebrigtsen, S., Remøy, S. W., Runde, S. N., & Austnes, E. L. (2022). Cosmoai: A study of gravitational lensing through simulation and machine learning [Final year dissertation]. https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3003634

Nightingale, J. W., Hayes, R. G., Kelly, A., Amvrosiadis, A., Etherington, A., He, Q., Li, N., Cao, X., Frawley, J., Cole, S., Enia, A., Frenk, C. S., Harvey, D. R., Li, R., Massey, R. J., Negrello, M., & Robertson, A. (2021). ‘pyautolens‘: Open-source strong gravitational lensing. *Journal of Open Source Software*, *6*(58), 2825. https://doi.org/10.21105/joss.02825

Normann, B. D., & Clarkson, C. (2020). Recursion relations for gravitational lensing. *General Relativity and Gravitation*, *52*(3). https://doi.org/10.1007/s10714-020-02677-z

Schneider, P., Ehlers, J., & Falco, E. E. (1992). *Gravitational Lenses.* https://doi.org/10.1007/978-3-662-03758-4