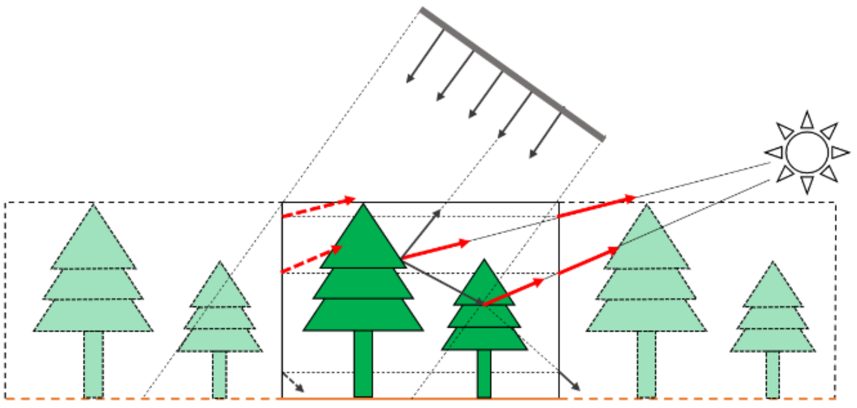
LESS User's Manual

Version 2.0



By Jianbo Qi

2022/11/4

Table of Contents

1. Introduction 4
2. Fundamentals of LESS 4
    2.1. Forward photon tracing (FPT) 4
        2.1.1. Photon tracing 4
        2.1.2. Virtual photon 6
    2.2. Backward path tracing (BPT) 6
    2.3. BPT in simulating thermal infrared brightness temperature (BT) 7
3. Installation 8
4. Graphic User Interface (GUI) 8
    4.1. Preview Panel 8
    4.2. Parameter Control Panel 9
        4.2.1. Sensor 9
        4.2.2. Observation 10
        4.2.3. Illumination & Atmosphere 10
        4.2.4. Optical Database 10
        4.2.5. Terrain 11
        4.2.6. Objects 11
    4.3. Progress Panel 12
    4.4. Menu Panel 12
        4.4.1. File menu 12
        4.4.2. Run menu 12
        4.4.3. Tools menu 13
        4.4.4. Display menu 13
        4.4.5. Process menu 13
        4.4.6. Help menu 13
5. Practical tutorial 13
    5.1. Major functionalities 13
        5.1.1. FPAR simulations 13
        5.1.2. LiDAR simulations 14
        5.1.3. Simulations of Multi-Spectral Image 20
        5.1.4. Fisheye camera simulation 25
        5.1.5. Simulation of Multi-angle Reflectance 26
        5.1.6. Directional Gap Fraction simulation 30
        5.1.7. Thermal Infrared Image Simulation 31
        5.1.8. Layered Illumination/Shaded Leaf Area Ratio* 32
        5.1.9. TRAC simulation* 33
        5.1.10. Fisheye image transmittance simulation* 34
        5.1.11. Light simulation* 35
        5.1.12. Re-collision probability simulation 36
        5.1.13. Cloudy medium expressed leaf 37
    5.2. Utility 38
        5.2.1. LAI calculation tool 38

5.2.2. Batch processing tool........ 38
5.2.3. Image and LiDAR waveform viewer........ 39
5.2.4. Simple crown generation tool........ 40
5.2.5. Reconstruction of 3D scene from airborne point cloud........ 41
5.2.6. Python code automatic generation tool........ 42
5.3. LESS file format........ 42
5.3.1. instance.txt........ 42
5.4. SDK for Python........ 43
5.4.1. Implement LESS simulation using SDK for Python........ 43
5.4.2. Crown formation........ 53
6. LESS extension modules........ 54
6.1. LESS1D........ 54
References........ 58

1. Introduction

Three-dimensional (3D) radiative transfer (RT) modeling of the transport and interaction of radiation through earth surfaces is a challenging and difficult task. The difficulties lie in the complexity of the landscapes and also in the intensive computational cost of 3D RT simulations. To reduce computation time, current models work with schematic landscapes or with small-scale realistic scenes. The most accurate and efficient models (known as renderers) are from the computer graphics community, but the usages are not straightforward for performing scientific RT simulations. That's the reason why we developed this model.

Basically, LESS employs a forward photon tracing method to simulate bidirectional reflectance factor (BRF) or flux-related data (e.g., downwelling radiation) and a backward path tracing method to generate sensor images (e.g., fisheye images) or large-scale (e.g. 1 km^2) spectral images from visible to thermal infrared spectral domain. We also provide a user-friendly graphic user interface (GUI) and a set of tools are developed to help construct the landscape and set parameters. LESS has already been evaluated with other models in terms of directional BRF and pixel-wise simulated images. It can be used as benchmarks for validating physical models or training artificial neural network (ANN) to do parameter inversion.

2. Fundamentals of LESS

2.1 Forward photon tracing (FPT)

2.1.1 Photon tracing

Forward photon tracing (FPT) traces photon packets with the power $P(\lambda)$ into the scene from light sources. The initial power $P^0(\lambda)$ of each packet is determined by the power of light sources and the number of generated packets N. When generating photon packet in a scene with multiple light sources, a light source is randomly chosen according to the importance weight w_k, which is proportional to the power of each light source, i.e., $w_k = \frac{L_k(\lambda)}{\sum_{k=1}^{K} L_k(\lambda)}$ with $L_k(\lambda)$ being the power of light source k and K being the number of light sources. This mechanism guarantees that a light source with larger power has more sampled photon packets. The initial power of each packet, in terms of watt (W), is given as

$$P^0(\lambda) = \frac{\sum_{k=1}^{K} L_k(\lambda)}{N} \tag{1}$$

When a photon packet enters the scene along a path defined by its origin and direction of propagation, its intersection with landscape elements is tested for. If an intersection is found, the power of this packet will be scaled according to the optical properties of the intersected surface, i.e., the reflectance or transmittance. For a packet with Q times of scattering before it escapes from the scene, the power becomes

$$P^Q(\lambda) = P^0(\lambda) \cdot \prod_{q=1}^{Q} \left[\pi f(q, \omega_{i,} \omega_{o,} \lambda)/p\right] \tag{2}$$

where the $f(q, \omega_{i,} \omega_{o,} \lambda)$ is the bidirectional scattering distribution function (BSDF) at the q^{th} intersection point during its trajectory. $\omega_{i,}$ and ω_o are the incident direction and outgoing direction of a photon packet, respectively. Since the scattering law of surfaces in LESS is defined as Lambertian, the BSDF is interpreted as bidirectional reflectance distribution function (BRDF) or bidirectional transmittance distribution function (BTDF), according to $\omega_{i,}$, surface normal ω_n and ω_o, i.e.,

$$f(q, \omega_{i,} \omega_{o,} \lambda) = \frac{1}{\pi} \begin{cases} \rho_{\perp,\lambda} \cdot \operatorname{sgn}(\omega_n \cdot \omega_{i,}) + \tau_\lambda \cdot \operatorname{sgn}(-\omega_n \cdot \omega_{i,}), \ \omega_o \cdot \omega_n \geq 0 \\ \rho_{\top,\lambda} \cdot \operatorname{sgn}(-\omega_n \cdot \omega_{i,}) + \tau_\lambda \cdot \operatorname{sgn}(\omega_n \cdot \omega_{i,}), \ \omega_o \cdot \omega_n < 0 \end{cases} \tag{3}$$

where $\operatorname{sgn}(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$; $\frac{\rho_{\perp,\lambda}}{\pi}$ and $\frac{\rho_{\top,\lambda}}{\pi}$ are the upper and bottom surface BRDF, respectively; $\frac{\tau_\lambda}{\pi}$ is the BTDF. In LESS, the transmittances of the surface from both the upper and the bottom side are assumed to be the same; The outgoing direction of a photon packet after scattering is determined by randomly sampling the BSDF function. For Lambertian surfaces, the model chooses a random direction in the outgoing hemisphere. Since a single photon trajectory can be used to simulate BRF for any wavelength by updating the power according to the spectral reflectance/transmittance, we have omitted the symbol λ in the following equations for simplicity.

A photon packet is collected by the sensor if it exits the scene through the top boundary. Lateral boundary effects are considered in order to simulate horizontally infinite scenes with a repetitive pattern. As shown in **Figure 1**, the photon packet which exits from the lateral boundaries will re-enter the scene from the opposite side with the same photon direction until it escapes through the top boundary of the scene. When the scattering order of a packet exceeds a user-defined threshold (e.g., 5), the propagation of the packet is randomly stopped according to the "Russian roulette" mechanism (Kobayashi and Iwabuchi, 2008), which terminates the trajectory of a packet with a probability p (e.g., 5%). If the packet survives, its power will be multiplied by $\frac{1}{1-p}$.

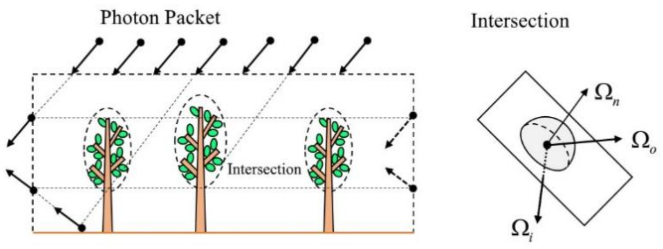


Figure 1. Forward photon tracing.

The collection of the escaped photon packets is achieved by placing a virtual hemisphere above the scene (Govaerts and Verstraete, 1998). As shown in **Figure 2**, the hemisphere is partitioned into N_P small surface elements (SE) with equal area $\Delta S = \frac{2\pi}{N_P}$ by utilizing the partition scheme of a disk via the equal area projection, i.e., $\Delta\Omega = \Delta S$. The zenithal zones are partitioned through (Beckers and Beckers, 2012):

$$\theta_i = \theta_{i-1} - \frac{2}{a_{aspect}} \sin\frac{\theta_{i-1}}{2} \sqrt{\frac{\pi}{k_{i-1}}}, k_i = k_{i-1} \left(\frac{r_i}{r_{i-1}}\right)^2 \tag{4}$$

Where (θ_i, θ_{i-1}) defines a zenithal zone on the hemisphere with $\theta_0 = \frac{\pi}{2}$; k_i is the total number of SEs inside a zenithal angle θ_i with $k_0 = N_P$; r_i is the radius corresponding to θ_i with $r_i = 2\sin\frac{\theta_i}{2}$ for a unit sphere due to the equal area projection; a_{aspect} is the aspect ratio of each SE, which is approximately enforced to 1. For each zenithal zone, it has $k_{i-1} - k_i$ SEs with equal area.

