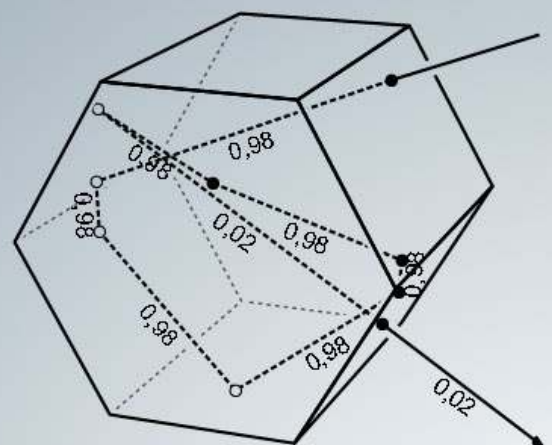
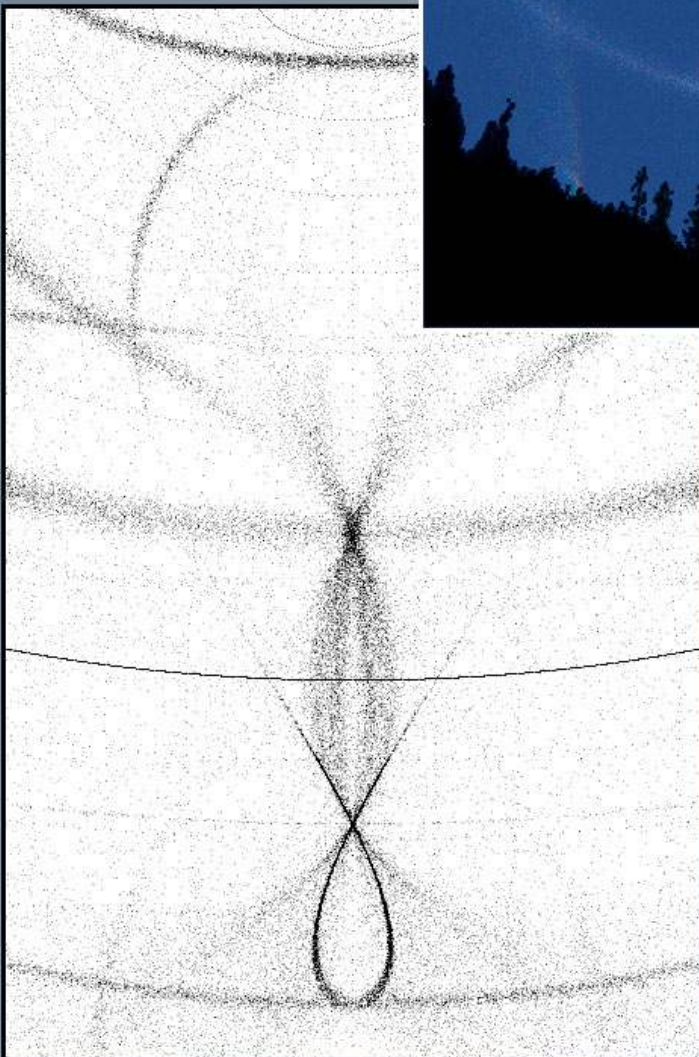
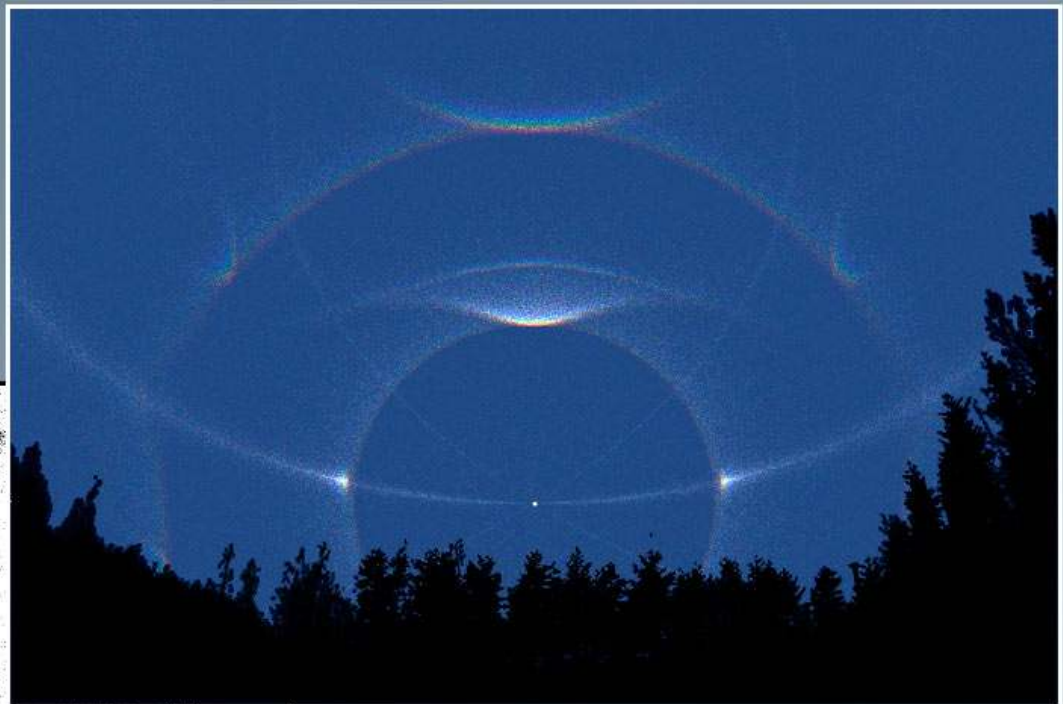


# HALOPOINT 2.0

## User Manual



Software for simulating halo phenomena



# HALOPOINT 2.0

## User manual

*Author:*  
Jukka RUOSKANEN



# Contents

<i>Preface</i> . . . . .	v
<b>1 Introduction</b>	<b>1</b>
<b>2 Getting started</b>	<b>3</b>
2.1 Installation . . . . .	3
2.2 System requirements . . . . .	3
2.3 Notices . . . . .	4
<b>3 Main window: GroupBoxes</b>	<b>5</b>
3.1 General parameters . . . . .	5
3.2 Simulation view . . . . .	6
3.3 Appearance . . . . .	8
3.4 Ice Crystal Population Parameters . . . . .	9
3.5 Prism Face Distances . . . . .	12
3.6 Ice Crystal Visualisation . . . . .	14
<b>4 Main window: Menu items</b>	<b>15</b>
4.1 File Menu . . . . .	15
4.2 Tools Menu . . . . .	17
4.2.1 Multiple Scattering . . . . .	19
4.2.2 Ray Path Filtering . . . . .	21
4.2.3 Simulation Series . . . . .	23
<b>5 Main window: Running a simulation</b>	<b>27</b>
<b>6 Analysis window</b>	<b>29</b>
6.1 Tools Menu . . . . .	30
6.2 Ray Path Analysis . . . . .	31
<b><i>Appendix A</i></b>	<b>39</b>
<b><i>Bibliography</i></b>	<b>43</b>



# Preface

Simulating halos with a computer. Robert Greenler started it in 1970's [1, 2, 3, 4]. In 1984 Eberhard Tränkle and Frank Pattloch introduced Monte Carlo -procedure in their program [5] and took halo simulations a big step forward [6]. Over the years we have seen simulation programs also from at least Walt Tape [7, 8], Lars Gislén [9, 10], Timo Kinnunen, Mika Sillanpää and Jarmo Moilanen [11], Les Cowley and Michael Schroeder [12]... and most likely many more I am not aware of.

Despite the large number of existing simulation programs my personal desire for years was to make one of my own, merely for the fun of it. Already a very long time ago in Oymyakon, Siberia, we were pondering the mathematics and ways to realize a some sort of a version with Jarmo Moilanen. We didn't get far then, and soon other tasks buried the thoughts of a simulation program deep into abyss and out of mind. The itch came back while in Chilean altiplano a few years ago, when discussions with Marko Riikonen finally led me to take concrete steps towards a simulation software. Two weeks after arriving home simulations were running, but in Matlab environment, which proved to be really slow regardless of seemingly endless code optimizations. It soon become obvious, that something had to be done. That "something" was C++.

With very little rudimentary programming skills I was terrified of the task ahead, since transporting code from Matlab into C++ or some other "real" language did not seem as straightforward as some sources advertised. With the assistance of my friend Sami Weissmann I decided to start developing the software in Visual Studio environment using VB.NET and C++.NET -languages. Sami's guidance in the beginning was tremendously valuable and he continued to be my mentor throughout the project.

Solving the actual physical problems and creating algorithms was fun, and as usual, fun things tend to be completed quickly. So the core of the simulation algorithm was already in its final form many years ago. But as appetite grew along the way and new features popped up to be implemented, the truly boring and time consuming software coding task also grew enormously. Available time slots for coding were sporadic and had months in between, so finalizing the software took more time than I initially anticipated.

Now it is here. I decided to share it in the internet in case there is someone out there who might need it. The program has been tested extensively, and the simulations are validated against other simulation softwares, namely those by Sillanpää, Tape and Cowley. Lot of effort has been put in the stability and robustness of the software, but most likely occasional crashes are to be expected.

Marko Riikonen was the primary program tester from the start of the project. His endlessly innovative ideas and criticism improved the software considerably. Also, I

couldn't imagine another person with such experience and insight in halo simulating business as Marko. So, thanks again for the valuable assistance!

The final test period was carried out by Ágnes Kiricsi, Jari Luomanen, Marko Mikkilä, Jarmo Moilanen and Marko Riikonen, to whom I am grateful about their valuable comments. There were some very good notes and most of the suggested improvements and new features will see daylight in the next version of the software.

Finally, I thank Les Cowley, Jarmo Moilanen and Walt Tape for interesting discussions and advices concerning certain details about different aspects of halo simulations. Many of your advices resulted in concrete solutions in the software.

*January 2010 in Riihimäki, Finland,*

*Jukka Ruoskanen*



# 1

## Introduction

This manual describes the functionality of a halo simulation software, HALOPOINT 2.0, that runs on Windows operating system. Although halos are a well understood group of atmospheric phenomena, we are still on the path of understanding them fully. One way to help the progress is to harness the processing capacity of a computer to help us solve the yet uncovered mysteries that the world of halos holds.

Very good books about halos have been written, especially during the last couple of decades, of which the most authoritative have been the books by Greenler [4], Tape [7] and Tape&Moilanen [8]. The reader is urged to get copies of these to become acquainted with halos and to have an inspiring reading experience. Furthermore, there are several interesting websites dedicated to atmospheric phenomena and especially halos, see e.g. Ice Crystal Halos -blog [13] and Atmospheric Optics by Les Cowley [12]. On my own site [14] I have photographs of halos taken over the past 20 years. Numerous other links can be found from Les's site and elsewhere<sup>i</sup>. Due to the available sources of information, a general introduction to halo phenomena is completely skipped in this manual.

In the following chapters all features of HALOPOINT 2.0 are presented, and after reading the manual the reader should have enough information to get started with simulating halos. Although most of the features are quite self-explanatory, there are some that may not be so intuitive. Therefore reading this manual is warmly recommended before starting to use the software. In case questions arise, please contact me by email:

`halopoint2@gmail.com`

HALOPOINT 2.0 uses a Monte Carlo -algorithm, very similar to those explained by Pattloch and Tränkle<sup>ii</sup>, Tape<sup>iii</sup> and Cowley<sup>iv</sup>. To put it short, the user defines a population of ice crystals, their shape and orientation, and HALOPOINT 2.0 calculates

---

<sup>i</sup>Link list of the Finnish Astronomical Association URSA's Halo Section is at: <http://www.ursa.fi/ursa/jaostot/halot/linkit.html>

<sup>ii</sup>Sections 1-3 of ref. [5]

<sup>iii</sup>Appendix F (pp. 132-133) of ref. [7]

<sup>iv</sup>"How HaloSim works", <http://www.atoptics.co.uk/halo/howwork.htm>

all halos arising from this set of ice crystals at a given sun elevation. Several populations can be defined simultaneously. Afterwards the simulations can be extensively studied. Raypaths can be sorted and the origin of the individual halo points can be examined from a visualisation of the raypath through the crystal.

One of the new features HALOPOINT 2.0 offers, which surely has been implemented by others, but to which users have not had access so far, is the possibility to vary prism face distances from the c-axis of the crystal, also in a synchronized manner. This allows one to simulate halos using variable profiles of ice crystals instead of regular hexagons. Also the crystal+raypath -images (see page 37) that can be drawn to every halo point of a simulation image is a feature that has not been offered in previous simulation programs.

# 2

## Getting started

### 2.1 Installation

---

HALOPOINT 2.0 can be downloaded from

<http://www.saunalahti.fi/~jukkrus/halopoint2.html>

where a packed file `HaloPoint2.zip` (size 404 kB) can be found. There is no installation software included in HALOPOINT 2.0. Simply unzip the packed file into a suitable directory in your computer and you are ready to go.

The software consists of two components, the executable `HaloPoint.exe` and a library file `Simulation.dll`, which need to be in the same directory. Some introductory parameter files can be downloaded from the homepage of the software, which help a new user to get started.

### 2.2 System requirements

---

The operating systems in which HALOPOINT 2.0 has been successfully tested are Windows XP (Pro and Home) and Windows Vista. In order for HALOPOINT 2.0 to function properly the Microsoft .NET Framework 1.1 has to be installed. Most (Microsoft) operating systems have it as a default (see if folder `C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322` exists). If it is missing in your computer, it can be freely downloaded from Microsoft pages<sup>i</sup>; look for "Microsoft .NET Framework Version 1.1 Redistributable Package".

Some tasks in HALOPOINT 2.0 Analysis-window eat RAM memory fairly voraciously, so a few gigabytes worth of memory modules may be needed. Basic operation should be possible with significantly less memory consumption.

---

<sup>i</sup><http://www.microsoft.com/downloads/details.aspx?FamilyId=262D25E3-F589-4842-8157-034D1E7CF3A3&displaylang=en>

## 2.3 Notices

---

HALOPOINT 2.0 is a free software, and intended for personal use. You may not redistribute or sell this software in websites or by any other means. HALOPOINT 2.0 is available for download in the hope that it will be useful, but it comes without any warranty. Extensive testing and validating measures have been taken to ensure correctness of the simulations, but since the code has been written as a hobby and not by a professional software developer, there may lurk some yet unidentified errors within the software. Users are encouraged to report all software faults and exceptions to the email address which can be found at the software homepage (currently: [halopoint2@gmail.com](mailto:halopoint2@gmail.com)).