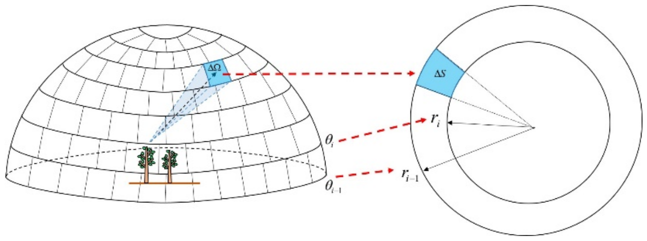


Figure 2. Unit hemisphere partition. The hemisphere is projected to horizontal plane as a disk using equal area projection. The radius r_i in the disk indicates zenith angle θ_i in the hemisphere.

When a photon packet exits the scene, the outgoing SE (solid angle) is determined by the photon direction only, i.e., the hemisphere is placed at an infinite position. The BRF in this SE can be estimated with (Govaerts and Verstraete, 1998)

$$f_{BRF_i} = \frac{\pi P_i^A}{\Delta\Omega_i \cdot \cos\theta_i^c \cdot P_{scene}} \tag{5}$$

where P_i^A is the power (watt) of all the captured photons in SE i, i.e., $P_i^A = \sum_{P^Q \in \Delta\Omega_i} P^Q$; $\Delta\Omega_i = \frac{2\pi}{N_P}$ is the solid angle of each SE; θ_i^c is the central zenith angle of solid angle $\Delta\Omega_i$; P_{scene} is the power of all the direct incident photon packets on a reference plane at the top of the scene, i.e., the incident radiation at the top of the scene. Once the power in each SE is determined, the scene albedo is computed as

$$\omega_{albedo} = \frac{\sum_{i=1}^{N_P} P_i^A}{P_{scene}} \tag{6}$$

2.1.2 Virtual photon

The real photon approach (described in section 2.2.1) estimates BRF by using small SEs on the sphere. More photon packets are needed to reduce the variance when smaller SEs are used. To solve this problem, a virtual photon approach, similar to the *virtual direction* in DART model (Yin et al., 2015), *secondary ray* in Rayspread model (Widlowski et al., 2006) or some "local estimates" methods (Antyufeev and Marshak, 1990; Marchuk et al., 1980), is introduced. If a packet is intercepted by an object (e.g., a tree) in the scene without complete absorption, the packet will be scattered in a direction which is randomly sampled by the BSDF function, and a virtual photon packet will be sent to each of the defined virtual directions. The possible scattered energy, in terms of intensity ($\mathrm{W \cdot sr^{-1}}$), is calculated as

$$I = V \cdot P^{q-1} \cdot f(q, \omega_i, \omega_v) \cdot \cos < \omega_v, \omega_n > \tag{7}$$

where P^{q-1} is the power of the incident photon packet at the q^{th} intersection point along its trajectory; ω_v is a virtual direction; V is a visibility factor which equals to zero if the a landscape element occludes the virtual photon packet, and equals to 1 otherwise. When sending the occlusion testing rays, the lateral boundary effect is also considered (**Figure 3**). The final BRF is then given as

$$f_{BRF_v} = \frac{\pi I_v^A}{\cos\theta_v \cdot P_{scene}} \tag{8}$$

where I_v^A is the power per unit solid angle ($\mathrm{W \cdot sr^{-1}}$) in virtual direction v and θ_v is the zenith angle of the virtual direction. An advantage of calculating a directional BRF using virtual photon approach is that the BRF is estimated within an infinity small solid angle (Thompson and Goel, 1998), which is the real directional BRF of a scene.

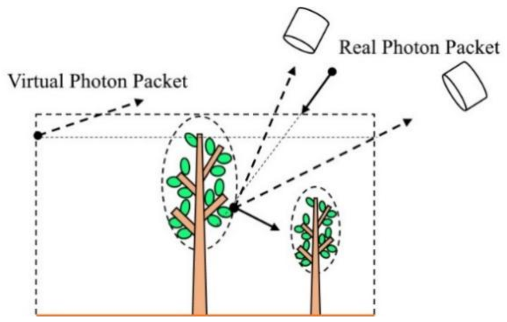


Figure 3. Virtual photon approach to calculate BRF.

2.2 Backward path tracing (BPT)

Instead of tracing photon packets from light sources, backward path tracing sends rays from sensors into the scene. The ray directions are controlled by sensor configurations (field of view, position, orientation, etc.). The main task of this ray-tracing algorithm is to establish a connection, which is called "path", between light sources and sensors and to determine the radiance incident onto the sensor. The radiance is estimated for every pixel in the output image. Radiance along direction ω_o can be calculated according to a rendering equation (Kajiya, 1986)

$$L_o(q,\omega_o) = L_e(q,\omega_o) + \int_{4\pi} f(q,\omega_i,\omega_o)\, L_i(q,\omega_i)|\cos\theta_i|d\omega_i \qquad (9)$$

where $L_o(q,\omega_o)$ is the outgoing radiance from point q along direction ω_o; $f(q,\omega_i,\omega_o)$ is the BSDF of the intersected surface, which determines the outgoing radiance along direction ω_o at point q induced by incoming radiance along direction ω_i; $L_i(q,\omega_i)$ is the incoming radiance; θ_i is the angle between ω_i and the surface normal and $L_e(q,\omega_o)$ is an emission term (e.g., thermal emission), which is described in detail in section 2.4. The algorithm that solves this equation is illustrated in **Figure 4**. A ray from the sensor intersects the elements in the scene at the point q. The outgoing radiance induced by the sun (or sky) is then calculated according to the BSDF. To calculate the multiple scattering radiation, a new ray is launched from the point q with a direction that randomly samples the BSDF. If this ray intersects the scene at another point q_1, the same procedure is applied to q_1. The outgoing radiance at point q_1 along the randomly selected direction is the incoming radiance of q along direction ω_1. The multiple scattering procedure is performed recursively until reaching the maximum scattering order (e.g., 5) specified by the user. To prevent energy loss due to the termination of scattering, the random cut-off technique ("Russian roulette") used in FPT is also applied to the sensor ray.

When simulating a horizontally infinite scene, the lateral boundary effect is considered for both the sensor ray and the illumination ray. At each intersected point (q_i), an illumination ray, which is built by randomly sampling a point q_e on the emitter, is sent towards the emitter. If this ray traverses the lateral boundary of the scene, it is also reintroduced into the scene to test whether the intersected point is occluded by other landscape elements or not.

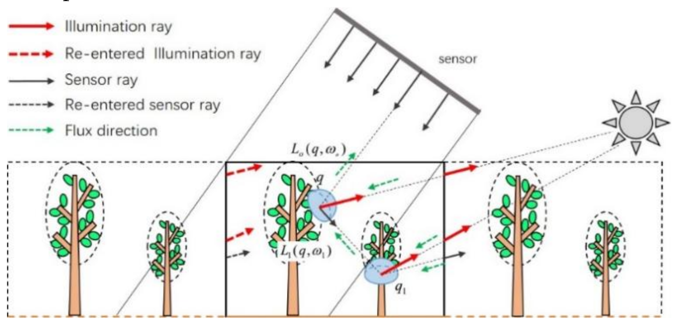


Figure 4. Backward path tracing.

2.3 BPT in simulating thermal infrared brightness temperature (BT)

When simulating thermal infrared radiation, the object itself, instead of the sun, becomes an "emitter", which emits thermal radiation according to Planck's law and its emissivity. Since the incident energy of the sun in thermal infrared bands (e.g., 10 μm) is too weak, we ignore this part of energy in our simulation. However, the presence of sun radiation will greatly influence the temperature distribution of objects due to the shadows cast between them. This generally classifies the scene elements into four components with specific temperatures, i.e., sunlit soil, shaded soil, sunlit leaves and shaded leaves (**Figure 5**). The determination of these four components is computed on the fly instead of a precomputing step adopted by most of other forward models (e.g., DART). This on-the-fly approach avoids the storage of emission points, which can greatly reduce the memory usage, especially for scenes with a large number of leaves.

If a sensor ray is intersected in the scene (i.e., q in the scene), an emission term at this point is added. To determine the emission power, an occlusion ray is traced towards the sun. If this point is directly illuminated, i.e., the occlusion ray intersects nothing, the temperature of the sunlit component is used. Otherwise it uses the temperature of the shaded component. The emitted radiation is calculated by using Planck's law with emissivity provided. Except for the emission term, another component of the power that goes into the sensor is the reflected power, which is emitted by other objects in the scene or sky radiation. In order to consider this part of the power, a random point on a randomly selected emitter (including the sky) is sampled (e.g., q_e in **Figure 5**). When this point is not on the sky, the emission power is determined by sending an occlusion ray towards the sun. The contribution of power from this point is calculated by using the BSDF defined at point q if this point is not occluded by other objects. This procedure can be recursively repeated, which is the same as the procedure described in section 2.2 except for the emission term. Finally, a thermal infrared image, which records the radiance value, can be simulated.

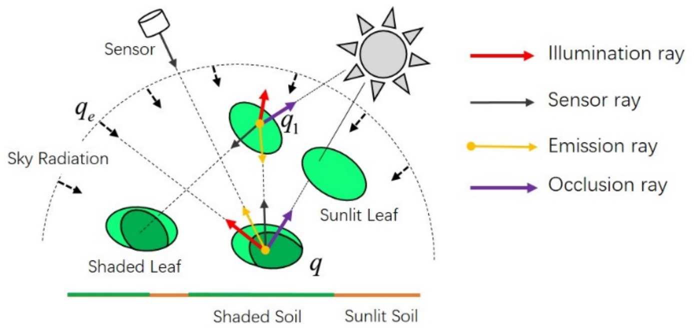


Figure 5. Simulation of thermal infrared radiation using backward path tracing.

3. Installation

• Windows

Download the installer from http://www.lessrt.org. All you need is just to choose a place to install it. (Usually, LESS cannot be installed in c:\program files, since it does not have administrative rights.). After installation, LESS will be automatically started.

4. Graphic User Interface (GUI)

LESS window is divided into four areas (**Figure 6**):

✧ The area 1 is Preview Panel
✧ The area 2 is Parameter Control Panel
✧ The area 3 is Progress Panel
✧ The area 4 is Menu Panel

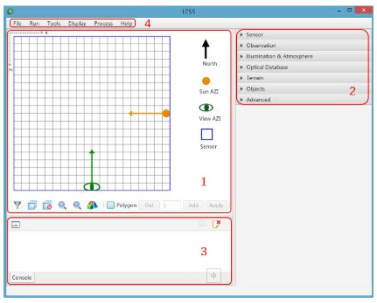


Figure 6. Main Screen of LESS

4.1 Preview Panel

Preview Panel is for displaying some information, such as view azimuth angle (👁), sun azimuth angle (●), sensor footprint (□) and tree positions. This will make it easier for user to set parameters, because this display is automatically updated when related parameters are changing.

Background Image ( ) - When you click the icon, you can choose a picture as background image, make sure that the image represents the same area defined in LESS.

Delete Background Image ( ) - When you click the icon, background image will be delete.

Zoom-in ( ) - This icon allows you to zoom-in the area covered by grids.

Zoom-out ( ) - This icon allows you to zoom-out the area covered by grids.

3D Viewer ( ) - If you click this icon, 3D Viewer will be activated. In 3D Viewer windows, you can visualize the scene in advance (before actual simulation) to verify the 3D scene.

Polygon (☐ Polygon) - When you check the icon, you can draw a polygon in the area covered by grid and set polygon parameter (such as the minimum distance between trees, tree position in the polygon) in the panel after it (Del 3 Add Apply). If you just want to allocate objects in some particular areas, the solution is using Polygon tool. Check the option of Polygon under the display panel. This time when your mouse enters the display panel, it will become a hand. you can create a polygon by left click. If you want to remove the polygon, just use your right click of mouse. After the creation of polygon, you can delete or add object instances in this area. If you click Add, then all the generated instances will only apprear in the polygon, the position and number of each object are still random within the polygon. However, if you choose one of the objects in the Objects List. then the generated instances only contains this object. This tool can create some forest scene which contains one species of trees in some area and another species of tree in another areas.

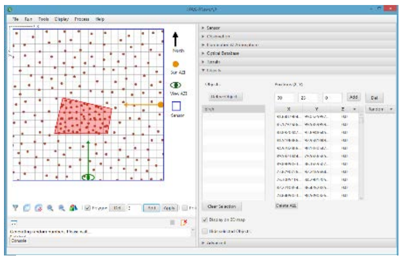


Figure 7. Polygon tools

Point (☐ Point) - When you check the icon, you will see the cursor changing to a hand. You can choose the position of tree by clicking the place where you want the tree to be, without inputting coordinates.

4.2 Parameter Control Panel

Parameter Control Panel is for setting parameter which can control the scene you simulate.

4.2.1 Sensor

To simulate a sensor you can input parameters here to control image type, the number of pixels in width and height, sample count per pixel.

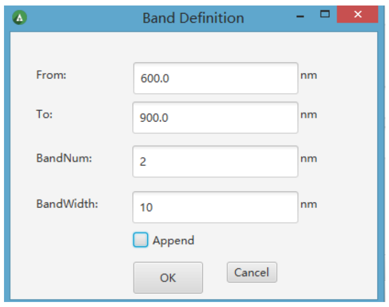
Type - Up to now, there are four types that you can choose. They are orthographic, perspective, CircularFisheye, PhotonTracing.

Width - The number of pixels in width.

Height - The number of pixels in height.

Samples - sample count per square meter. Usually 128 is sufficient. If you increase to 256, 512, the quality of the simulated image will be better.

Spectral Bands - It represents the which band you want to simulate. For example, RED and NIR: 660nm，900nm. Now, you must also input a bandwidth for each band with the format of "center band: bandwidths", e.g. 660:10,900:10. These bandwidths will be used for determining the irradiance. If you have no special requirement, it can just set the bandwidth to zero. And when you click [Define], you can define the bands by input some parameter in the pop-up window (**Figure 8**).


Band Definition

From: 600.0 nm

To: 900.0 nm

BandNum: 2 nm

BandWidth: 10 nm

☐ Append

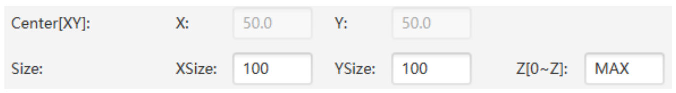
OK Cancel

Figure 8. Spectral bands

Image Format - "Spectrum" means generating spectral image. "Synthesized RGB Image" means generating RGB image.

Only First Order? - If it is true, it means LESS only simulate the first-scattering event (sun * radiation reflected by objects only one time). If set to false, then both first-scattering and multi-scatting will be simulated.

Virtual Plane – When you check it, a virtual plane is defined, and only the radiant that exits through the plane is calculated. In **Figure 9**, you can define the location of the virtual plane in "Center" and the size of the virtual plane in "Size". Usually, the height of the plane (Z) should be left as default, i.e., MAX.


Center[XY]: X: 50.0 Y: 50.0

Size: XSize: 100 YSize: 100 Z[0~Z]: MAX

Figure 9. Virtual plane

Thermal Radiation - When you need to get thermal infrared image, you can check it. After checking, you can input surface temperature parameter in Optical Database (**Figure 10**).

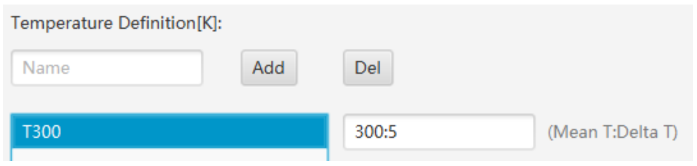


Figure 10. Temperature definition

NoData Value - Set the background value when sensor footprint is beyond the scene scale. If the image opened by ENVI, you should set it as “0”.
Repetitive scene - Sets the number of copies that distributed around the scene.
Width Extent/Height Extent - The actual extend of the orthographic sensor can simulate. It is similar to FOV of perspective sensor.
Four Components Product - If you check it, a classification image will be generated. The classification image has four components, and they are sunlit ground, sunlit canopy, shaded ground and shaded canopy. The corresponding pixel values are 1, 2, 3 and 4, respectively. Please note that some pixels may mixed pixels, in this case, the dominant component is set as the pixel component type. However, if you set the number of bands >= 5, the generated four component image will contain 5 spectral bands, the first band is the classification image, the other four bands are the proportions of the four components. This can be used to estimate directional gap fraction of forest canopy.

4.2.2 Observation

View Zenith - Set view zenith angle (angle between vertical direction).
View Azimuth - Set view azimuth （0° is north, clock-wise to 360°).
Sensor Height - Set the height of the sensor.

4.2.3 Illumination & Atmosphere

There are two groups of illuminations, one is from sun, other is from atmosphere.
Sun Zenith - Zenith angle of sun.
Sun Azimuth - Azimuth angle of sun (0° is north, clock-wise to 360°).
This defines the position of sun. That means if you set it as 90° (East), then it will produce shadows in West.
Sun Position Calculator - If you clicked it, you can fill in above two parameters by inputting the time and place parameters for the scene.
Next is the parameter setting for the sky.
Type - It represents the type of atmosphere radiation, now only one option can be used - “SKY_TO_TOTAL”， it means the ratio between diffuse radiation and total incident radiation. Thus (1-SKYL) * T (T is sun radiation above atmosphere) is the sun radiation under atmosphere. Under this mode, atmosphere is isotropic diffuse radiation from upper hemisphere.
Percentage - It defines the actual ratio between atmosphere radiation and total radiation. What should be noticed is that when you input the values, the number of values should be equal to the number of bands (under the sensor section). That is, for each band, it may have different values.
Input solar spectrum manually - If you check it, you can input the solar spectrum and the sky spectral in terms of wavelength manually (in W/m^2/nm).

4.2.4 Optical Database

Since objects in LESS are represented as triangular meshes, thus for each triangle, it can have at least three kinds of optical properties: reflectance of front side, reflectance of back side, and transmittance (we assume transmittance of both side is the same). In Optical database, we should first define some optical model, and then they can be used in the following terrain or forest definition. By default, there are three optical models defined. Please note that, these three optical models cannot be modified, if you want to use them with modified value, you should copy a new one and then perform the modification You can also add new optical model directly. For each reflectance or transmittance, values of different bands are connected by comma.

Please note that for some object component, such as trunk, the normal of some triangles may face inside, which shows a zero reflectance outside. To avoid this situation, a convenient solution is to set the reflectance of both outer and inner side to the same values, but with zero transmittance.

4.2.5 Terrain

There are mainly three types of terrain: PLANE, RASTER, MESH.
PLANE just represents simple plane lies at altitude of 0. RASTER is a raster image file in ENVI standard format.
XSize - Size in the X direction.
YSize - Size in the Y direction.
There are two BRDF types: Lambertian, Soilspect. If Lambertian is chosen, it means you think of the ground as a Lambertian. If Soilspect is chosen , you can input parameters to define the model.
Land Cover - If you check it, you can input surface classification data generated by ENVI, and then set different spectrum for different ground classes.

4.2.6 Objects

Objects define what you want to put in the scene. For example, forest is formed by a number of single trees, which are describe by triangle mesh (obj file) in LESS. Usually, we cannot input a obj for each single trees, since forest contains a lot of trees and a tree itself contains numerous triangles, it may not be possible even for large memory computers. The alternative is to use a “instance” technique. That means we define a single tree, and we can place it at different places, just using reference, thus the program only keeps one copy of the triangle mesh, but it represents trees at different places (they have exactly the same structures, but we can do rigid transformations).

Objects window is divided into three areas:
• The area (1) is Objects Define Panel;
• The area (2) is Position Parameter Control Panel;
• The area (3) is Display Panel.

(1) Objects Define Panel

The first step is to define some objects (single tree).
Define Object… - You can click it to define objects. It will open a new dialog that allows to import obj file (**Figure 11**).
Add - We give a name for our first object, such as “birch”. After clicking [Add] button, “birch” will appear in the Objects list.

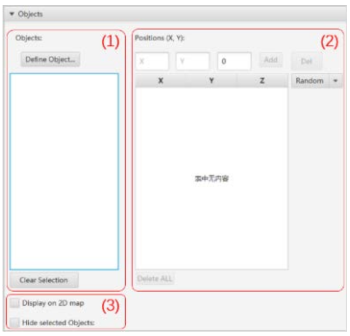


Figure 11. Objects definition

Import OBJ - Selecting the name we write in “Objects” area, then the button [Import OBJ] is activated. If you click the [Import OBJ], you can choose a obj file in the window and input it as the object (**Figure 12**).
And then you need to determine the scale by the units of the tree model that you enter into. If the units of the tree model that you enter into is “cm”, the scale should be 0.01. And if the units of the tree model that you enter into is “m”, the scale should be 1.00.