Please, acknowledge the use of HALOPOINT 2.0 in publications and websites and use the following reference in your publications:

HALOPOINT 2.0

<http://www.saunalahti.fi/~jukkruos/halopoint2.html>

©Jukka Ruoskanen

## Main window: GroupBoxes

### 3.1 General parameters

---

In General Parameters -groupbox (fig. 3.1) the basic simulation parameters are defined. These are:

**Sun Elevation:** Enter sun elevation in degrees between  $-90^\circ$  and  $90^\circ$ .

**Sun Diameter:** Select the diameter of the light source from the drop-down list, which has two choices: "True Diameter" ( $= 0.5^\circ$ ) and "Point Source".

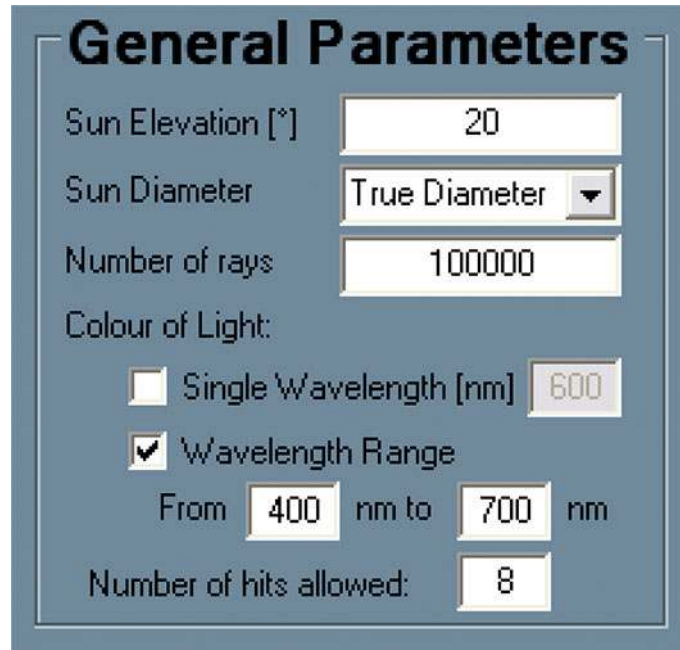
**Number of Rays:** Enter number of rays to be considered in the simulation run. Notice, that this number is not the number of halo points created in the sky. If a ray misses a crystal, or does not get plotted for other reasons, the ray counter is still incremented. Therefore the resulting number of halo points in any simulation is going to be less than the number entered in this textbox. As a rough guideline a simulation with a few hundred thousand light rays takes less than a minute to complete whereas several millions of rays require tens of minutes, even hours, on a regular PC.

**Colour of Light:** Here are two checkboxes to choose from: monochromatic light or a continuum of wavelengths. The limits of the wavelength range are 380 nm in the blue end and 700 nm in the red end.

**Number of Hits Allowed:** Enter the maximum allowed number of crystal face encounters experienced by a single ray. If the light is still trapped by the crystal after this number is exceeded, the treatment is terminated and another ray is taken under consideration.

The refractive index of hexagonal ice as a function of wavelength has been studied by a handful of groups. In HALOPOINT 2.0 the wavelength dependence data has been taken from [15]. A polynomial was fitted to measured data to interpolate the whole range from 380 nm to 700 nm.

Mapping wavelengths to RGB values was solved by using Dan Bruton's model [16], which is an empirical algorithm and produces very good looking spectrum colours.



**General Parameters**

Sun Elevation [°]

Sun Diameter  ▼

Number of rays

Colour of Light:

☐ Single Wavelength [nm]

☒ Wavelength Range

From  nm to  nm

Number of hits allowed:

Figure 3.1: *General Parameters -groupbox.*

In addition, solar irradiance weighting is applied for the wavelength range and on top of that some experimenting with arbitrary factors was needed in order to keep the simulation white balance approximately correct. Thanks to Les Cowley [12] for helpful tips and fruitful discussions concerning the colour issues.

## 3.2 Simulation view

---

In Simulation View -groupbox (fig. 3.2) the camera parameters are defined. The field of view and projection of a photograph can be matched by entering the camera lens type and focal length, camera sensor (or film) size and of course the pointing angle and rotation of the imaging device. The parameters are:

**Lens Type:** Select the projection type of the imaging lens. There are two main types in photographic use: a rectilinear lens and a fisheye lens. Most SLR- and compact camera lenses are of rectilinear type. There are many fisheye projections of which equal area<sup>i</sup> and equidistant projections are most popular by camera lens manufacturers. Both of these are selectable. Most of the modern fisheyes use the equal area projection. See Appendix A.

**Focal Length:** Enter the true focal length of the lens in millimeters. Valid range is 1 - 300 mm.

---

<sup>i</sup>also known as equisolid angle projection

**Sensor Size:** Enter the width (W) and height (H) of the imaging chip or film in millimeters. Valid range is 1 - 300 mm.

**Custom parameters:** There are two parameters,  $a$  and  $b$ , available for the user to fine tune the perspective projection of the resulting image. If unchecked, the chosen projection type will match the ideal projection for the given focal length. If checked, the parameters  $a$  and  $b$  will be used, but for fisheye models only. For rectilinear projection the parameters  $a$  and  $b$  have no effect. For equal area -fisheye both parameters are used, and for the equidistant only parameter  $a$  is used. See additional information in Appendix A.

**Center of Image:** Select image center from the drop-down list, where eight special celestial points are available. The ninth entry in the list is *User Defined*, which when selected enables the textboxes *Azimuth* and *Elevation*. Enter the desired azimuth and elevation of the center of the image (in degrees) in these boxes. Azimuth of the Sun is  $0^\circ$  and increases westwards (observer is inside the celestial sphere). Allowed azimuth values are from  $-180^\circ$  to  $180^\circ$ .

**Rotation:** Enter the rotation angle of the photo. Rotation angle increases clockwise, and angles from  $-180^\circ$  to  $180^\circ$  are accepted.

**Portrait/Landscape:** Check the appropriate checkbox.

In the EXIF data of most cameras the focal length value given is the true focal length of the lens at the time of shooting. Some manufacturers, however, prefer to give the 35mm equivalent focal length of the camera lens, in which case the sensor width and height have to be those of the full frame 35mm system, namely 36 and 24 mm, respectively. If the true focal length of the lens and the sensor size of the camera are known, it is advisable to use these values in order to obtain perfect match between a photo and simulation. In this case one does not need to care about crop factors and such. Sensor sizes can be found from numerous websites, one of the most comprehensive being DPreview<sup>i</sup>. Note, that to get a circular fisheye image, the values of e.g. fig. 3.2 can be used. Additional information about the lens projections and sensor sizes are in Appendix A.

Figure 3.2: *Simulation View* -groupbox.

<sup>i</sup><http://www.dpreview.com/>

### 3.3 Appearance

---

Now that we have the simulation view taken care of we still need to define the appearance of the outcome. This is done in Appearance -groupbox, which has the following parameters:

**Dots:** Select *Coloured* to run coloured simulations. The colours depend on the wavelength range. In case of monochromatic light (single wavelength - checkbox checked) all halo points will have the colour corresponding to the selected wavelength. If black or white dots are desired, the background will automatically be white or black, respectively.

**Background:** In case of coloured dots the background colour can be selected too. List offers *black*, *white* or *user defined* background colours. If *user defined* is selected, a standard Windows colour dialog can be opened by clicking the right mouse button on top of the drop-down list. A custom background colour can be selected from the dialog, and if accepted, the background colour of the drop-down list will change into the selected colour (as well as the simulation sky background).

**Shade Levels:** Enter a number of intensity levels for halo points. A computer display uses 256 intensity levels for each colour channel (**R**ed, **G**reen, **B**lue), so the maximum number of shade levels is 256. Large shade levels require more rays, since the overall intensity of the simulation builds up more slowly then. On the other hand, the outcome is smoother than with fewer rays and more crude shade level selection.



Figure 3.3: *Appearance -groupbox*.



**Show Coordinate Grid:** Coordinate grid becomes visible when checked. The spacing of major- and minor steps can be selected (options are 1, 2, 5 and 10 degrees) and the strength of the grid has three levels to choose from.

**Horizon:** When checked, the horizon is drawn to the simulation.

**Hide Subhorizon:** When checked, the halo points below horizon are not plotted.

Each halo point's intensity and colour is built gradually as the simulation progresses. The three colour channel values of the halo point to be plotted are added to the colour channel intensities of the underlying pixel. If a colour channel intensity reaches value 255, it does not change any more. The other two colour channel values can increase until they too reach the maximum. Then the pixel is white. If the number of shade levels is low, a given pixel on the screen saturates more easily than in case of higher levels setting. Note, that if the shade levels setting is quite low and the number of rays in the simulation is millions, the resulting colours in some parts of the sky can appear unrealistic due to the very limited dynamic range of the computer screen.

### 3.4 Ice Crystal Population Parameters

The shape and orientation of the ice crystals for each population are defined in this groupbox. In HALOPOINT 2.0 there can be ten active populations simultaneously in one simulation run. The checkmark next to the population indicates whether it is active or not. For each crystal parameter a mean, standard deviation and distribution function are defined. The parameters of a population are:

Ice Crystal Population Parameters																	
%	C-axis			Rotation			Prism Aspect Ratio			Upper Pyramid			Lower Pyramid				
	Mean	Std	Distribution	Mean	Std	Distribution	Mean	Std	Distribution	Apex	h	Std	Distribution	Apex	h	Std	Distribution
<input checked="" type="checkbox"/> 1	1	0	3 Gaussian	0	360	Uniform	0.3	0.1	Uniform	56,142	0	0	Uniform	56,142	0	0	Uniform
<input checked="" type="checkbox"/> 2	2	90	1 Gaussian	0	360	Uniform	1.5	0.4	Uniform	56,142	0	0	Gaussian	56,142	0	0	Gaussian
<input checked="" type="checkbox"/> 3	1	90	0.5 Gaussian	90	0.3	Gaussian	1.5	0.1	Uniform	56,142	0	0	Gaussian	56,142	0	0	Gaussian

Figure 3.4: *Topmost part of the Ice Crystal Population Parameters -groupbox.*

**%:** The number in this textbox marks the weight of the current population. It is not percentage despite the symbol; the numbers of the active populations do not need to add up to 100. If two populations are active and have weights 1 and 2, then two thirds of all rays will be cast to the crystals of the latter population.

**C-axis:** In its initial orientation (see fig. 3.5) the crystal has its c-axis vertical. So in plate orientation the c-axis mean has to be 0° and in column orientation 90°. The std of the c-axis marks the tilting angle of the c-axis about the mean in a manner specified by the selected probability distribution function (see below for additional information). The angles are in degrees.

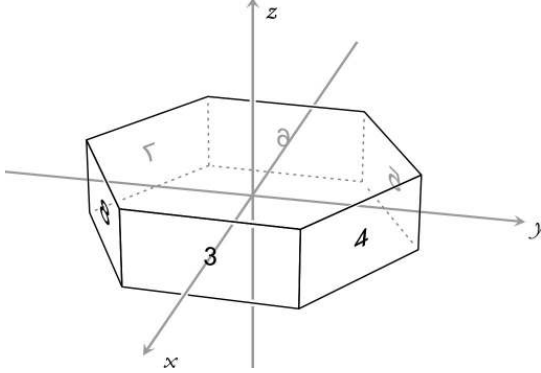


Figure 3.5: The ice crystal in its original orientation:  $c$ -axis is vertical and face 3 normal is towards positive  $x$ -axis and parallel to it.

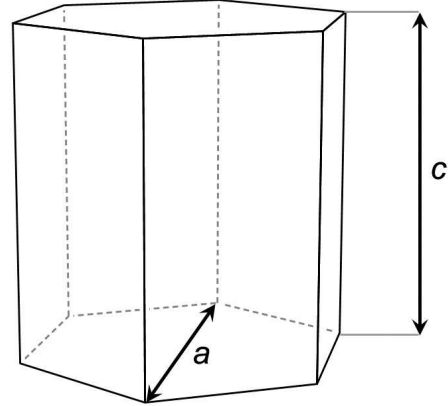


Figure 3.6: The aspect ratio is the ratio of the length of the prism in  $c$ -axis direction to the width of a regular hexagon in  $a$ -axis direction ( $c/a$ ).

**Rotation:** In its initial orientation (see fig. 3.5) the crystal has the normal of face 3 pointing towards positive  $x$ -axis. If crystal is allowed to rotate freely (like in case of random, plate- and column orientations), the mean value is insignificant but the std has to be  $360^\circ$  and the distribution function *uniform*. If rotation is restricted, the direction of the face 3 normal has to be taken into account in order to enable correct raypath labelling. For Parry- and Lowitz-orientations it is advisable to use  $90^\circ$  for the rotation mean in order to turn the crystal in an orientation compliant with the widely used raypath conventions. Enter the allowed rotation deviation into std-textbox and choose the appropriate distribution. The angles are in degrees.

**Prism aspect ratio:** In HALOPOINT 2.0 the prism aspect ratio is defined as  $c/a$  (length of the prism in  $c$ -axis direction / width of the crystal in  $a$ -axis direction), see fig. 3.6. As the profile of the cross-section of the prism may vary, the  $a$ -axis width is as in the case of a regular hexagon. For platelike crystals the aspect ratio is less than 1.

**Upper/Lower pyramid:** A full pyramid has a value of 1. Therefore the height ( $h$ ) mean can have values between 0 (no pyramid) and 1 (full pyramid). The apex angle is given in degrees. For a platelike crystal the *upper* and *lower*-terms make sense, but for other orientations, e.g. column or Parry, they just mean the pyramids at the opposite ends of the column (which may or may not exist between the pyramids). The std and distribution refer to the pyramid height; the apex angle remains fixed.

HALOPOINT 2.0 has two choices for a distribution function: Gaussian and uniform (see figs. 3.7 and 3.8). Standard deviation (*std*) is well defined for Gaussian prob-

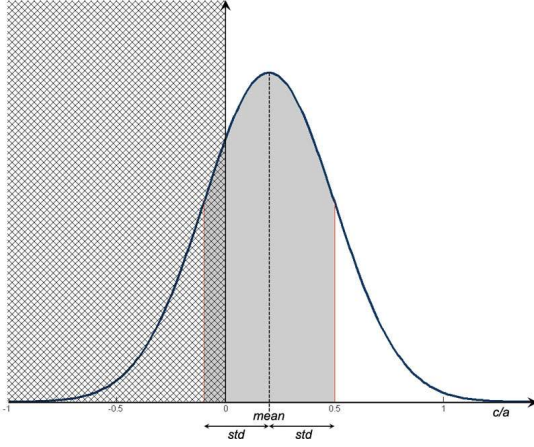


Figure 3.7: An example of a prism aspect ratio distributed according to a Gaussian probability distribution function with *mean* = 0.2 and *std* = 0.3. 68% of all normally (Gaussian) distributed values are within *mean*-*std* and *mean*+*std* (gray area). Here, however, the lower limit has to be  $>0$ , and therefore all negative values of the distribution are brutally cut off (crossed area).

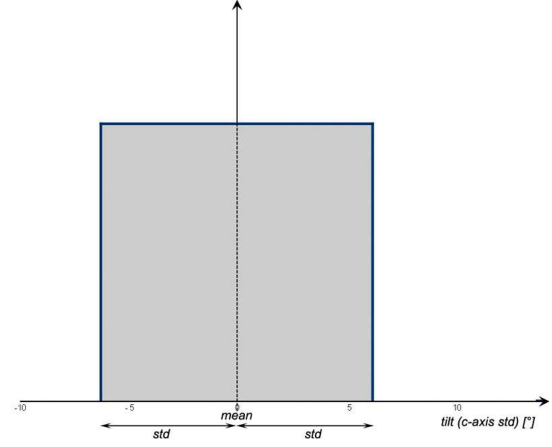


Figure 3.8: A hypothetical example of a crystal tilt with a zero mean and having a uniform distribution of  $6^\circ$  (somewhat unrealistic for tilt to have a uniform distribution, this is an example only). All angles within  $\pm 6^\circ$  from the mean are equally probable.

Table 3.1: Below are examples of ice crystal population parameters (in degrees) for mostly used types: plate orientation, column (singly oriented columns) orientation, Parry (double oriented columns) orientation, random orientation and Lowitz (this case platelike Lowitz-) orientation. Numbers are arbitrary and are here only to illustrate how different orientations are set up. Fine tuning is necessary for each individual case.

		Plate	Column	Parry	Random	Lowitz (plate-)
c-axis	mean	0	90	90	0	0
	std	2	1	0.5	360	40
	distr.	Gaussian	Gaussian	Gaussian	Uniform	Gaussian
rot	mean	0	0	90	0	90
	std	360	360	0.5	360	0.5
	distr.	Uniform	Uniform	Gaussian	Uniform	Gaussian

ability distribution function, but not for a uniform one. In HALOPOINT 2.0 *std* of the uniform distribution means the starting- and endpoints of the interval in which the parameter is allowed to vary in uniform manner. For the parameters whose value can not be less than zero (i.e. prism aspect ratio and pyramid heights), the distribution function is simply cut at zero even though the tail of Gaussian distribution or part of the uniform distribution would fall below zero.

### 3.5 Prism Face Distances

The profile of the ice crystal can be controlled in this groupbox. Ice crystal samples under microscope seldom contain regular hexagons (see fig. 3.9). Instead, most of the crystals deviate from symmetrical shape, or are triangular or tabular in their cross-section. Therefore, in order to approach reality in simulations, it may be necessary to adjust the face distances from the c-axis (see fig. 3.10). The parameters to control are:

**Apply:** If this checkbox is checked, the face distance parameters are taken into account when forming the shape of the crystal. Otherwise the crystals are symmetric, regular hexagons.

**Face Distance from c-axis:** Each crystal face's (3 to 8) distance from the c-axis can be adjusted individually. Examples of the resulting profiles with different face distance combinations are shown in fig. 3.12.

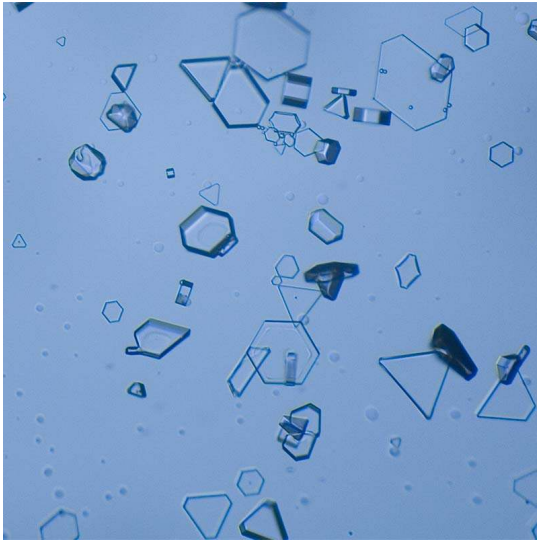


Figure 3.9: *Ice crystal profiles typically have various shapes, and symmetrical hexagons represent a small minority. Photo by Marko Riikonen.*

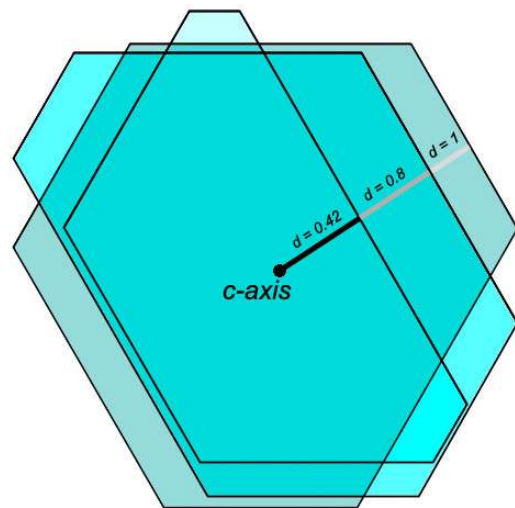


Figure 3.10: *By changing the distance of a face from the c-axis the profile of the crystal can be tuned.*

**Std of Distance:** If face distances are allowed to vary, the std of the variation is defined here individually for each face. Std values are true standard deviations for Gaussian distribution and endpoints of the interval for uniform distribution.

**Sync:** If this checkbox is checked, the face distance variation is synchronized. The face with the lowest number (prism faces are 3 to 8) determines the std in case a synchronization is selected. Then, all faces whose std is  $>0$  vary the same amount. Thus, if a varying triangular crystal profile is desired, a suitable std is entered into face 3 or 4 (does not matter if the rotation angle is not restricted) and all subsequent alternate faces are set to larger than zero. In case face 3 has a desired std, say 0.5, faces 4, 6 and 8 are set to zero and faces 5 and 7 are given a std larger than zero (what ever the number is, the std is determined by face 3 as long as sync-checkbox is checked). The numbers shown in fig. 3.11 result in a synchronized variation of faces 3 and 6 (ranging from a regular hexagon to a clearly tabular profile).

**Distribution:** Select the distribution function according to which the faces vary.

Careless use of face distance variation may result in unrealistic combinations of ice crystal form and orientation. There is no warning mechanism in HALOPOINT 2.0 for unrealistic parameter combinations; as long as the parameters are within allowed bounds, the simulation starts. Verification of the "physical correctness" of the entered parameters is left for the user. The lack of programmatical verification mechanism can also be seen as an advantage; creativity is not restricted by a narrow minded code writer! Almost anything can be done, physically improbable or not.

Apply	Face Distance from c-axis						Std of Distance						Sync	Distribution
	3	4	5	6	7	8	3	4	5	6	7	8		
<input checked="" type="checkbox"/> 1	1.5	2	2	1.5	2	2	0.5	0	0	0.5	0	0	<input checked="" type="checkbox"/>	Uniform
<input type="checkbox"/> 2	1	1	1	1	1	1	0.2	0.2	0.2	0.2	0.2	0.2	<input type="checkbox"/>	Uniform

Figure 3.11: *Topmost part of the Prism Face Distances -groupbox.*

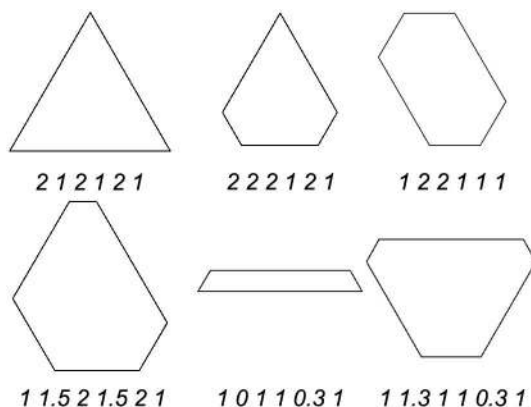


Figure 3.12: *Prism profiles with different face distance -combinations.*

## 3.6 Ice Crystal Visualisation

The parameters which control the shape and orientation of the ice crystal can be confusing without graphical output. In this groupbox the ice crystal shape, dimensions and orientation can be examined before running the simulation. The controllers which adjust the viewing angle and appearance of the crystal drawing are quite self-explanatory. Initially the *view angle control* is set to  $y = 20^\circ$  and  $z = 30^\circ$  (rotation angles of the axes, see fig. 3.5). The angles can be set with the sliders. The image view angles are set to their initial values with the *Reset* -button. The line widths and the backline colour can be tuned from their respective controls.

The crystal drawing does not update automatically whenever the parameters change. The drawing can be updated to agree with the current parameter values by selecting the desired population from the drop-down list and pressing *Draw Selected Crystal* -button. Alternatively the drawing is updated when *Enter* is pressed while the cursor is in some textbox of *Ice Crystal Population Parameters*- or in *Prism face Distances* -groupboxes.

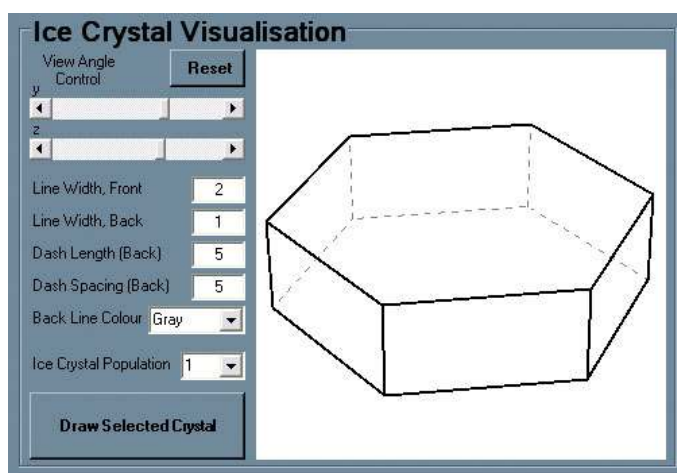


Figure 3.13: *Ice crystal visualization -groupbox.*

# 4

## Main window: Menu items

### 4.1 File Menu

---

File menu in HALOPOINT 2.0 contains the following items (keyboard shortcuts in brackets):

**Save Parameters as... (Ctrl+S):** Complete simulation parameters can be saved to a \*.par -file. Note, that although the parameter files are plain text files, they should not be edited in text editors to assure the compatibility with HALOPOINT 2.0. Same applies to other files generated with HALOPOINT 2.0 as well.

**Load Parameters... (Ctrl+O):** Load parameters from a file. Available file types are the actual parameter file (\*.par), a parameter file associated with a *simulation data* file (\*.oup) and a parameter file associated with a *light simulation data* file (\*.olp). When parameter files are transported from one computer to the next, the language versions of the operating systems tend to cause problems since the numeric formats are typically very different. Compatibility between different language versions is taken care of within the code, but in case problems arise, please, report the issue back to me.

**Save Simulation Data... (Ctrl+Shift+S):** In order to use full functionality of the Analysis-window, detailed simulation data has to be saved into a file (\*.out). Several pieces of data are stored for each halo point of the simulation, so the file size can become quite large. A rough rule of thumb is that an average simulation of 100 000 rays results into a data file of about 25-30 MB in the hard disk. It scales upwards quite linearly. This is, however, a generalisation; some parameters may lead to a larger file size and some smaller. The maximum number of rays that can be stored is not restricted by the file size but rather RAM memory when the data is loaded back into Analysis window. This topic is discussed in Chapter 6, p.29. The corresponding simulation parameters are saved into a *simulation data parameter file* (\*.oup) which, if exchanged with other HALOPOINT 2.0 users, should always accompany the actual data (\*.out) file.

**Save Light Simulation Data... (Ctrl+Shift+L):** If crystal- and raypath images of the halo points are not to be examined in Analysis -window, the simulation data can be saved into a light simulation data file (\*.olt). In this saving format only a handful of data for each halo point is stored thus reducing the file size with a factor of ten compared to full data file. The size of a light data file of a simulation with one million rays is about 30 MB. Light data file enables full raypath analysis in Analysis -window, but the crystal- and raypath images can not be drawn. Similarly with the case of full data file, the corresponding simulation parameters are saved into a *light simulation data parameter file* (\*.olp) which, if exchanged with other HALOPOINT 2.0 users, should always accompany the actual light data (\*.olt) file.

**Save Only Visible Points (F1):** The *data*- and *light data* files contain data of all halo points on the celestial sphere - including those which fall outside the selected field of view. Sometimes this is not needed and causes an unnecessary increase in the file size. If this menu item is checked, only the halo points which are within the selected field of view (i.e. those which are actually plotted on the screen) are saved. This enables a way to examine a certain area of the sky without the burden of the uninteresting halo points.

**Save Simulation Image... :** The result of the next simulation run can be saved as an image in PNG (\*.png), bitmap (\*.bmp) or TIF (\*.tif) format. PNG and TIF -images are compressed losslessly and result in file sizes in the order of a few hundred kilobytes or less (depending on the screen pixel dimensions). PNG image format is understood by most *html* browsers and is suitable for web use. Bitmap image is largest in size, but may be preferred for some end uses.

**Simulation Description... :** Additional information about the simulation can be saved with the parameter files. By clicking this menu item a textbox appears where text can be entered. The text will be saved along with the rest of the parameters.

**Simulation Image Size... (Ctrl+I):** HALOPOINT 2.0 scales the simulation image according to the screen pixel dimensions. If, however, a smaller image size is needed, a customized pixel dimension of the longest side of the rectangular image can be given here. Size scaling of simulation images, which constitute basically of points of colour here and there, can be tricky in a photo editing software. Sometimes the end result is acceptable and sometimes it seems hard to get a decent scaled image. Therefore it may be convenient to give the correct pixel dimensions beforehand for i.e. web use so that no scaling is needed afterwards. Note, however, that if the entered image pixel dimensions are small, the number of rays and shade levels probably need to be fine tuned to obtain a satisfactory outcome.

**Exit:** Closes down the application.



## 4.2 Tools Menu

The first three items of the Tools menu open a dialog with which important features of HALOPOINT 2.0 can be controlled. These dialogs are presented in more detail in their respective subsections after the short list of all menu items:

**Multiple Scattering... (Ctrl+M):** This menu item opens the multiple scattering dialog. See detailed information in subsection 4.2.1.

**Ray Path Filtering... (Ctrl+F):** This menu item opens the raypath filtering dialog. See detailed information in subsection 4.2.2.

**Simulation Series... (Ctrl+B):** This menu item opens the simulation series control dialog. See detailed information in subsection 4.2.3.