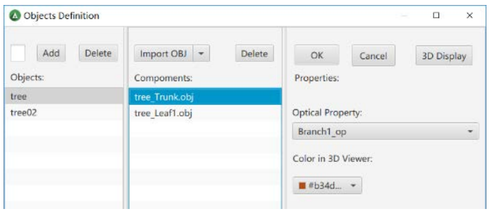


Figure 12. Import obj file

Import from RAMI - Import objects from the model file on the rami website. After importing objects, each component of the obj file will appear in the “Components” list. Component is a part of the object, e.g. a tree contains leaves and branches (usually they have different optical properties). **You should be aware that the obj file itself will be copied to the simulation folder, so you can safely delete it from your original place.**

Optical Property - Selecting one of the components, the [Optical Property] is activated and you can choose an optical property for the selected component. These optical properties are also from “Optical Database”. If you need to define a new optical model, you can go back to “Optical Database” page without closing the current window. When you finish, the optical models are automatically synchronized.
3D Display - After clicking it, you can see the 3D model of object (**Figure 13**).

After defining objects, you should click [OK] button, and then we will go back to Objects page with the defined objects shown in Objects list.

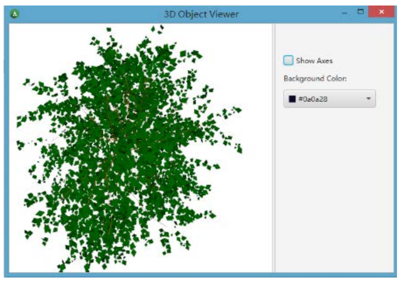


Figure 13. 3D Display

(2) Position Parameter Control Panel

If we select the defined objects, we can define its positions. There are three methods to define positions. The first method is inputting coordinate for the object directly. The second method is using [Point] tool provided by LESS (introduced in Preview Panel). The third method is using [Random] tool provided by LESS. The last method is import CHM data to control position.

Add - You can enter coordinates in the table before [Add] button, and then clicking [Add], the object is added to the specified location.

Random - Clicking the button [Random], it will open a new window, now the only choice is Poisson distribution (This is very normal for trees in forest). The only parameter you need to provide is the minimum distance between two objects. Click [OK], then LESS will automatically generate instances of defined objects. The number and position of each object in the Objects list is also random. Thus, you can get a reasonable distribution of objects.

From CHM - Clicking the inverted triangle next to the button [Random], you can open the drop-down menu, and find the [From CHM] button. If click it, you can choose a CHM data as a basis for objects location.

(3) Display Panel

Display on 2D map - If a position is added when you check it, the position will appear on the grid area in Preview Panel, which will help you to check whether it is correct.

Hide selected Objects - If there are multiple plants in this scene, you can hide the location of the model you don't want to see by checking it.

4.3 Progress Panel

This panel shows the running state of the current program. When an error occurs, you can find the reason of the error by reading this panel.

4.4 Menu Panel

Menu bar holds 6 menus: **File, Run, Tools, Display, Process and Help**

4.4.1 File menu

File > New Simulation - Create a new simulation.
File > Open Simulation - Open a chosen LESS file and load its parameter values into LESS modeler.
File > Save - Save current parameter values on a disk as LESS file.
File > Save as - Save the tree image into another file.
File > Close - Close the simulation.

4.4.2 Run menu

Run > Run All - Run this program and output results.
Run > Generate 3D Model - Generate 3D objects, e.g., convert obj file into binary format.
Run > Generate View & Illumination - Generate viewing parameters for simulation.

Run > less - Run less simulation

4.4.3 Tools menu

Tools > Open Results Folder - Open Results Folder of current simulation.
Tools > Batch Tools - Do batch processing.
Tools > Server Setting - Do network parallel simulation
Tools > LAI Calculator - Calculate LAI in different resolution, and output the results as txt.
Tools > Python Console - Run Python scripts (under experiment).

4.4.4 Display menu

Display > 3D Viewer - Display scene you simulate from multiple angles.
Display > 3D Viewer (Bounding BOX) - Display scene you simulate from multiple angles with replacing objects with boxes.
Display > 2D Polygon - Draw a polygon in the area covered by grid and set polygon parameter (such as the minimum distance between trees, tree position in the polygon) in the panel after it.

4.4.5 Process menu

Process > BRF Processing - Generate BRF image.
Process > Brightness Temperature Processing

4.4.6 Help menu

Help > Documentation - Documentation of LESS.

5. Practical tutorial

5.1 Major functionalities

5.1.1 FPAR simulations

FPAR is the fraction of total absorbed radiation, usually from 400 to 700 nm. Therefore, to simulate FPAR using LESS, we need first to set the band range from 400 to 700 nm (or other spectrum if you want), the band number can be, e.g., 30. Then you need to set the leaf and soil optical properties in [**Optical Database**] section, the number of spectrum bands should be the same with the band number you set in the [**Sensor**] section, e.g., the 30 you have previously set.

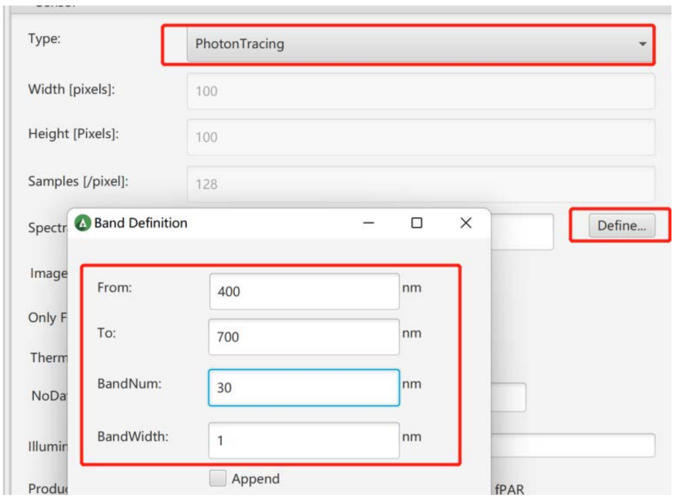
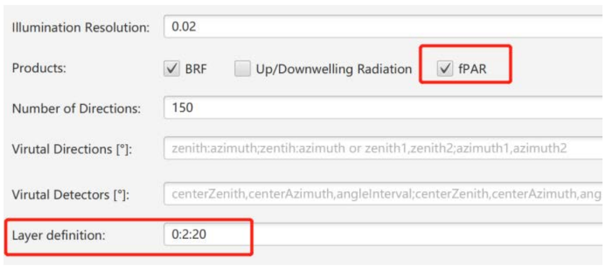

Type: PhotonTracing
Width [pixels]: 100
Height [Pixels]: 100
Samples [/pixel]: 128
Band Definition
From: 400 nm
To: 700 nm
BandNum: 30 nm
BandWidth: 1 nm

Figure 14. Set the spectral bands for FPAR simulation.

Next step is to enable the FPAR option in [**sensor**] section as illustrated in **Figure 15**. An important parameter is the **Layer definition (Start:Interval:End),** which defines how FPAR is outputted. Specifically, LESS only outputs the FPAR values between **Start** and **End** range with interval equal to **Interval.** For example, if you define it as 0:2:10, then you will get FPAR for layers: [0,2), [2,4) , [4,6) , [6,8) , [8,10).



Illumination Resolution:	0.02
Products:	✓ BRF ☐ Up/Downwelling Radiation ✓ fPAR
Number of Directions:	150
Virutal Directions [°]:	zenith:azimuth;zentih:azimuth or zenith1,zenith2;azimuth1,azimuth2
Virutal Detectors [°]:	centerZenith,centerAzimuth,angleInterval;centerZenith,centerAzimuth,ang
Layer definition:	0:2:20

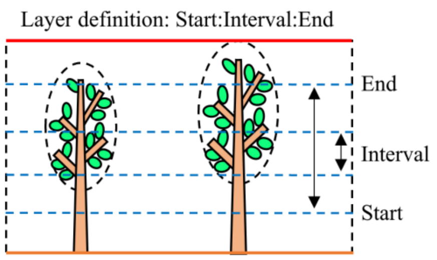
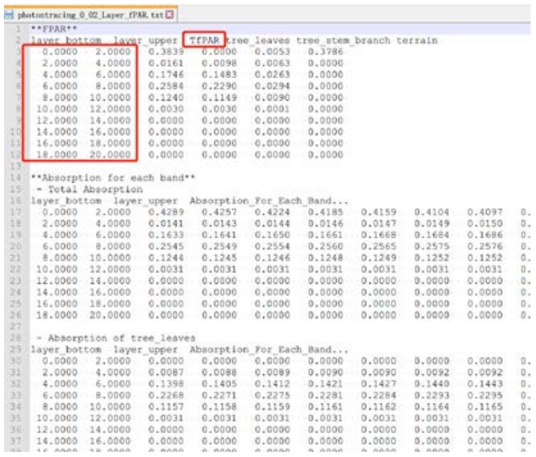


Figure 15. Enable FPAR simulation in LESS and define layers.