**Lock Azimuthal Orientation... :** It has been suggested, that in certain situations, as for example on a snow- or ice sheet surface, the ice crystals might be coherently aligned. To test this hypothesis the azimuthal direction of the crystals of a given population have to be locked. This menu item opens a dialog where the azimuthal locking can be done by specifying a direction angle

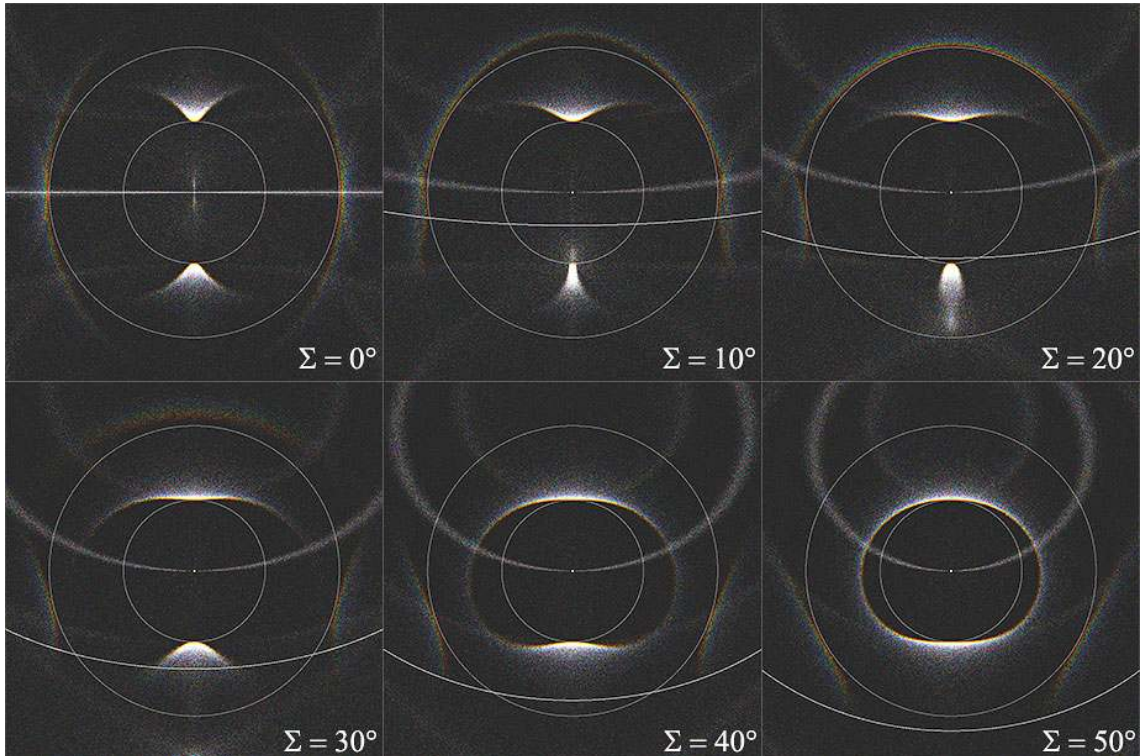


Figure 4.1: An example of the use of sun-concentric rings. In certain simulations, e.g. in publications, the rings give a scale to the simulated halos without overcluttering the image with additional halo points. Here the two drawn circles are  $22^\circ$  and  $46^\circ$  rings.

(in degrees). The azimuthal angle is the third Euler angle (system  $xxz$ ), which normally can obtain any value between  $0^\circ$  and  $360^\circ$ . As an indication of the azimuth locking, the locked populations numbers in the main window turn red.

**Draw Sun-concentric Rings** ► : Sometimes it may be desirable to have sun-concentric rings in a simulation as drawn circles rather than simulated rings. This way the interesting halos stand out better (without the additional halo points created by random populations) and yet a scale is provided to the image (see fig. 4.1). Select the rings from the submenu list; clicking an unchecked ring enables its drawing and clicking a checked ring removes the selection. At the bottom of the list two submenu items can be found, which allow selecting or deselecting all rings at one click.

**Transparency:** By clicking this menu item it is possible to create images, as in fig. 4.2, often seen in textbooks. A suitable projection is set by the parameters in the *Simulation View* -groupbox. The parameters for the leftmost image in fig 4.2 were: Fisheye equal area, focal length 12 mm, sensor size 34 x 34 mm and custom parameters  $a = 1.2$  and  $b = 1$ .

**Zenith/Nadir Marker:** If a marker is needed to indicate the zenith and/or nadir, this menu item has to be checked.

**Frame Image** ► : Simulation frames are selected from the submenu items, which are: *Lens Image Boundaries*, *Film Boundaries* and *Both*. Film boundaries are always rectangular. In case of fisheye lenses, however, the resulting image area may be circular or partly so. Therefore there is also possibility to choose a frame for the lens image as well. Clicking an unchecked frame enables its drawing and clicking a checked one removes the selection. The submenu item *Both* selects and deselects both frames.

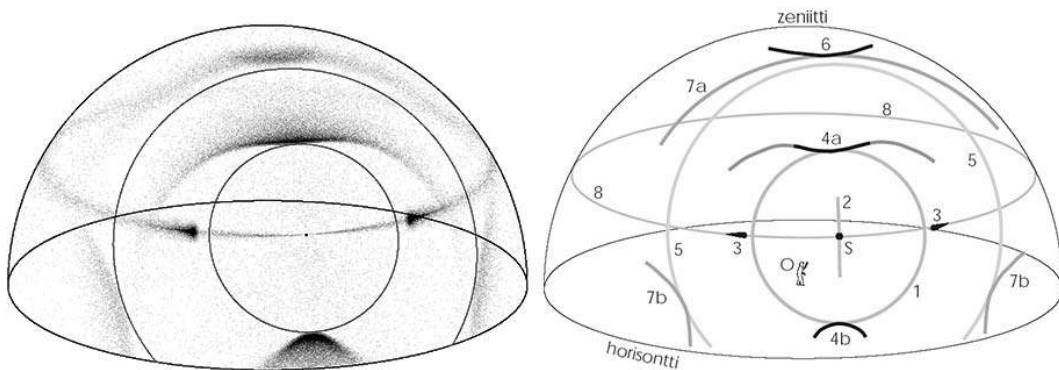


Figure 4.2: In textbooks one often finds general halo images such as the one on the right (drawing by Veikko Mäkelä). With transparency selected HALOPOINT 2.0 can produce similar images (on the left).

### 4.2.1 Multiple Scattering

In its normal mode only single scattering is considered in HALOPOINT 2.0. In case halo phenomena arising from multiple scattering [17] are of interest, it can be enabled and the parameters set in *Multiple Scattering Parameters* -dialog. There are two approaches for multiple scattering: a horizontally infinite slab, a layer, into which light enters, and a homogeneous space with a fixed probability of multiple scattering.

If a fixed probability of multiple scattering is chosen, it is simply set up by checking the checkbox *Use Fixed Multiple Scattering Probability* ① (the circled numbers refer to fig. 4.3) and entering a fixed probability of multiple scattering (a value between 0 and 1) into the textbox ②. This probability value is not a sole player though, since even with a multiple scattering probability of 1 you will get single scattering halos as well. In each ray encounter with a crystal a minimum bounding circle is drawn which comprises the ice crystal envelope on a plane perpendicular to the incident ray. A random point is chosen from within this circle which dictates whether the incoming ray hits or misses the crystal. In both cases - hit or miss - the "hit counter" is incremented, which explains why single scattering halos come through in all cases.

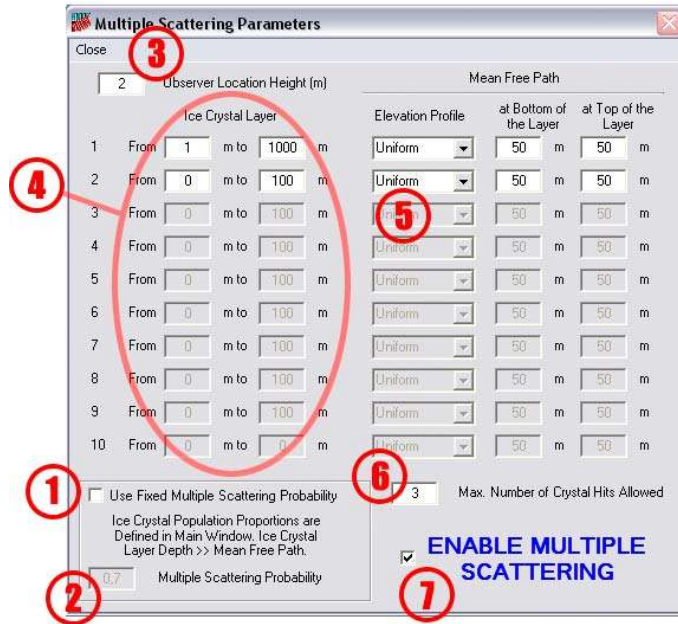


Figure 4.3: The multiple scattering parameters - dialog. Circled red numbers refer to circled numbers in text, where the corresponding parameter controls are discussed.

The second approach is more complicated, but may be a better approximation of what really goes on in the nature. Several parameters have to be defined, of which first is the *observer location height* ③ given in meters. Then, for each population that is active in main window, the ice crystal layer lower- and upper bounds are given ④ (in meters as well). Note, that all active populations undergo the same multiple scattering algorithm - it is not possible to exclude one population from the treatment. With the layer height- and mean free path parameters one can adjust the average number of crystal hits that take place within the layer. This basically enables one to approximate the true mean free path in a crystal layer by tweaking the parameters until a satisfactory outcome is obtained. Mean free path is the average distance travelled by a light ray (a photon) until a crystal is met.

A short description of the algorithm is as follows. A light ray enters the slab in

an angle determined by sun elevation. The slab is a combination of all ice crystal layers of active populations, see fig. 4.4. The layers can be partly interlaced, or even have a crystal free gap between the topmost and lowermost part of the slab. A cumulative distribution function (*cdf*) of the ray (photon) free distance is constructed, based on the given parameters. A uniform random number between 0 and 1 is drawn, and the inverse value of the *cdf* at that point tells the distance which the ray has travelled in the crystal layer, see fig. 4.5. At this point an ice crystal is met, and the new ray direction is determined by normal single scattering procedure. A new *cdf* is again constructed and this procedure is repeated until: 1) the ray intensity sinks below an allowed value (after only 1/10000 of the initial intensity of the ray is left after all crystal face encounters) or 2) a specified maximum number of crystals are met or 3) if the ray leaves the layer. A halo point is drawn every time the ray crosses the observers height, either on the way up or down.

Mean free path is specified for each crystal layer (all active populations). There are two distribution functions available ⑤: *uniform* and *linear*. In case uniform is selected, the mean free path is constant for the whole layer, and its value is given in "at Bottom of the Layer" -textbox only. The value in textbox "at Top of the Layer" does not play any role in uniform mean free path and can have any value. Both textboxes are used if linear mean free path profile is selected. In this case the mean free path of the layer changes in linear fashion from the top value to the bottom value. The longer the mean free path, the thinner the medium (less crystals).

The maximum number of allowed crystal hits is given in ⑥. Finally, multiple scattering is enabled by checking the checkbox ⑦, and closing the dialog from *Close* -menu item. When multiple scattering is enabled, it is indicated with a red text in

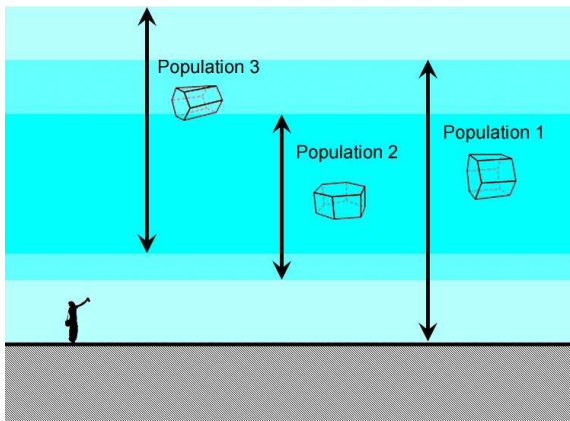


Figure 4.4: An independent ice crystal layer is defined for all active populations.

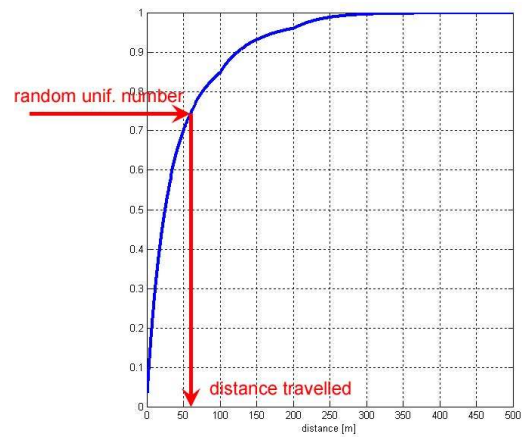


Figure 4.5: Inverse of the cumulative distribution function dictates the distance travelled by a ray before meeting a crystal.

main window.

Walt Tape kindly provided insight in the multiple scattering algorithm, which is greatly appreciated. Due to Walt's input the algorithm is improved from my original one, but all flaws it still may have are entirely my doings, not Walt's.

## 4.2.2 Ray Path Filtering

This dialog has two tabpages: *Specific Ray Path* and *General Ray Path* (see figs. 4.6 and 4.7). In the first one the wanted raypath is entered as an exact raypath, whereas in the second one the raypath can be specified more freely. At the bottom of both tabpages is a checkbox *Enable raypath filtering*, which can only be checked for one tabpage at a time. As an indication of activated raypath filtering a red text appears at the main window. raypath filtering can not be applied in case multiple scattering is enabled - it is only for single scattering halos.

If changes are made in the raypath filtering parameters, they must be updated by closing the dialog or clicking the menuitem *OK (Accept without closing)* or from the keyboard with F10 -key. After that the simulation can be started.