A simulation example is shown in **Figure 16**, the file can be found in the Results folder within the simulation folder. The results five two kinds of product, one is the spectrally integrated total FPAR, the other is the absorption for each band. For each layer, we first integrate the total PAR over the whole spectrum range, and then divided by the total incident energy received by a plane at the top of the canopy（also integrated over the whole spectrum range）(**Figure 16**). The total FPAR is presented at the top of the result file ((**Figure 16** left), it illustrates the total FPAR (TfPAR) for each layers, e.g., from 0 m to 2 m, the total FPAR is 0.3839, which may be mainly absorbed by soil. If you are interested in leaves, maybe you should look at the 4th column, it outputs the FPAR for the leave components. Please note that the name of the column is not fixed, it is according to the component you defined in LESS when you import your OBJ file into LESS. If you don't want the spectrally integrated FPAR, you can refer to the **Absorption for each band**, here, the FPAR for each band is presented. '- Total Absorption' means the total absorbed energy for all the landscape components (e.g., leaves, soil and branches) for each band. The column means the spectral bands, thus you would see 32 columns if you set the spectral bands to 30, the first two columns are layer heights. Next content is the FPAR or absorption for other landscape components, for example the tree_leaves. In conclusion, if you are interested in the total and spectrally integrated FPAR, you only need to refer to the first section of the result file.



photontracing_0_02_Layer_fPAR.txt

```
**FPAR**
layer_bottom  layer_upper  TfPAR tree_leaves tree_stem_branch terrain
  0.0000   2.0000   0.3839   0.0000   0.0053   0.3786
  2.0000   4.0000   0.0161   0.0098   0.0063   0.0000
  4.0000   6.0000   0.1746   0.1483   0.0263   0.0000
  6.0000   8.0000   0.2584   0.2290   0.0294   0.0000
  8.0000  10.0000   0.1240   0.1149   0.0090   0.0000
 10.0000  12.0000   0.0030   0.0030   0.0001   0.0000
 12.0000  14.0000   0.0000   0.0000   0.0000   0.0000
 14.0000  16.0000   0.0000   0.0000   0.0000   0.0000
 16.0000  18.0000   0.0000   0.0000   0.0000   0.0000
 18.0000  20.0000   0.0000   0.0000   0.0000   0.0000

**Absorption for each band**
 - Total Absorption
layer_bottom  layer_upper  Absorption_For_Each_Band...
  0.0000   2.0000   0.4289   0.4257   0.4224   0.4185   0.4159   0.4104   0.4097   0.
  2.0000   4.0000   0.0141   0.0143   0.0144   0.0146   0.0147   0.0149   0.0150   0.
  4.0000   6.0000   0.1633   0.1641   0.1650   0.1661   0.1668   0.1684   0.1686   0.
  6.0000   8.0000   0.2545   0.2549   0.2554   0.2560   0.2565   0.2575   0.2576   0.
  8.0000  10.0000   0.1244   0.1245   0.1246   0.1248   0.1249   0.1252   0.1252   0.
 10.0000  12.0000   0.0031   0.0031   0.0031   0.0031   0.0031   0.0031   0.0031   0.
 12.0000  14.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
 14.0000  16.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
 16.0000  18.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
 18.0000  20.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.

 - Absorption of tree_leaves
layer_bottom  layer_upper  Absorption_For_Each_Band...
  0.0000   2.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
  2.0000   4.0000   0.0087   0.0088   0.0089   0.0090   0.0090   0.0092   0.0092   0.
  4.0000   6.0000   0.1398   0.1405   0.1412   0.1421   0.1427   0.1440   0.1443   0.
  6.0000   8.0000   0.2268   0.2271   0.2275   0.2281   0.2284   0.2293   0.2295   0.
  8.0000  10.0000   0.1157   0.1158   0.1159   0.1161   0.1162   0.1164   0.1165   0.
 10.0000  12.0000   0.0031   0.0031   0.0031   0.0031   0.0031   0.0031   0.0031   0.
 12.0000  14.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
 14.0000  16.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.
```

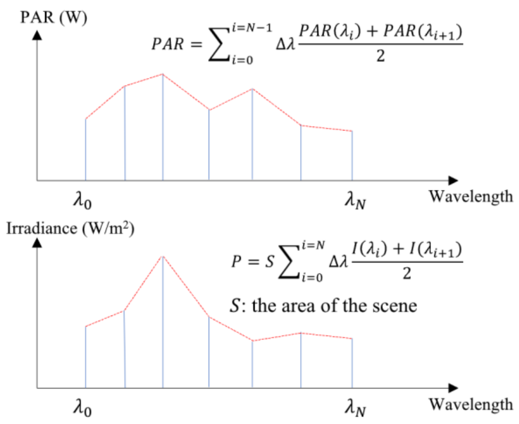
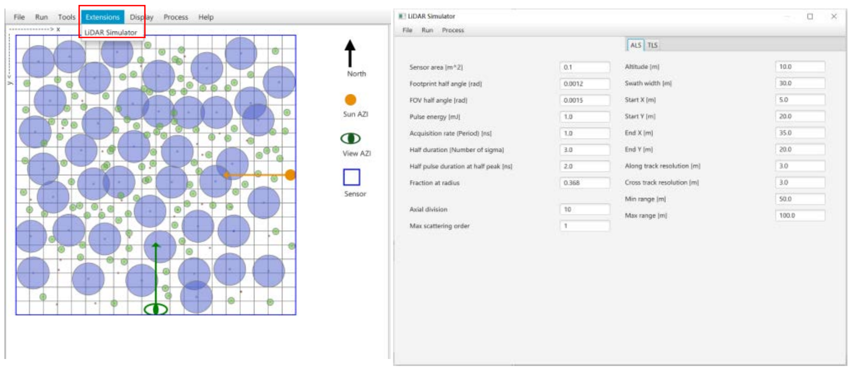


Figure 16. FPAR simulation example: simulation results (left); total FPAR calculation.

5.1.2 LiDAR simulations

Up to now, less has provided airborne laser scanning (ALS) and ground laser scanning (TLS) data simulation in graphical user interface, The simulator can get different types of data through file configuration, such as full waveform and 3D point cloud data, including single-band, multi-band and even hyperspectral LiDAR scanning (HLS) data. The laser simulator is located in GUI extension module. You should use the mouse to indicate [**Extensions**] and click on the button [**LiDAR Simulator**]. You can see **Figure 17** for the specific operation of opening the simulator.



Sensor area [m^2]	0.1
Footprint half angle [rad]	0.0012
FOV half angle [rad]	0.0015
Pulse energy [mJ]	1.0
Acquisition rate (Period) [ns]	1.0
Half duration (Number of sigma)	3.0
Half pulse duration at half peak [ns]	2.0
Fraction at radius	0.368
Axial division	10
Max scattering order	1
Altitude [m]	10.0
Swath width [m]	30.0
Start X [m]	5.0
Start Y [m]	20.0
End X [m]	35.0
End Y [m]	20.0
Along track resolution [m]	3.0
Cross track resolution [m]	3.0
Min range [m]	50.0
Max range [m]	100.0

Figure 17. LiDAR simulator interface

Before starting the simulation, you also need to understand the configuration of each parameter so that you can get interested LiDAR data in the simulator. Parameters are divided into two categories. The sensor parameters are located on the left side of the GUI, and the platform parameters are located on the right side. The meaning of each parameter is presented in Table 1, Table 2 and Table 3, Some parameters are also presented in **Figure 18** to help users better understand.

Table1. Sensor parameters	
Sensor Area	Sensor size (area) of LiDAR receiving energy, The sensor size (area) of LiDAR receiving energy, the larger the sensor area, the more energy it can receive.
Footprint half angle	Field angle of LiDAR pulse. Note: This is half of the cone angle of the field of view.
FOV half angle	FOV represents the field of view range of LiDAR receiving energy, which is set to half of the cone angle of the field of view. The larger the value, the larger the acceptable range. Generally, you need to set FOV half angle larger than footprint half angle. See Figure. 15 specifically
Acquisition rate (Period)	The acquisition rate represents a certain sampling interval and is usually determined by the instrument itself.
Half duration (Number of sigma)	The emission energy of laser pulse is usually assumed to be a Gaussian distribution function related to time and amplitude. ±3σrepresents the effective information range of the whole pulse energy. Note: Half duration represents half of the effective energy range.
Half pulse duration at half peak	Half pulse duration at half peak represents the standard deviation of Gaussian pulse energy. Half duration and Half pulse duration at half peak together determine the effective information range of pulse energy.
Axial division	The cross section of laser pulse is divided into n*n equal parts, and then sub-beams are emitted. The n here is the Axial division. This is only valid for multi-ray point cloud simulation. The larger the value, the finer the simulation of each pulse, but the simulation time is longer.
Table 2. ALS platform parameters	
Altitude	The flying height of the platform where the LiDAR is located, which

	is relative to the zero point of the scene rather than the terrain. See Figure. 15 specifically.
StartX, StartY, EndX, EndY	The start point and end point of ALS scanning, the projection of these points on the ground, the platform will fly in a straight line from the point (StartX, StartY, Altitude) to (EndX, EndY, Altitude). See Figure. 15 specifically.
Swath width	Width of Lidar scanning (horizontal width of vertical to flight direction). See Figure. 15 specifically.
Along track resolution (Yaw resolution)	Pulse interval along flight direction (ground distance). See Figure. 15 specifically.
Cross track resolution (Range resolution)	Pulse interval of vertical to flight direction (ground distance). See Figure. 15 specifically.
Min range, Max range	The range data [Min range, Max range] in this vertical direction can be collected and saved by the LiDAR sensor. See Figure. 15 specifically.
Table 3. TLS platform parameters	
TLS position X, Y, Z	Location of TLS scanning. X, Y is the horizontal projection position of the ground, and Z is the height of the instrument.
Center zenith, Center azimuth	The position of the bisector of zenith angle or azimuth angle.
Delta zenith, Delta azimuth	Range size of zenith angle or azimuth angle. Zenith angle range is generally set from 0 to180. Azimuth angle range is generally set from 0 to 360.
Resolution zenith, Resolution azimuth	The angle formed by two connected sampling points in zenith angle or azimuth is called angular resolution. The higher the angular resolution, the higher the point cloud density.
Cross track resolution (Range resolution)	Pulse interval of vertical to flight direction (ground distance). See Figure. 15 specifically.
Echo Detection Model	LiDAR simulator provides two detection modes, one is simple and the other is Gaussian. Gaussian decomposition is used to solve the parameters, and the waveform is decomposed to obtain discrete points, so the simulation is slow but accurate.