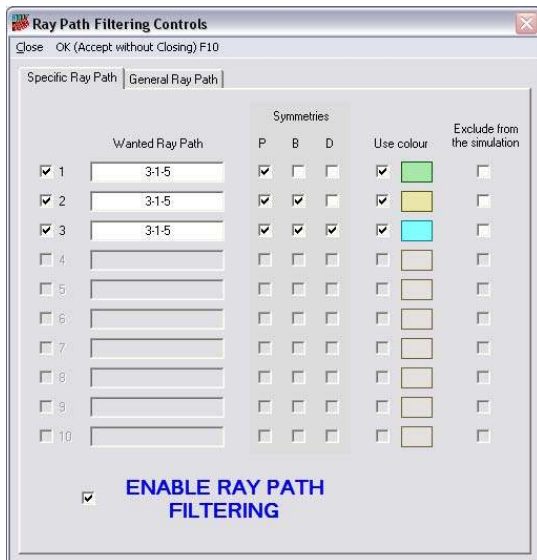


Figure 4.6: Tabpage: *Specific Ray Path*

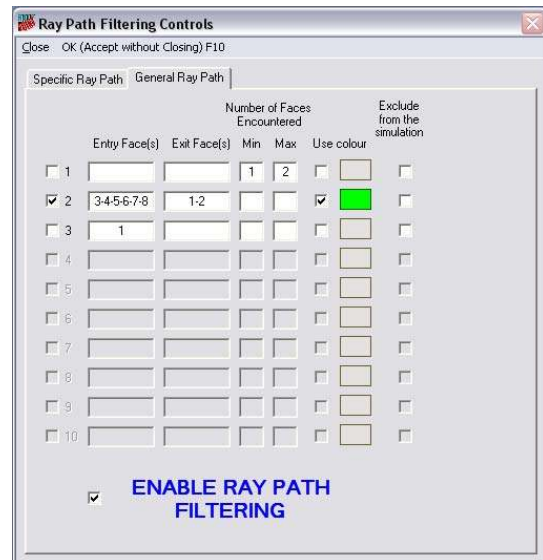


Figure 4.7: Tabpage: *General Ray Path*

**Specific Ray Path.** If there are several active populations, the ones to which raypath filtering is applied, must be checked. The raypath is entered in the form **3-1-5** or **1-23** etc., in other words with a line (" - ") between the faces. The program will prompt the user in case a false entry is given. Note, that if you want to clear the textbox completely, all characters, including spacebar, must be removed.

If none of the *symmetries* -checkboxes are checked, only the entered raypath will be filtered. In case of parhelia, for example, an entry **3-5** will find only 3-5 -raypaths but lets all the other parhelson raypaths off the hook (4-6, 5-7, 6-8, 7-3, 8-4 and the opposite directions 5-3, 6-4, 7-5, 8-6, 3-7 and 4-8). Therefore only 1/12th of all parhelson raypaths are caught with this entry, and only 1/6th from one side. Sometimes, when a specific raypath is studied, this functionality is suitable, but with the three available symmetries-checkboxes the range of filtered raypaths is widened markedly:

- ***Symmetries P***, where P stands for Prism. If this checkbox is checked, all identical raypaths, regardless of the first prism face encountered, are found. For parhelia this means, that all raypaths contributing to left parhelson are found if raypath entry is 3-5 and P-symmetry is checked. Right parhelson is found if raypath entry is changed to 3-7.
- ***Symmetries B***, where B stands for Basal. If this checkbox is checked, both basal faces are treated as equal. For raypath entry 3-1-5 (Wegener anthelic arc) only the loop starting towards west from 22° upper tangent arc (and closing in to 22° lower tangent arc from the east) is found unless B is checked. When checked, the reflections from basal face 2 are also considered, and thus the raypath 3-2-5 is also found.
- ***Symmetries D***, where D stands for Direction. This checkbox enables "directional symmetry", which allows equal raypaths, but coming from different directions, to be found. For parhelia this means, that raypaths 3-5 and 3-7 are equal, and so both parhelia are found.

An example of the effect of the three symmetries is illustrated in fig. 4.8. In all of the simulations an identical singly oriented column population was used. The raypath to

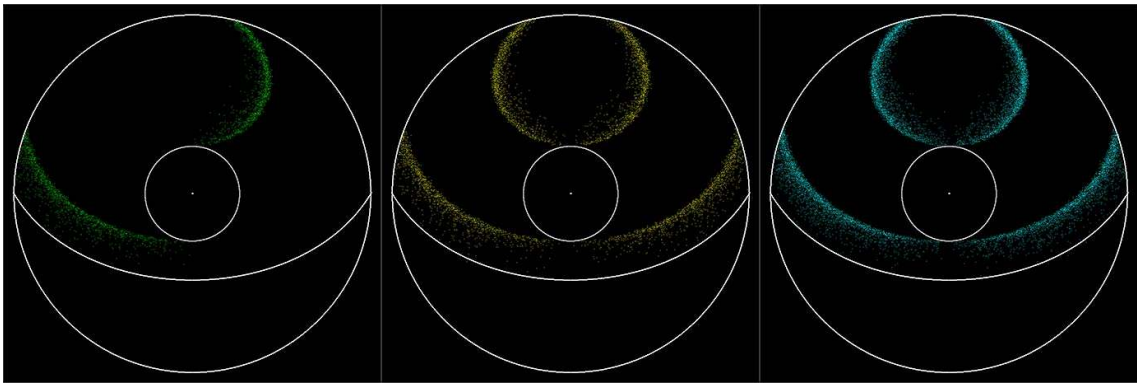


Figure 4.8: *An example of the use of symmetries in raypath filtering. The raypath for all three images was 3-1-5. The checked symmetries from the left were: P, PB and PBD. One can see, that in the rightmost image the Wegener anthelic arcs have twice as much points as in middle image due to checking of the directional symmetry.*

be filtered was 3-1-5 in each simulation. In the leftmost image only prism-symmetry was checked, in the middle image prism- and basal-symmetries were checked and in the rightmost image all symmetries-checkboxes were checked.

The filtered raypaths can be drawn with a user defined colour. The colour is chosen from the colour-button under *Use colour* text. The button opens a standard Windows colour dialog. To enable the use of the colour the checkbox has to be checked. Finally, the rightmost column has checkboxes, which control whether the filtered raypath is shown or excluded from the simulation.

**General Ray Path.** The filtering conditions are defined by giving the allowed entry and exit face(s) and maximum and minimum number of crystal faces encountered by the ray. Any number of these combinations can be defined (as in fig. 4.7). The entry and exit faces, in case there are many of them, are separated again with a " - " character. As in the previous tabpage, the colour of the halo points arising from the filtered raypaths can be defined by the user, as well as the exclusion -checkbox.

### 4.2.3 Simulation Series

In HALOPOINT 2.0 series of simulations can be executed automatically. These series are defined in *Simulation Series* -dialog (fig. 4.9).

First, before opening the dialog, the parameters of a simulation have to be defined in main window and saved to a parameter (\*.par) file. The parameter files associated with simulation data (\*.oup) and light simulation data (\*.olp) are also suitable formats for series parameters.

The steps for setting up a simulation series are:

1. Open a parameter file that has all basic settings for the simulation. This is done by pressing the button *Select Parameter File* and choosing the desired file. Once selected and accepted, the parameter filename is shown in the textbox below the button.
2. Choose the variable parameter and the population whose parameters will be varied from the top row. If the parameter to vary is something general and affects all populations, like *Sun Elevation* for instance, the selected population still has to be one of the active populations of the parameter file, although any active population will do. If on the other hand the variable parameter is such that it affects directly a selected population, like *Aspect Ratio mean* for instance, choose the correct population from the list.
3. Enter the values for the parameter sweep: starting value, stepsize and end value. Use the same units as in main window.
4. If another variable parameter is desired, repeat steps 2 and 3 for the second row. If not, leave the selected population of the second row to *None*.



5. Choose the desired output format. Checkmark in *Table* creates a html table with number of columns given in adjacent textbox. If two parameters are varied, the number of columns is determined by steps of the second variable and number of rows by first variable. In *Table* -output format the largest side of a simulation image is always 500 px, unless a custom simulation size is defined and activated in parameter file. If the *Single Images* -checkbox is checked, single, normal size images are saved into a folder.
6. If all steps 1-5 are taken care of, the series can be accepted to the list of series. It will be appended to the existing list.

Several series can be created with the procedure above. If a series needs modification, it can be returned back up for editing with the *Modify* -button (with red arrow). A row is selected by selecting the running serial number (#) of the row. Once editing is complete, it can be re-accepted with the *Accept to List* -button (with green arrow). A series row can be completely deleted by selecting the running serial number (#) and pressing the *Remove Selected Item* -button.

Below the list there are two textboxes which indicate the number of individual images that will be created if the whole list is run through, and also the number of rays to be considered to complete the list. Based on this information it is possible

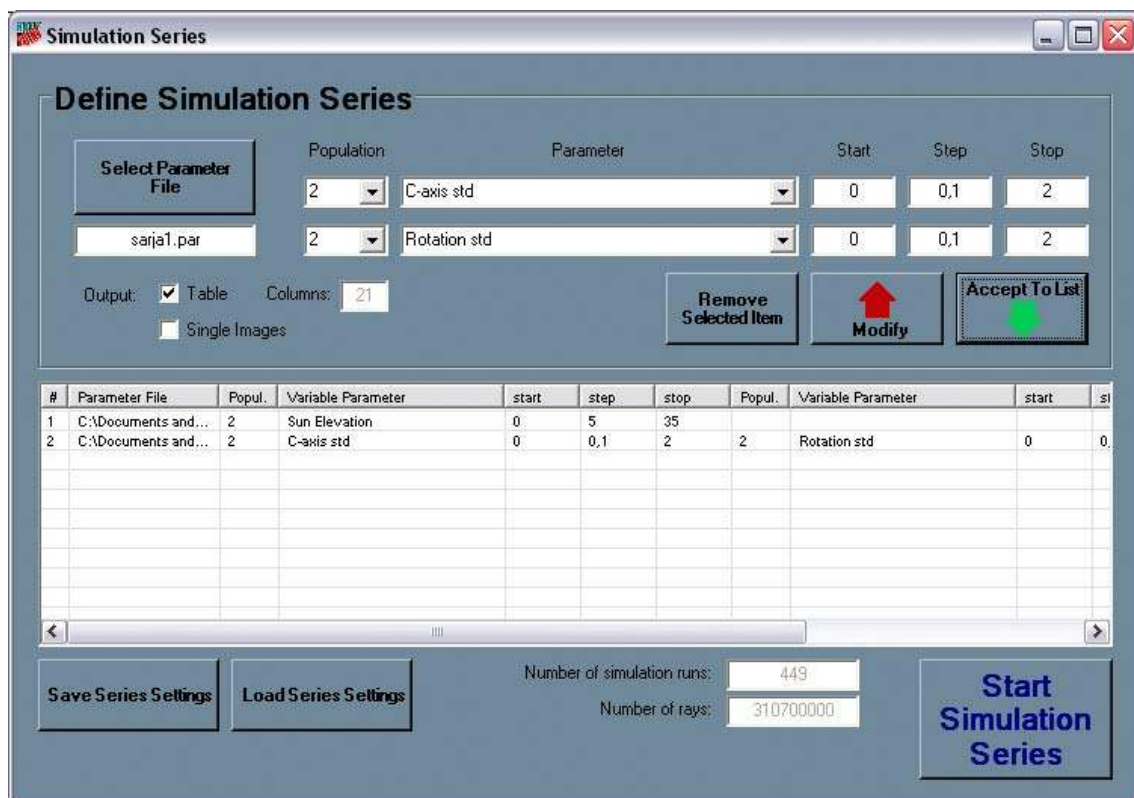


Figure 4.9: *The simulation series control -dialog.*



to evaluate the total time needed to complete the task.

It is possible to save current series settings and load it back again later. Only the accepted list of series goes into the file; none of the controls carry interesting information after the list is created. The files are of type *Simulation series files (\*.ser)*.

When everything is set, and *Start Simulation Series* -button is pressed, user is asked to specify a name for the series and the folder into which all images and such are created. Files and folders that will be created are:

- a log-file where time stamped events are logged (name.log)
- a folder for each series in the list named Series1, Series2... etc.
- simulation images in PNG-format. They are 500px -images if *Table* is checked and custom image size is not defined in parameter-file. Otherwise the images are either "full size" (determined by screen dimensions) or as defined in custom image size -parameter of the parameter-file.
- a html-file for each series in the list: name\_Series1.html, name\_Series2.html... etc., but only if *Table* is checked.

When the execution of the series starts, a status dialog appears, which is always on top of all windows (but can be minimized into the taskbar). In the status dialog the current status of the process is given along with an estimation of the time remaining until the end of the list is reached. There is also a *Stop Simulation Series Run* -button, which terminates the process. All so far completed simulations are preserved, but the simulation in progress is most likely lost. The stop-button may need a few clicks until its impact becomes effective. While the series is running it is advisable to leave the other dialogs and controls of HALOPOINT 2.0 alone to ensure smooth operation. The windows and dialogs can be minimized and one can continue working with other software. HALOPOINT 2.0 will, however, grab majority of the computer's processor time in its own use while simulation is being completed.



# 5

## Main window: Running a simulation

When all parameters controlling a simulation are set up, it is time to run the actual simulation. Simulation starts from the button *Start Simulation* (fig. 5.1). A simuplot -window (fig. 5.2) opens and the simulation starts to build up. User can minimize the simuplot -window, and continue working with other software while the simulation is in progress. A status bar is at the bottom of the simuplot -window, where information regarding the state of the simulation are given, namely number of rays considered so far and elapsed time. At bottom right there is also a progress bar, from which one can roughly see the percentage of rays completed so far.

Simulation can be stopped from the menu item *Stop Simulation* or from the keyboard with F10 -key. The close button at the top right corner of the window becomes available only when the simulation is stopped or finished. After the simulation is stopped, one can save the image from *File - Save as Image...* (Ctrl+S).

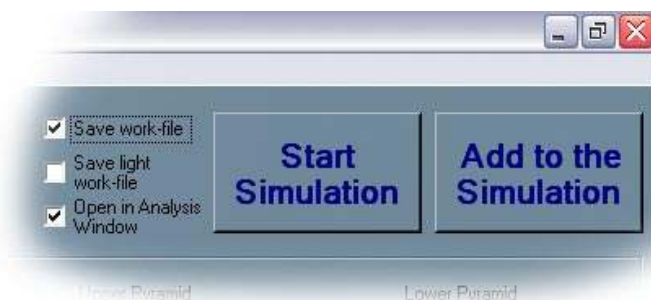


Figure 5.1: The top right portion of the main window.

If number of rays in the simulation seems insufficient, simulation can be continued with the *Add To the Simulation* -button in the main window. Do not close the simuplot -window if you wish to add points to the simulation! Otherwise the simulation that you already had, is lost. The simulation parameters may be altered before pressing the *Add To the Simulation* -button, but this should be avoided.