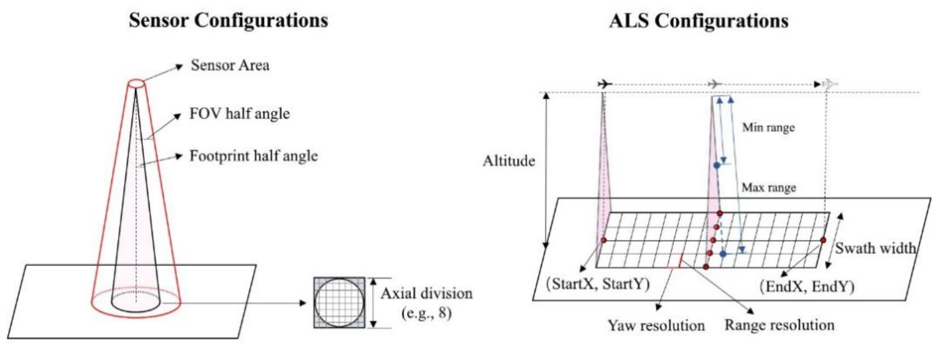
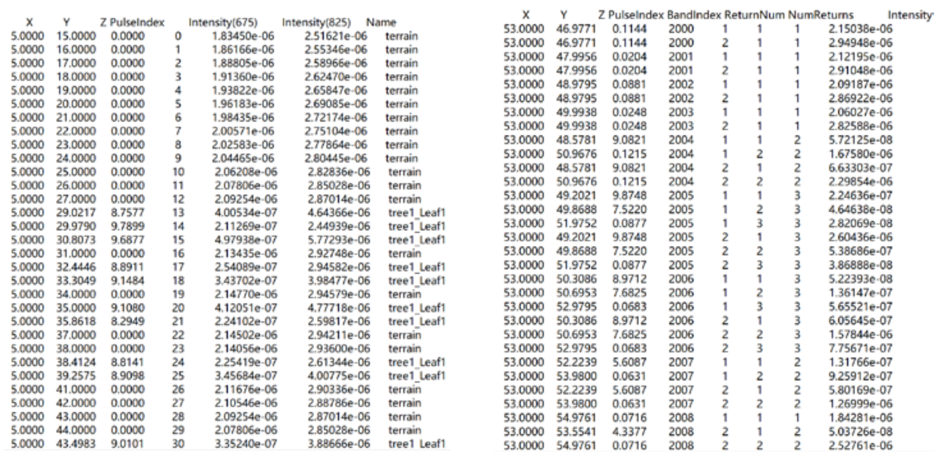


Figure 18. Illustration of LiDAR related parameters

- **Discrete point cloud data simulation**

Note that before starting the simulation, you must build a project and configure the scenes you are interested in according to the operation tutorial in section 4.2. Furthermore, you need to configure the parameters of sensors and platforms according to the simulator interface. Finally, you can start your discrete point cloud simulation work. Using the mouse to indicate [**RUN]** and click the button [**Generate ALL**], The simulator starts to generate 3D scenes and LiDAR configuration parameters. Then click button [**Single ray point cloud**] or button [**Multi rays point cloud]** to simulate point cloud. The former emits one light for each pulse, so the computer is faster. The latter emits multiple rays for each pulse (the number is determined by Axial division). In this mode, the full waveform is first simulated internally, and then discrete points are obtained by Gaussian decomposition, so the calculation speed is slower than [**Single ray point cloud**] mode. The simulation results can be found in the **Results** file. The results of two different modes correspond to **singleRay** and **pointcloud** folder. The results of point cloud generated by [**Single ray point cloud**] and [**Multi rays point cloud**] are presented in **Figure 19**, respectively.



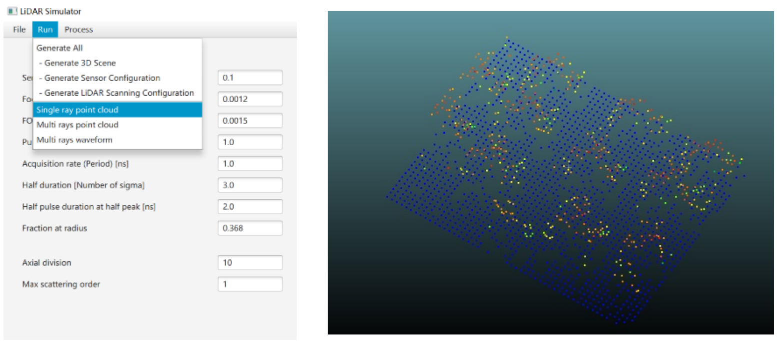
X	Y	Z	PulseIndex	Intensity(675)	Intensity(825)	Name
5.0000	15.0000	0.0000	0	1.83450e-06	2.51621e-06	terrain
5.0000	16.0000	0.0000	1	1.86166e-06	2.55346e-06	terrain
5.0000	17.0000	0.0000	2	1.88805e-06	2.58966e-06	terrain
5.0000	18.0000	0.0000	3	1.91360e-06	2.62470e-06	terrain
5.0000	19.0000	0.0000	4	1.93822e-06	2.65847e-06	terrain
5.0000	20.0000	0.0000	5	1.96183e-06	2.69085e-06	terrain
5.0000	21.0000	0.0000	6	1.98435e-06	2.72174e-06	terrain
5.0000	22.0000	0.0000	7	2.00571e-06	2.75104e-06	terrain
5.0000	23.0000	0.0000	8	2.02583e-06	2.77864e-06	terrain
5.0000	24.0000	0.0000	9	2.04465e-06	2.80445e-06	terrain
5.0000	25.0000	0.0000	10	2.06208e-06	2.82836e-06	terrain
5.0000	26.0000	0.0000	11	2.07806e-06	2.85028e-06	terrain
5.0000	27.0000	0.0000	12	2.09254e-06	2.87014e-06	terrain
5.0000	29.0217	8.7577	13	4.00534e-07	4.64366e-06	tree1_Leaf1
5.0000	29.9790	9.7899	14	2.11269e-07	2.44939e-06	tree1_Leaf1
5.0000	30.8073	9.6877	15	4.97938e-07	5.77293e-06	tree1_Leaf1
5.0000	31.0000	0.0000	16	2.13435e-06	2.92748e-06	terrain
5.0000	32.4446	8.8911	17	2.54089e-07	2.94582e-06	tree1_Leaf1
5.0000	33.3049	9.1484	18	3.43702e-07	3.98477e-06	tree1_Leaf1
5.0000	34.0000	0.0000	19	2.14770e-06	2.94579e-06	terrain
5.0000	35.0000	9.1080	20	4.12051e-07	4.77718e-06	tree1_Leaf1
5.0000	35.8618	8.2949	21	2.24102e-07	2.59817e-06	tree1_Leaf1
5.0000	37.0000	0.0000	22	2.14502e-06	2.94211e-06	terrain
5.0000	38.0000	0.0000	23	2.14056e-06	2.93600e-06	terrain
5.0000	38.4124	8.8141	24	2.25419e-07	2.61344e-06	tree1_Leaf1
5.0000	39.2575	8.9098	25	3.45684e-07	4.00775e-06	tree1_Leaf1
5.0000	41.0000	0.0000	26	2.11676e-06	2.90336e-06	terrain
5.0000	42.0000	0.0000	27	2.10546e-06	2.88786e-06	terrain
5.0000	43.0000	0.0000	28	2.09254e-06	2.87014e-06	terrain
5.0000	44.0000	0.0000	29	2.07806e-06	2.85028e-06	terrain
5.0000	43.4983	9.0101	30	3.35240e-07	3.88666e-06	tree1_Leaf1

X	Y	Z	PulseIndex	BandIndex	ReturnNum	NumReturns	Intensity
53.0000	46.9771	0.1144	2000	1	1	1	2.15038e-06
53.0000	46.9771	0.1144	2000	2	1	1	2.94948e-06
53.0000	47.9956	0.0204	2001	1	1	1	2.12195e-06
53.0000	47.9956	0.0204	2001	2	1	1	2.91048e-06
53.0000	48.9795	0.0881	2002	1	1	1	2.09187e-06
53.0000	48.9795	0.0881	2002	2	1	1	2.86922e-06
53.0000	49.9938	0.0248	2003	1	1	1	2.06027e-06
53.0000	49.9938	0.0248	2003	2	1	1	2.82588e-06
53.0000	48.5781	9.0821	2004	1	1	2	5.72125e-08
53.0000	50.9676	0.1215	2004	1	2	2	1.67580e-06
53.0000	48.5781	9.0821	2004	2	1	2	6.63303e-07
53.0000	50.9676	0.1215	2004	2	2	2	2.29854e-06
53.0000	49.2021	9.8748	2005	1	1	3	2.24636e-07
53.0000	49.8688	7.5220	2005	1	2	3	4.64638e-08
53.0000	51.9752	0.0877	2005	1	3	3	2.82069e-08
53.0000	49.2021	9.8748	2005	2	1	3	2.60436e-06
53.0000	49.8688	7.5220	2005	2	2	3	5.38686e-07
53.0000	51.9752	0.0877	2005	2	3	3	3.86888e-08
53.0000	50.3086	8.9712	2006	1	1	3	5.22393e-08
53.0000	50.6953	7.6825	2006	1	2	3	1.36147e-07
53.0000	52.9795	0.0683	2006	1	3	3	5.65521e-07
53.0000	50.3086	8.9712	2006	2	1	3	6.05645e-07
53.0000	50.6953	7.6825	2006	2	2	3	1.57844e-06
53.0000	52.9795	0.0683	2006	2	3	3	7.75671e-07
53.0000	52.2239	5.6087	2007	1	1	2	1.31766e-07
53.0000	53.9800	0.0631	2007	1	2	2	9.25912e-07
53.0000	52.2239	5.6087	2007	2	1	2	5.80169e-07
53.0000	53.9800	0.0631	2007	2	2	2	1.26999e-06
53.0000	54.9761	0.0716	2008	1	1	1	1.84281e-06
53.0000	53.5541	4.3377	2008	2	1	2	5.03726e-08
53.0000	54.9761	0.0716	2008	2	2	2	2.52761e-06

Figure 19. Simulation results of single ray point cloud (left); Simulation results of multi ray point cloud (right).

• Single ray point cloud simulation results

Single ray point cloud simulation results are located in the **singleRay** folder, the results of single ray point cloud simulation record the XYZ coordinates, intensity and other related information of discrete point cloud data (Figure16). Less In order to meet the needs of different users, better use LiDAR simulator to obtain more useful data. It is worth noting that when users input spectral information of different bands, multi-spectral or even hyperspectral discrete point cloud data will be obtained in [**Single ray point cloud**] mode. At the same time, the [**Single ray point cloud**] mode may provide users with more humanized data, and the simulation results also record the types of ground objects that each pulse touches during transmission, which may make users analysis more convenient. Different types of ground objects touched by lidar pulses are recorded in the last column of the result file. An example of simulating discrete point clouds with a single ray is shown in **Figure 20**.

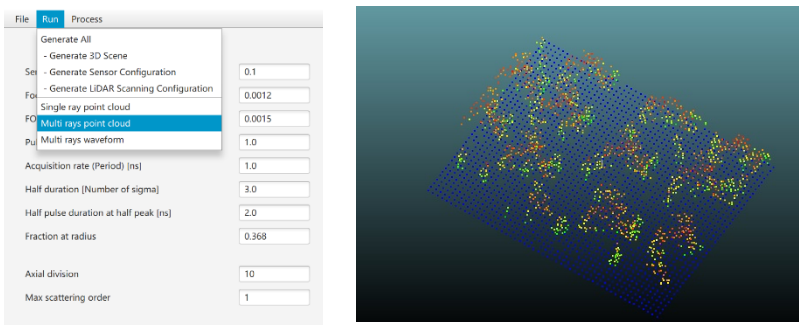


Se	0.1
Fo	0.0012
FO	0.0015
Pu	1.0
Acquisition rate (Period) [ns]	1.0
Half duration [Number of sigma]	3.0
Half pulse duration at half peak [ns]	2.0
Fraction at radius	0.368
Axial division	10
Max scattering order	1

Figure 20. Example of simulating discrete point cloud with single ray.