In Chapter 4 the main window menu items were presented, and they included saving *simulation data* and *light simulation data* -features. Unless the data files are meant to be distributed to other users of HALOPOINT 2.0 or the intention is to save the data for a longer period of time, it may be more convenient to use the quicker method provided in main window next to the *Start Simulation* -button. If *Save work-file*- or *Save light work-file* -checkbox is checked, a respective data file named `work.out` or

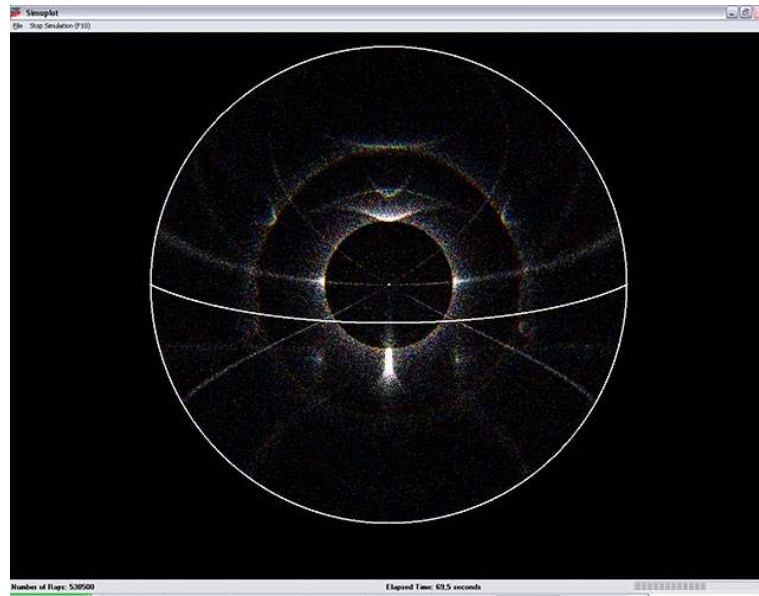


Figure 5.2: *The simuplot window, where the simulation image is shown.*

`work.olt` (along with corresponding parameter file) is saved into the folder where HALOPOINT 2.0 `.exe` and `.dll` -files are located. This work-file can be opened later manually in Analysis-window, or more conveniently, automatically right after the simulation is finished. Automatic loading into Analysis -window takes place if the *Open in Analysis-window* -checkbox is checked before starting the simulation.

# 6

## Analysis window

Analysis -window is opened from the main window (Menu item: Analysis - Open Analysis Window (Ctrl+A)). The saved simulation runs can be opened in Analysis -window and examined more carefully. The type of saved data format, simulation data or light simulation data, determines what analysis tools are available. In this chapter we first take a look at the general functions in Analysis -window and the actual analysis functions are presented in subsequent sections.

Saved data files can be loaded back into memory from **File - Open Simulation File...** (Ctrl+O) for the simulation data file and **Open Light Simulation File...** (Ctrl+L) for the light simulation data file. The data files consume a lot of RAM memory when loaded back into Analysis -window. Again, precise figures can not be given, since all simulations are different, but as a rough guideline one can assume that one (*full*) *simulation data* halo point requires about 0.5 - 1 kilobytes and in case of a *light simulation data* one halo point requires one third or even only one fifth of that. Therefore one can easily make simulations with millions of rays using light data format and up to a few millions with full data format. Usually a much lesser amount of rays should be sufficient for any analysis, however.

The number of internal reflections allowed is one parameter which affects the resulting file size. Therefore keep it as low as possible, at the same time keeping in mind the analysis process you want to perform. Also remember that there is a possibility to select the *Save Only Visible Points* (F1) -menu item in main window and adjust the field of view to cover only the part of the sky that is to be examined in Analysis -window. This way no halo points outside the wanted area are saved. For fainter halos (when a greater number of rays is needed) this is a good way to save RAM memory.

While the data is being loaded into memory, a percentage describing the status of the process is running at the bottom left corner of the Analysis window. When loading is ready, the simulation image appears on the screen and the number of loaded data points can be seen at the status bar of the Analysis -window. Furthermore, in status bar, at lower right corner, the celestial coordinates of the mouse position are displayed. The azimuth is, as mentioned before, relative to the Sun position (Sun is at azimuth of 0°).

The simulation image can be saved into file similarly as in main- and simuplot -windows. Also, a loading of a image is provided, but note that the loaded image is only for viewing. No functional features of Analysis -window are supported.

## 6.1 Tools Menu

**Refresh Simu Drawing (F5).** The simulation image can be refreshed at any time from this menu item (or from keyboard with F5 -key). HALOPOINT 2.0 grabs the current image from the picturebox prior to any function being activated. At (hopefully) rare occasions it might happen, that the grabbed image gets interfered by an external software window, in which case pressing F5 helps.

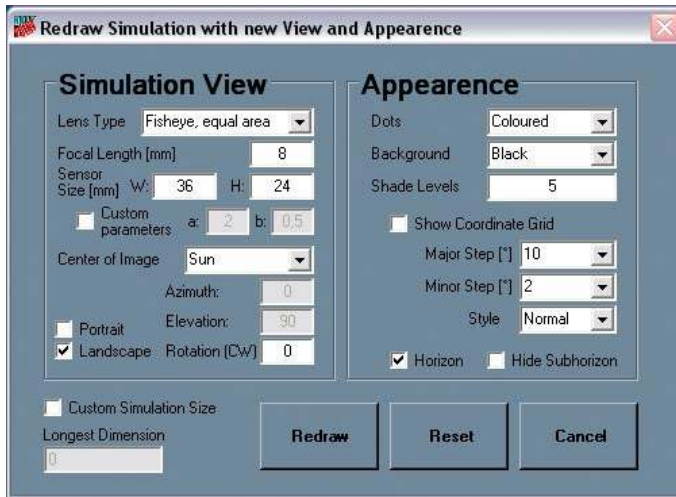


Figure 6.1: *The Redraw -dialog.*

**Redraw With New Viewangle... (Ctrl+R).** This menu item opens a dialog (fig. 6.1) with which the loaded simulation can be redrawn from any direction and with any field of view, since all halo points of the simulation are stored (unless *Save Only Visible Points* - was selected in main window). The appearance, i.e. shade levels, simulation colours etc., as well as pixel dimensions can also be changed with this dialog. The changes are accepted with *Redraw* -button. Depend-

ing on the number of rays in the image, the drawing can take a few seconds to complete. *Reset* -button restores the original settings that were selected when the simulation was originally run. With *Cancel* -button the dialog can be closed without making any changes.

**Angle Measurement... (Ctrl+A).** When selected, a line can be drawn with a mouse (by keeping the left button of the mouse pressed). When the mouse button is released, the distance between the endpoints of the line, as well as azimuth- and elevation difference, are displayed on the screen, see fig. 6.2. New measurements can be made until the measuring operation is stopped. Stopping can be done by clicking the right mouse button and selecting the appearing *Stop Measurement* -text.

**Zenith/Nadir Marker.** A marker can be drawn to mark the location of the zenith and nadir, if selected.

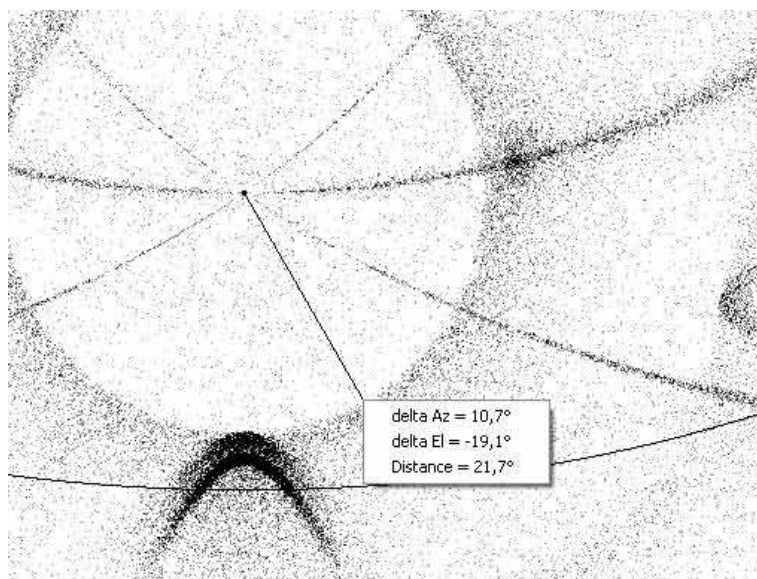


Figure 6.2: A measurement can be made with a mouse.

**Frame Image ►.** Simulation frames are selected from the submenu items, which are: *Lens Image Boundaries*, *Film Boundaries* and *Both*. Film boundaries are always rectangular. In case of fisheye lenses, however, the resulting image area may be circular or partly so. Therefore there is also possibility to choose a frame for the lens image as well. Clicking an unchecked frame enables its drawing and clicking a checked one removes the selection. The submenu item *Both* selects and deselects both frames.

## 6.2 Ray Path Analysis

The main reason why Analysis -window exists is the possibility to study the raypaths responsible for the dots in the simulation image. There are two functional features in Ray Path Analysis -menu: a possibility to see the raypaths of a certain area in text format and a possibility to draw a raypath through a crystal for any halo point in the image. Raypath analysis can be performed to both data formats, but the crystal+raypath drawings are only possible for the full data file. Therefore it is advisable to use light data file if there is no intention to look at the raypath images, since it consumes considerably less RAM memory than the full data file format. On the other hand, it is quite fun to see the raypath images, and besides fun, perhaps it is useful as well.

Raypaths are expressed as sequences of numbers, each number representing the face that was encountered by a ray on its way. The first and last number represent the entry- and exit faces and the numbers in between the internal reflections.

**Define Analysis Area... (F1).** First step of the raypath analysis is to define the area from which the halo points for the analysis are taken. This menu item opens a dialog, where there are several options for selecting the area (see fig. 6.3), described in the list below:

- **Using Mouse.** Draw a rectangle with the mouse (by keeping the left button of the mouse pressed). If the rectangle is not in a correct place, you can draw a new one. After a rectangle is in place, press the right button of the mouse and select one of the two options: *Accept area* or *Cancel Drawing*. The first one opens a Raypath Report -window (in a few seconds time - depending on the number of halo points in the rectangle) and the second one dismisses the rectangle and cancels the raypath analysis.
- **Using Celestial Coordinates.** A rectangle on a sphere can be drawn by specifying its four corners using azimuth and elevation in degrees. Two sides of the rectangle will lie on meridians (vertical, azimuth lines) and two on lines of elevation, see example on fig. 6.4. After *Go!* -button is pressed, the Raypath Report -window opens.
- **Visible Image.** This option uses all halo points from the visible image area. After *Go!* -button is pressed, the Raypath Report -window opens, which may take a while if the number of halo points in the image is large.
- **All Points.** This option uses all halo points of the loaded file, no matter

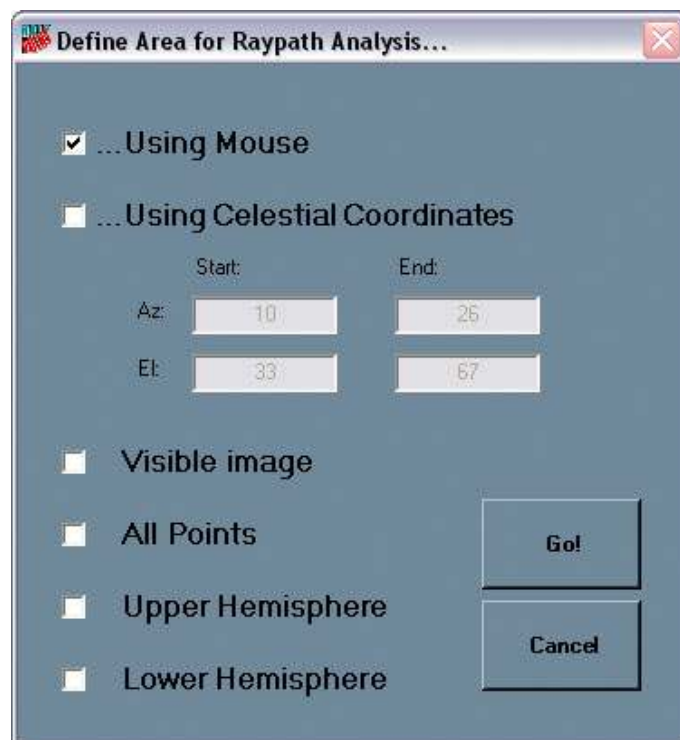


Figure 6.3: In this dialog the area for the raypath analysis is chosen.



where in the sky they are located at. This option should be used only when the total number of rays in the simulation is not very large.

- ***Upper Hemisphere.*** All halopoints from the upper hemisphere are taken for the analysis.
- ***Lower Hemisphere.*** All halopoints from the lower hemisphere are taken for the analysis.

The Raypath Report -window (fig. 6.5) opens after the analysis area has been selected and accepted. In the topmost textbox of the Raypath Report -window a summary of the populations used for the simulation is shown. Essential information from main window groupboxes *Ice Crystal Population Parameters* and *Prism Face Distances* is gathered. Also the number of halo points in the analysis is shown at the top of the textbox.

Below the summary-textbox is the main area where the raypaths are shown. They are grouped by population and arranged by count. The three columns on the right show details for each raypath: total number of occurrences (within the selected analysis area), percentage of all population rays and percentage of all analysis rays (all populations included).

As a default the raypaths are reduced (simplified) versions of the original raypath, the so-called subtypes. Here the naming convention and reduction procedure introduced by Tape [18] is followed. As Tape explains, the reducing is done by first deleting pairs of internal basal face reflections. Then any remaining internal basal face reflection is placed in the second position of the raypath. For population whose rotation angle is not restricted by any means the prism faces are renumbered so that first prism face hit by the ray is face 3. Finally, pairs of opposite prism faces

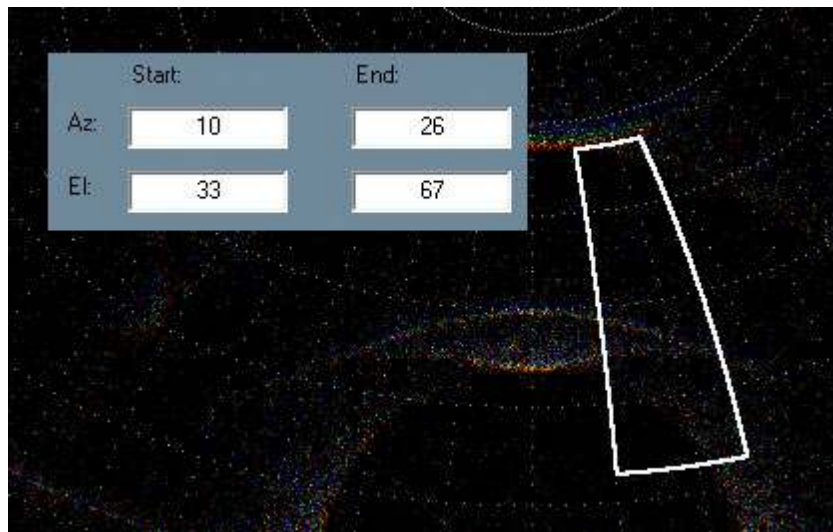


Figure 6.4: The analysis area can be defined using azimuth- and elevation.

are deleted, as they do not change the direction of the ray. This way the original raypath is reduced into a subtype (see more detailed description with lots of examples in Tape's book [18]). One could go further, as is explained by Tape, but in HALOPOINT 2.0 the subtyping is as far as the raypath is processed.

The *Show Raypaths Reduced into Subtypes* -checkbox is initially checked when the window opens. If one wants to examine the original raypaths which have not undergone the reduction process described above, the list can be updated by checking the *Show Original Raypaths* -checkbox and pressing the *Raypath list below* -button. Furthermore, to avoid cluttering the list with less significant raypaths, one can define a percentage the raypath is required to have contributed into the whole simulation in order to be shown in the list.

The raypath analysis can be saved into a *html* -file. Press *Complete Report* -button, and a standard Windows save dialog opens, where a name and path are given. An example of the resulting *html* -file is shown in fig. 6.6. The numeric data is taken straight from the *Raypath Report* -window, so before saving remember to refresh the data in the window by pressing the *Raypath list Below* -button in order to make the possible changes effective. Also note, that the simulation image from the Analysis -window is captured at the time of saving, so whatever you have in there goes into the *html*-file (as a separate *png*-image). Unfortunately the area selection drawing is lost if the simulation image is, for example, refreshed. Therefore make the saving right away if you intend to do so, before the rectangular area selection drawing is lost for some reason.

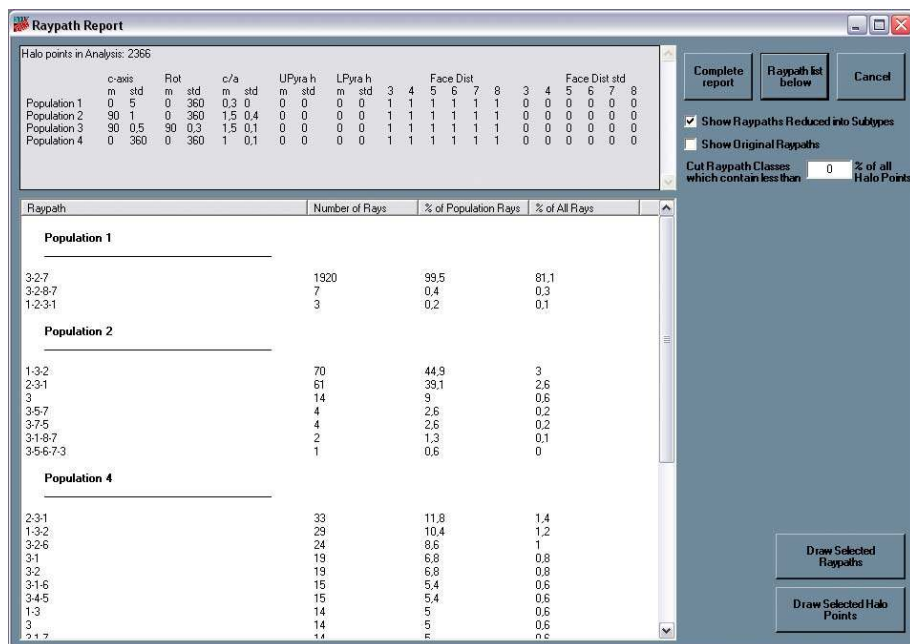


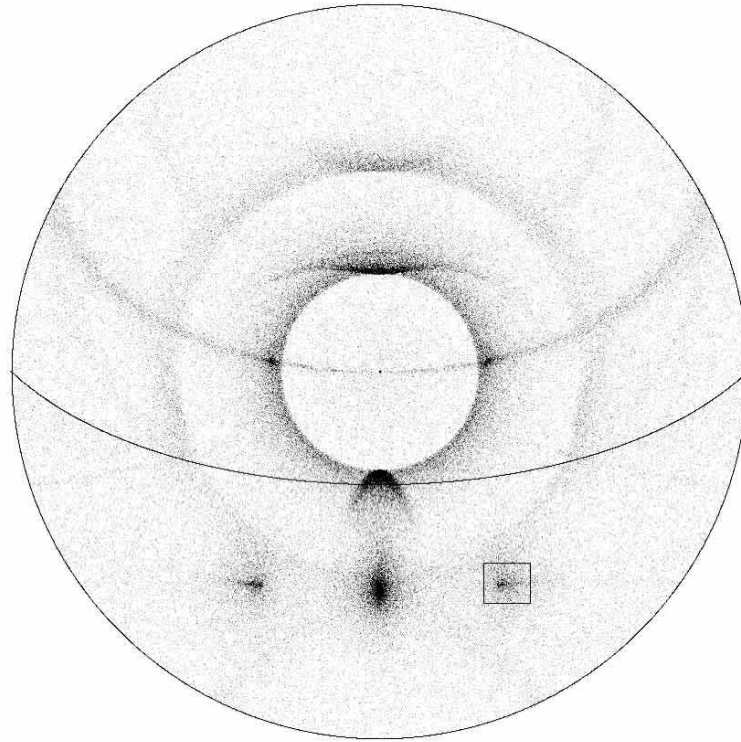
Figure 6.5: The *Raypath Report* -window.

## Raypath Analysis

HaloPoint 2.0

3.1.2010

Halo points in Analysis: 2366



Raypaths are reduced into subtypes

Raypaths which contribute less than 1 % of all halo points are not shown.

	c-axis		Rotation		Aspect Ratio c/a		Upper Pyramid Height		Lower Pyramid Height		Face Distances						Deviation of Face Distances					
	mean	std	mean	std	mean	std	mean	std	mean	std	3	4	5	6	7	8	3	4	5	6	7	8
Population 1	0	5	0	360	0,3	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
Population 2	90	1	0	360	1,5	0,4	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
Population 3	90	0,5	90	0,3	1,5	0,1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
Population 4	0	360	0	360	1	0,1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0

Raypath	Number of Rays	% of Population Rays	% of All Rays
<b>Population 1</b>			
3-2-7	1920	99,5	81,1
<b>Population 2</b>			
1-3-2	70	44,9	3
2-3-1	61	39,1	2,6
<b>Population 4</b>			
2-3-1	33	11,8	1,4
1-3-2	29	10,4	1,2
3-2-6	24	8,6	1

Figure 6.6: An example of the complete raypath html -report.

At bottom right of the Raypath Report -window there are two buttons. The upper one, *Draw Selected Raypath*, opens the Raypath Visualisation -window and will be presented later in the paragraph concerning the crystal- and raypath drawing (the paragraph below in boldface type: Choose a Single Point). The Lower button, *Draw Selected Halo Points*, draws the halo points of the selected raypaths into the Analysis -window.

The raypath selection is made by clicking a raypath in the list. In a standard Windows style one can make multiple selections using Shift- and Ctrl -keys; Making a multiple selection with Shift-key is done by first clicking on the first wanted raypath, then holding the Shift key down while clicking on another raypath. As a result all raypaths between these two will be selected. With Ctrl-key you can select and deselect individual raypaths - every click appends or subtracts from the selection. Note, that the lines containing no raypath can be selected, but result in a prompt to deselect them before drawing can be completed.

With this feature it is possible to examine what parts of the halos do certain raypaths contribute to, or simply to see where the halo points of a certain raypath pile up. If the total number of rays of the selected raypaths is large, the drawing takes a while. When the *Draw Selected Halo Points* -button is pressed, the Analysis -window comes forward and displays the halo points of the selected raypaths. The Raypath Report -window can be brought into front again with a spacebar -key.

**Choose a Single Point (F2).** The crystal+raypath image can be drawn for all loaded halo points. User simply clicks the simulation image, and the crystal+raypath image corresponding to the halo point at the clicked location is drawn. The halo point selection is activated by selecting this menu item or from the keyboard with the F2 -key. The (invisible) area around mouse tip is three times three pixels to ease selecting a tiny pixel from the image. If the selection is off target, the user will be notified and a new mouse click can be done immediately.

If there is at least one halo point within the 3x3 pixel area, a Raypath Visualisation window opens (fig. 6.7) and displays the ice crystal and the raypath of the selected point. Typically there are many halo points within the 3x3 selection area, which all become available in the Raypath Visualisation -window. There can be a halo point in every pixel of the selection matrix, and in addition each pixel may have accumulated light from many different raypaths. Therefore it is not unusual to have tens, even a few hundred, crystal+raypath images piled up. The different halo points can be accessed with the *Show number* -dropdown list. Once the cursor is in the list one can use the mouse wheel to scroll through the images.

In Raypath Visualisation -window the appearance of the crystal edges and ray segments can be adjusted. The view angle can be changed with sliders as in main window or by moving the mouse on top of the image. The initial orientation of the crystal is not the same as in main window. The crystal is exactly in the orientation

it was at the time the light ray hit it.

In the topmost textbox the true (not reduced) raypath is shown to assist in realisation of a possibly cumbersome raypath. In the drawing a black dot marks a ray - face encounter that is directly visible, whereas an open circle marks an encounter that is behind another face. When a ray is inside the crystal or behind an obstacle it is drawn with dashed line and otherwise with a solid black line. An arrowhead indicates the direction of the ray. Furthermore, the length of the entering and exiting ray can be adjusted with the sliders.

The other features of the Raypath Visualisation -window include an option to turn the one-point perspective off or on (initially on) and adjust the perspective strength with a scale factor (range 4 to 50). The smaller the figure, the more powerful the perspective is. The initial value is 10. One can also enable the raypath segment intensities and remove the raypath completely and observe only the crystals.

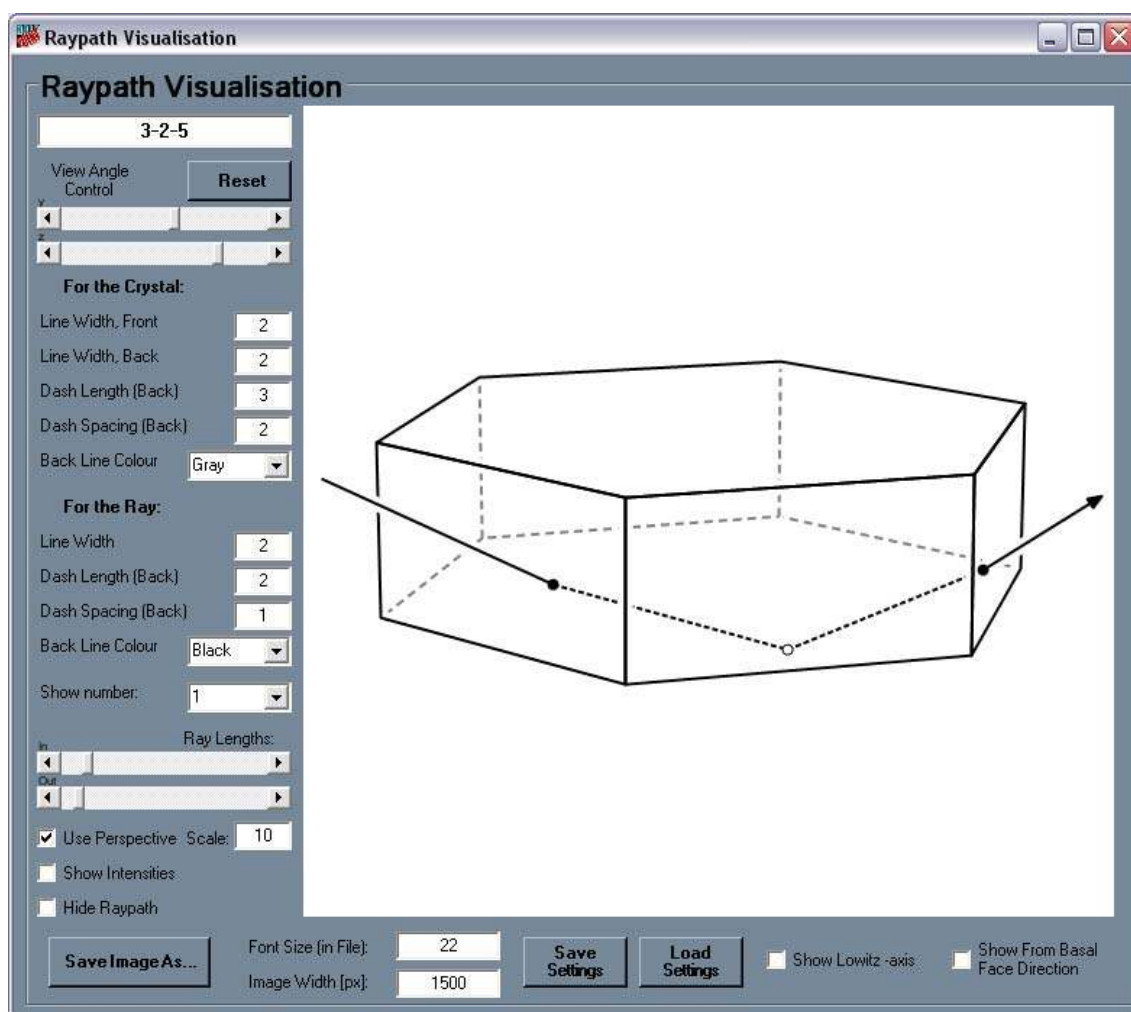


Figure 6.7: *Raypath Visualisation -window.*

The image can be saved (PNG, Bitmap or TIF) into a file. The resulting file pixel width can be given and in case the intensities are shown, the font size of the intensity numbers can be given. The saved image will be a scaled duplicate of the image shown in Raypath Visualisation -window.

The settings of the Raypath Visualisation -window can be saved into a \*.cim -file and loaded back. This enables the user to define suitable drawing settings for different purposes. The window remembers its last settings and loads them into use when the window is opened the next time.

The two last checkboxes help in visualisation of the raypath. First one is *Show Lowitz -axis*, which brings the Lowitz axis into the image. It is for illustrative purposes only, and does not meet the requirements of publications. The last checkbox, *Show From Basal Face Direction*, views the crystal from the other end, as if camera was slid along the c-axis. This way the raypath angles become clearly discernible, helping to understand the lights paths through crystal.

# Appendix A

## About mapping functions of camera lenses

In HALOPOINT 2.0 the user can influence the perspective projection of the image in several ways: selecting a focal length, a mapping function and on top of these utilize the two additional perspective projection parameters (custom parameters  $a$  and  $b$  in Simulation View- groupbox) if needed. Typically five ideal radial projection functions are mentioned when talking about camera lens design and mathematics to map an angular distance  $\theta$  from the optical axis onto a distance  $r$  from the image center: perspective-, stereographic-, equidistant-, equal area- (also known as equi-solid angle-) and orthogonal projection.