- **Multi ray point cloud simulation results**

Multi ray point cloud simulation results are located in the **pointcloud** folder, the results of Multi ray point cloud simulation record the XYZ coordinates, Return Number, Number of Return, intensity and other related information of discrete point cloud data (Figure17). Note that the result file also records the band index fields of different point clouds. Therefore, multispectral or even hyperspectral discrete point cloud data acquisition is also supported in [**Multi rays point cloud]** mode. Compared with [**Single ray point cloud**] mode, [**Multi rays point cloud]** mode requires users to further filter and separate the discrete point cloud data of different bands according to the band index fields. An example of simulating discrete point clouds with a Multi rays are shown in **Figure 21**.



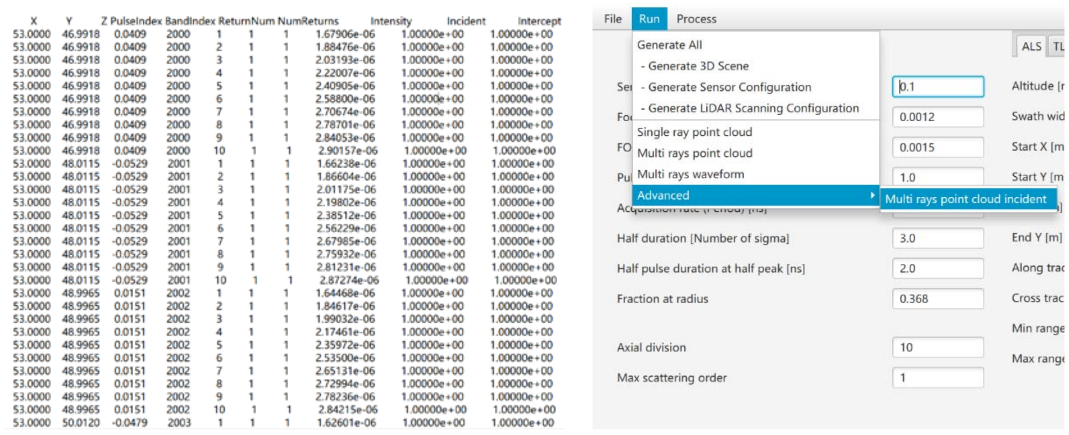
```
Se                                          0.1
Fo                                          0.0012
FO                                          0.0015
Pu                                          1.0
Acquisition rate (Period) [ns]              1.0
Half duration [Number of sigma]             3.0
Half pulse duration at half peak [ns]       2.0
Fraction at radius                          0.368

Axial division                              10
Max scattering order                        1
```

Figure 21. Example of simulating discrete point cloud with multi ray.

The above shows the LiDAR point cloud data simulated in two different modes: [**Single ray point cloud**] and [**Multi rays point cloud]**. Compared with [**Multi rays point cloud]** mode, [**Single ray point cloud**] provides more friendly simulation results, which is convenient for users to further analyze. [**Multi rays point cloud]** mode obtains finer results， In the above example, two different modes simulate the same scene, and the number of point clouds obtained by [**Single ray point cloud**] mode and [**Multi rays point cloud]** mode is 2091 and 6172 respectively.

Of course, in addition to the above two modes, LESS also provides an advanced mode of [**Multi rays point cloud]**. The user can use the mouse to indicate **[Advanced]** and click the [**Muti rays point cloud incident**] button (Figure 20). Like other simulation forms, users can find the results of advanced version simulation in the **Results** folder. The corresponding folder is **pointcloudIncident**. [**Muti rays point cloud incident**] mode is based on the development of [**Multi rays point cloud]** mode, Therefore, the simulation results not only record the results of [**Multi rays point cloud]** mode, but also record the incident energy of each pulse and the energy intercepted by ground objects during the pulse transmission, Incident energy and intercepted energy are recorded in the last two columns in the result file respectively. Different modes depend on the research needs of users, which also allows users to choose different modes of simulation data more flexibly.



X	Y	Z	PulseIndex	BandIndex	ReturnNum	NumReturns	Intensity	Incident	Intercept
53.0000	46.9918	0.0409	2000	1	1	1	1.67906e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	2	1	1	1.88476e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	3	1	1	2.03193e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	4	1	1	2.22007e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	5	1	1	2.40905e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	6	1	1	2.58800e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	7	1	1	2.70674e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	8	1	1	2.78701e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	9	1	1	2.84053e-06	1.00000e+00	1.00000e+00
53.0000	46.9918	0.0409	2000	10	1	1	2.90157e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	1	1	1	1.66238e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	2	1	1	1.86604e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	3	1	1	2.01175e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	4	1	1	2.19802e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	5	1	1	2.38512e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	6	1	1	2.56229e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	7	1	1	2.67985e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	8	1	1	2.75932e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	9	1	1	2.81231e-06	1.00000e+00	1.00000e+00
53.0000	48.0115	-0.0529	2001	10	1	1	2.87274e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	1	1	1	1.64468e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	2	1	1	1.84617e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	3	1	1	1.99032e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	4	1	1	2.17461e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	5	1	1	2.35972e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	6	1	1	2.53500e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	7	1	1	2.65131e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	8	1	1	2.72994e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	9	1	1	2.78236e-06	1.00000e+00	1.00000e+00
53.0000	48.9965	0.0151	2002	10	1	1	2.84215e-06	1.00000e+00	1.00000e+00
53.0000	50.0120	-0.0479	2003	1	1	1	1.62601e-06	1.00000e+00	1.00000e+00

Half duration [Number of sigma]	3.0
Half pulse duration at half peak [ns]	2.0
Fraction at radius	0.368
Axial division	10
Max scattering order	1

Figure 22. Example of simulating Muti rays point cloud with incident mode.

➢ Full-waveform Multi/Hyperspectral Lidar

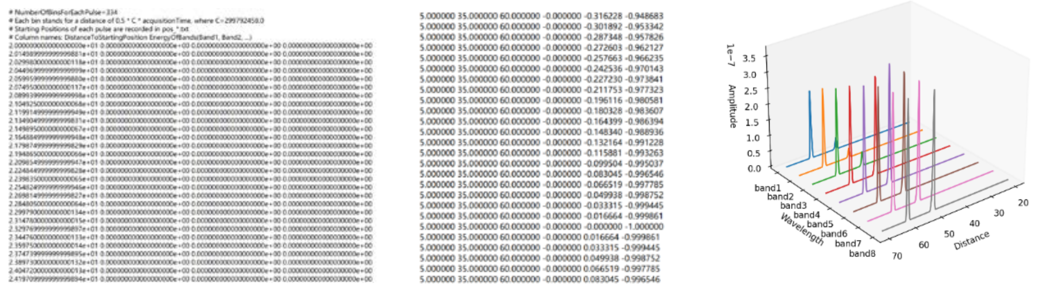
Using the mouse to indicate **[RUN]** and click the button **[Generate ALL]**, The simulator starts to generate 3D scenes and LiDAR configuration parameters. Then click button **[Single ray point cloud]** or button **[Multi rays waveform]** to simulate the data. The simulation results can be found in the **Results** folder. The results of two different modes correspond to **singleRay** and **records** folder. However, the result is only the data of intermediate process, which is not the full waveform data that can be used effectively. Using the mouse to indicate **[Process]** and click button **[Convolve]** to perform convolution operation. Finally, the full waveform data is obtained, and the results can be found in **convolved_waveform** folder. This result is composed of two files, It's **0.txt** and **pos_0.txt** respectively (**Figure 23**).

[**0.txt**] records all waveform data.

The number of pulses is the number of rows of recorded data divided by bins in which each pulse is dispersed. Therefore, the first pulse is 0 to the **NumberofBinsForEachPulse-1** line. The second pulse is **NumberofBinsForEachPulse to the 2* NumberofBinsForEachPulse-1** line. By analogy, all pulses can be read. The first column of the file is the distance from the starting position of each pulse bin to the sensor, and the second column to the last column is the echo intensity of each band. The visualization of full waveform multispectral LiDAR data is shown in.

[Pos_1.txt] records the position information of the pulse.

The first three columns are the XYZ coordinates of the pulse starting position, that is the sensor position. The last three columns are the reflection direction of each pulse, at the same time, the coordinate system is consistent with the coordinate system of LESS GUI.