Ideally a rectilinear camera lens follows a perspective camera projection<sup>i</sup>, in which straight lines remain straight, and is described with a formula

$$r = f \tan(\theta), \quad (\text{A.1})$$

where  $f$  is the focal length and  $\theta$  the angle between the optical axis and the incoming ray. If the optical axis coincides with the center of the film or sensor,  $r$  is the distance from the center in millimeters. The two fisheye-projections available in HALOPOINT 2.0 are described with the following equations:

$$r = 2f \sin(\theta/2) \quad \text{equal area projection} \quad (\text{A.2})$$

$$r = f\theta \quad \text{equidistant projection} \quad (\text{A.3})$$

Majority of the SLR (and DSLR) fisheye lenses are of equal area -type, or at least that has been the starting point of the design. This lens is suitable for area measurements, since the ratio of the incident solid angle and its resulting area in the image is constant. In equidistant projection, which is much rarer implementation of a fisheye projection in commercial lenses, the angles of incidence are translated linearly into radial distances in the image. The real lenses, however, do not follow exactly the ideal projection model and there are differences even between the individual lenses of the same manufacturer's model. Manufacturers do not provide charts

---

<sup>i</sup>also known as pinhole- or gnomonic projection

or any measurement results of the mapping function of their lenses.

Calibration of lenses is a widely discussed problem, see e.g. [19, 20, 21]. Many applications, such as machine vision etc., need a precisely calibrated lens to perform their task safely and with required precision. Typically some geometric shapes or chessboard -type of calibration patterns are used, but these involve laborous distance measurements between the object and lens. More suitable calibration pattern for halo enthusiasts is a starfield on a clear night.

The radial projection model can be altered by modifying the coefficients of the function or supplementing it with distortion terms. A more generic approach, suitable for all fisheye -projections, is to use a polynomial function [22]. In HALOPOINT 2.0 deviations in the order of  $0.5^\circ$  from the true perspective function are quite acceptable, and therefore the user is not offered elaborate correction tools such as the polynomial coefficients. Modification of the perspective model coefficients (parameters  $a$  and  $b$ ) should be sufficient to obtain uncertainty of  $0.5^\circ$ . Precision measurements are an other issue, but for these the precise model of the lens is not needed anyway. As long as the sun/moon disk is visible in the image it can be superimposed with a starfield photo (taken with the same camera+lens combination, of course), and using the stars in the vicinity of the halo of interest its radial distance from sun/moon can be calculated.

The projection produced by commercial non-fisheye wide-angle lenses, called rectilinear in HALOPOINT 2.0, is typically very close to the ideal model (eq. (A.1)). Therefore the parameters  $a$  and  $b$  do not affect the perspective projection of a rectilinear lens at all, even if *Custom parameters* -checkbox is checked. For lens type *Fisheye, equal area* both parameters are in use and are defined as:

$$r = af \sin(b\theta). \quad (\text{A.4})$$

Ideal coefficients are  $a = 2$  and  $b = 0.5$ , as in eq. (A.2). These values are used if *Custom parameters* -checkbox is not checked. For lens type *Fisheye, equidistant* only parameter  $a$  is in use and is defined as:

$$r = af\theta. \quad (\text{A.5})$$

The ideal model coefficient is  $a = 1$ , as in eq. (A.3). This value is used if *Custom parameters* -checkbox is not checked.

What coefficients should one give for  $a$  and  $b$  then? For most of the time the default values should be sufficient. However, for some lenses the deviations from the ideal model seem to be surprisingly large, eg. **Peleng 8mm f3.5 Fisheye Lens** and **AF Fisheye Nikkor 10.5mm f/2.8 DX**, to mention a few. Information about the subject is hard to find, but one interesting graph is shown at Michel Thoby's webpage [23], which illustrates the deviations of radial projection from the ideal model for several common fisheye lenses (fig. A.1). Coefficients for a few lenses can



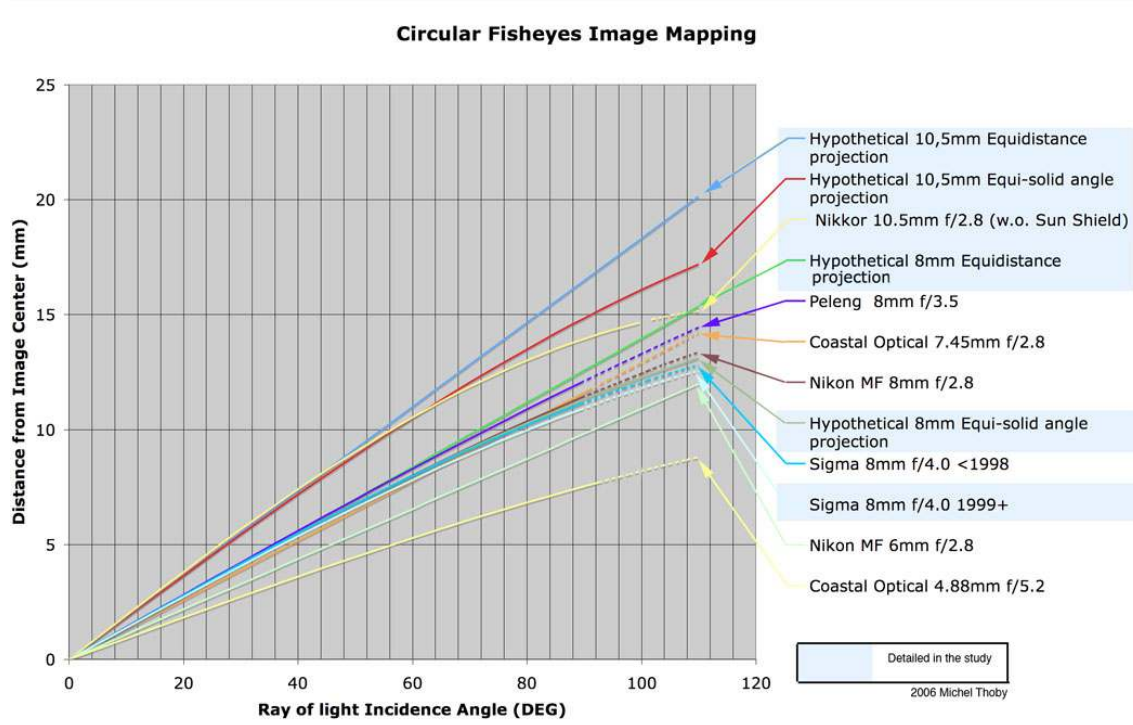


Figure A.1: *Radial mapping functions of several lenses from Michel Thoby's webpage [23].*

be found from different sources, but their correctness is not guaranteed since the measurement procedures are not explained.

I also did a measurement for two designs of Sigma 8mm f/4.0 Fisheye (pre 1998 and 1999 forward) to familiarize myself with the procedure. At first I estimated the centerpoint of the circular image (the lenses are not exactly collimated to assure that the image sensor and optical axis would coincide). Then, using starfield photos taken with these lenses I could determine the true radial mapping dependence between the distance (in mm) and angle. I also knew that the focal length of both of the lenses is actually 7.82 mm, so with these pieces of information I was able to find a best fit for coefficients of the eq. (A.4) using method of least squares. The uncertainty of my measurements is about  $0.6^\circ$  mainly because of difficulties in determining the center of the optical axis. To ease this problem a slightly overexposed (+1-2EV) photograph where the edge of the circular image is clearly visible, is useful. Additionally one could make the radial mapping measurement in several directions from the center of the sensor; this would also reveal the true position of the optical axis.

In the short list below estimations of the coefficients of a few lenses are given (collected from various open sources):

- AF Fisheye Nikkor 10.5mm f/2.8 DX:  $a = 1.47$ ,  $b = 0.713$  and  $f =$

10.58mm.

- Sigma 8mm f/4.0 AF EX:  $a = 1.88$ ,  $b = 0.54$  and  $f = 7.82\text{mm}$ , (my own measurement:  $a = 1.93$ ,  $b = 0.52$ ).
- Sigma 8mm f/4.0 (pre 1999):  $a = 1.88$ ,  $b = 0.54$  and  $f = 7.82\text{mm}$ , (my own measurement:  $a = 1.88$ ,  $b = 0.53$ ).

Reader is encouraged to measure his/her own lenses, if not for nothing else but pure fun. A good quality starfield photo is essentially all that is required. The process could be automated as well, which might be next programming project...

The parameters  $a$  and  $b$  enable also an ortographic projection when they both are 1, see fig. A.2 below.

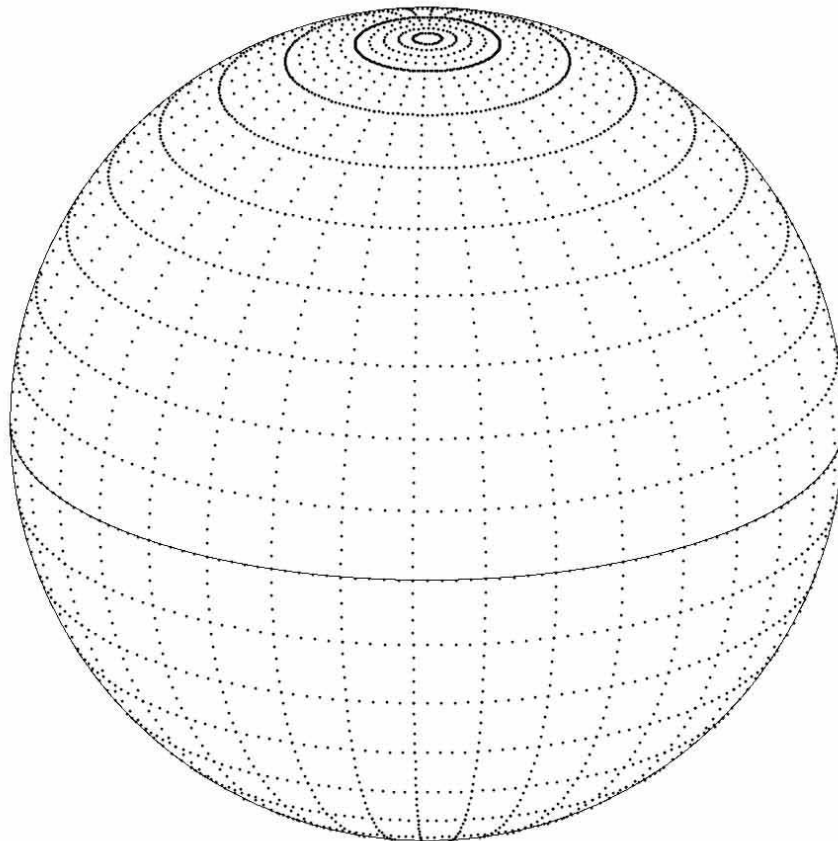


Figure A.2: *Ortographic projection is enabled, when  $a = 1$  and  $b = 1$ .*

# Bibliography

- [1] Greenler R. G. and Mallmann A. J., "Circumscribed haloes - Circumscribed halos which appear around the sun are simulated by a computer treatment of a simple model", *Science*, **176**(4031):128-131, April 1972.
- [2] Greenler R. G., Mallmann A. J., Mueller J. R. and Romito R., "Form and Origin of the Parry Arcs - Computer simulation shows how arcs of light above and below the sun are produced.", *Science*, **195**(4276):360-367, January 1977.
- [3] Mueller J. R., Greenler R. G. and Mallmann A. J., "Arcs of Lowitz", *Journal of the Optical Society of America*, **69**(8):1103-1106, 1979.
- [4] Greenler R. G., *Rainbows, Haloes, and Glories*, Cambridge University Press, Cambridge, 1980
- [5] Pattloch F. and Tränkle E., "Monte Carlo simulation and analysis of halo phenomena", *Journal of the Optical Society of America A* **1**(5):520-526, 1984.
- [6] Greenler R.G. and Tränkle E., "Anthelic arcs from airborne ice crystals", *Nature* **311**:339-343, 1984.
- [7] Tape W., *Atmospheric Halos*, Antarctic Research Series, Vol. 64, American Geophysical Union, Washington, 1994.
- [8] Tape W. and Moilanen J., *Atmospheric Halos and the Search for Angle X*, American Geophysical Union, Washington D.C., 2006.
- [9] Gislén L. and Mattsson J. O., "Observations and simulations of some divergent-light halos", *Applied Optics* **42**(21):4269-4279, 2003.
- [10] Gislén, L., "Procedure for simulating divergent Light Halos", *Applied Optics* **42**(33):6559-6563, 2003.
- [11] Sillanpää M., Moilanen J., Riikonen M. and Pekkola M., "Blue spot on the parhelic circle", *Applied Optics* **40**(30):5275-5279, 2001.
- [12] Cowley L., "Atmospheric Optics", [online], Available at: <http://www.atoptics.co.uk/>.
- [13] Finnish Astronomical Association URSA, "Ice Crystal Halos", [online], Available at: <http://www.ursa.fi/blogit/haloreports/index.php>
- [14] Ruoskanen J., "Halos", [online], Available at: <http://www.saunalahti.fi/~jukkrus/halos/halos.html>
- [15] Warren, S. G., "Optical constants of ice from the ultraviolet to the microwave", *Applied Optics*, **23**(8):1206-1225, 1984.
- [16] Bruton D., "Approximate RGB values for Visible Wavelengths, FORTRAN Code", [online], Available at: <http://www.physics.sfasu.edu/astro/color/spectra.html>.
- [17] Tränkle E. and Greenler R. G., "Multiple scattering effects in halo phenomena", *Journal of the Optical Society of America A*, **4**(3):591-599, 1987.
- [18] Tape W., *Atmospheric Halos, Appendix E (pp. 123-131)*, Antarctic Research Series, Vol. 64, American Geophysical Union, Washington, 1994.
- [19] Kumler J. J. and Bauer M. L., "Fish-eye lens designs and their relative performance", Proceedings of SPIE, **4093**(360), San Diego, CA, USA, August 2000.
- [20] Shah S. and Aggarwal J. K., "Intrinsic parameter calibration procedure for a (high-distortion) fish-eye lens camera with distortion model and accuracy estimation", *Pattern Recognition*, **29**(11):1775-1788, 1996.
- [21] Schneider D., Schwalbe E. and Maas H.-G., "Validation of geometric models for fisheye lenses", *ISPRS Journal of Photogrammetry and Remote Sensing*, **64**:259-266, 2009.
- [22] Kannala J. and Brandt S. S., "A Generic Camera Model and Calibration Method for Conventional, Wide-Angle and Fish-Eye Lenses", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(8):1335-1340, 2006.
- [23] Thoby M., "About Panography", [online], available at: <http://michel.thoby.free.fr/>.