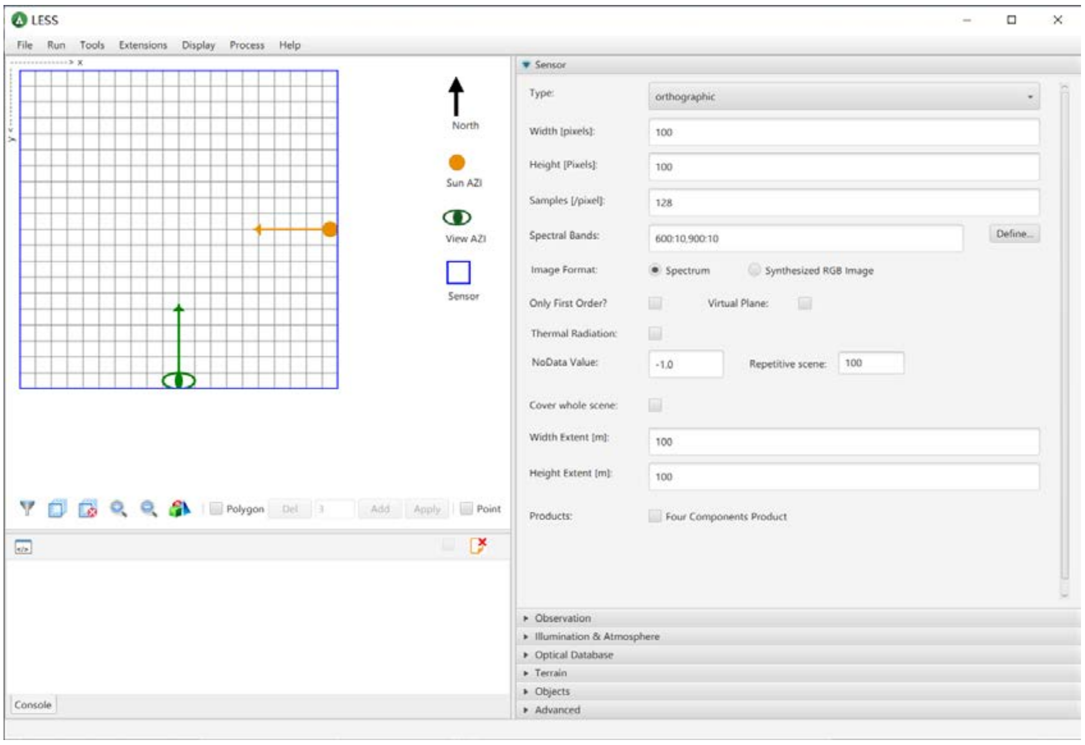
```
# NumberOfBinsForEachPulse=334
# Each bin stands for a distance of 0.5 * C * acquisitionTime, where C=299792458.0
# Starting Positions of each pulse are recorded in pos_*.txt
# Column names: DistanceToStartingPosition EnergyOfBands(Band1, Band2, ...)
2.000000000000000000e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.014989999999999881e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.029980000000000118e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.044969999999999999e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.059959999999999880e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.074950000000000117e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.089939999999999998e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.104925000000000068e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.119914999999999949e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.134904999999999831e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.149895000000000067e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.164884999999999948e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.179874999999999829e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.194865000000000066e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.209854999999999947e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.224844999999999828e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.239835000000000065e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.254824999999999946e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.269814999999999827e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.284805000000000064e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.299790000000000134e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.314780000000000015e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.329769999999999897e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.344760000000000133e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.359750000000000014e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.374739999999999895e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.389730000000000132e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.404720000000000013e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
2.419709999999999894e+01 0.000000000000000000e+00 0.000000000000000000e+00 0.000000000000000000e+00
```

```
5.000000 35.000000 60.000000 -0.000000 -0.316228 -0.948683
5.000000 35.000000 60.000000 -0.000000 -0.301892 -0.953342
5.000000 35.000000 60.000000 -0.000000 -0.287348 -0.957826
5.000000 35.000000 60.000000 -0.000000 -0.272603 -0.962127
5.000000 35.000000 60.000000 -0.000000 -0.257663 -0.966235
5.000000 35.000000 60.000000 -0.000000 -0.242536 -0.970143
5.000000 35.000000 60.000000 -0.000000 -0.227230 -0.973841
5.000000 35.000000 60.000000 -0.000000 -0.211753 -0.977323
5.000000 35.000000 60.000000 -0.000000 -0.196116 -0.980581
5.000000 35.000000 60.000000 -0.000000 -0.180328 -0.983607
5.000000 35.000000 60.000000 -0.000000 -0.164399 -0.986394
5.000000 35.000000 60.000000 -0.000000 -0.148340 -0.988936
5.000000 35.000000 60.000000 -0.000000 -0.132164 -0.991228
5.000000 35.000000 60.000000 -0.000000 -0.115881 -0.993263
5.000000 35.000000 60.000000 -0.000000 -0.099504 -0.995037
5.000000 35.000000 60.000000 -0.000000 -0.083045 -0.996546
5.000000 35.000000 60.000000 -0.000000 -0.066519 -0.997785
5.000000 35.000000 60.000000 -0.000000 -0.049938 -0.998752
5.000000 35.000000 60.000000 -0.000000 -0.033315 -0.999445
5.000000 35.000000 60.000000 -0.000000 -0.016664 -0.999861
5.000000 35.000000 60.000000 -0.000000 -0.000000 -1.000000
5.000000 35.000000 60.000000 -0.000000 0.016664 -0.999861
5.000000 35.000000 60.000000 -0.000000 0.033315 -0.999445
5.000000 35.000000 60.000000 -0.000000 0.049938 -0.998752
5.000000 35.000000 60.000000 -0.000000 0.066519 -0.997785
5.000000 35.000000 60.000000 -0.000000 0.083045 -0.996546
```

Figure 23. Example of 0.txt (left); example of pos_0.txt (middle); simulated hyperspectral waveform (right).

5.1.3 Simulations of Multi-Spectral Image

Open the LESS main program (as shown in **Figure 24**), and then create a new simulation project by selecting **[File]-> [New Simulation].** Create a new directory named “Ex01” and choose it as the directory for the project. If the project is created successfully, you will see the message “Succeed: ‘save path’” in the **[Console]** area.



Sensor

Type:	orthographic
Width [pixels]:	100
Height [Pixels]:	100
Samples [/pixel]:	128
Spectral Bands:	600:10,900:10
Image Format:	[✓ Spectrum] Synthesized RGB Image
Only First Order?	☐
Virtual Plane:	☐
Thermal Radiation:	☐
NoData Value:	-1.0
Repetitive scene:	100
Cover whole scene:	☐
Width Extent [m]:	100
Height Extent [m]:	100
Products:	☐ Four Components Product

Observation
Illumination & Atmosphere
Optical Database
Terrain
Objects
Advanced

Figure 24. LESS main program window

Define the parameters related to the sensor in the **[sensor]** window. The main parameters and their descriptions are as follows (default parameters can be kept for those not mentioned):

Parameter	Number	Explanation
Type	Orthographic	Sensor type, where Orthographic represents a parallel projection camera.
Width [pixels]	500	The final simulated image width in pixels.
Height [pixels]	500	The final simulated image height in pixels.
Samples [/pixel]	32	The number of rays emitted per pixel, which is generally related to the complexity of the scene, with common values such as 32, 64, 128, etc. A larger value will result in more accurate results, but

		will also increase computation time. (When the spatial resolution is 1m, it can be set to 64).
Spectral Bands	600:2,900:2	The simulated bands are 600nm and 900nm, with a band width of 2nm.
Image Format	Spectrum	The output format of the final image, where Spectrum represents outputting the multi-band format and saving it in the ENVI standard format. RGB image outputs a png format image that is automatically synthesized by the program in visible light.
Only First Order	Uncheck	If selected, it means that only a single scatter is simulated, and the multiple scattering term is ignored.
NoData Value	-1.0	If the sensor field of view exceeds the scene size, the pixel values of the area outside the range are set to the specified value.
Repetitive scene	100	The horizontal expansion of the scene to simulate the scenario of extending infinitely in the horizontal direction.
Width Extent [m]	100	The field of view size for a parallel projection sensor (other types of sensors do not have this parameter).
Height Extent [m]	100	The field of view size for a parallel projection sensor.

- **【Observation】**

For a parallel projection camera, the main parameters to be set for the observation geometry are the **[view zenith]** angle and the **[view azimuth]** angle. The azimuth angle increases in a clockwise direction starting from the north direction of 0°, with the south direction being 180°. In LESS, the field of view for the parallel projection camera is actually a cube, with the scene contained within the cube. The final image plane is the bottom of the cube, and the distance between the image plane and the scene center is determined by the **[sensor height]** .(**Figure 25**)

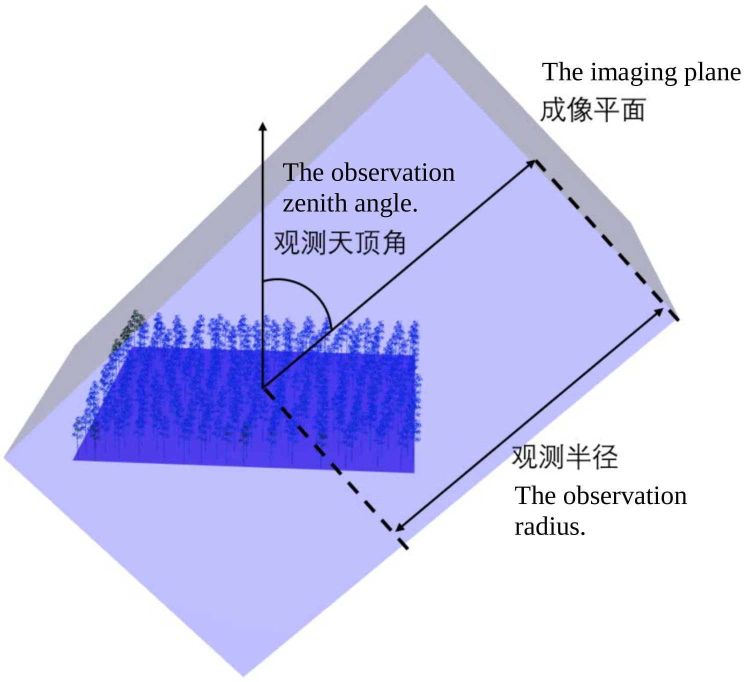


Figure 25. Field of view for a parallel projection camera.

- **【Illumination & Atmosphere】**

Parameter	Number	Explanation
Sun Zenith	45	Solar zenith angle.
Sun Azimuth	90	Solar azimuth angle.
Sky-Type	SKY_TO_TOTAL	Type of scattered light in the sky: SKY_TO_TOTAL represents the proportion of sky light to total incident light.
Sky-Percentage	0,0	Proportion of sky light incident in each band.

Note: If "Input solar spectrum manually" is checked,the solar and sky incident spectra can be manually entered.

- **【Optical Database】**

In this exercise, the default values can be kept. In fact, the Optical Database defines a series of spectra for setting the optical properties of the ground and scene elements. In LESS, spectra include front reflectance, back reflectance, and transmittance. "birch_branch", "dark_soil_mollisol", and "birch_leaf_green" are the default spectra in LESS, and should not be changed.

- **【Terrain】**

Parameter	Number	Explanation
Type	PLANE	PLANE: Flat terrain; RASTER: Use ENVI raster image as terrain; MESH: Use OBJ file as terrain;
XSize	100	Scene Size - Width in meters
Ysize	100	Scene Size - Height in meters
BRDF Type	Lambertian	Ground has Lambertian reflectance
Optical Property	dark_soil_mollisol	This list shows all the spectral curves defined in the [Optical Database].

- **【Objects】**

Click the **[Define Object]** button, and the window shown in the figure below will pop up. Enter "tree" on the far left and click **[add]** to add an object named "tree". Select the object, click **[Import OBJ]** in the middle window, and select an OBJ file. LESS comes with some OBJ files in the installation directory, such as the "BEPE_FROM_HET07_JPS_SUM.obj" file in "…\app\Database\3D_Objects\Trees\RAMI". After import, all groups defined in the OBJ file (defined by the "g" in the OBJ file) will be displayed. Select each group separately and set the corresponding **[Optical Property]**. "tree_leaves.obj" spectral property is set to "birch_leaf_green", and "tree_stem_branch.obj" spectral property is set to "birch_branch".(**Figure 26**)

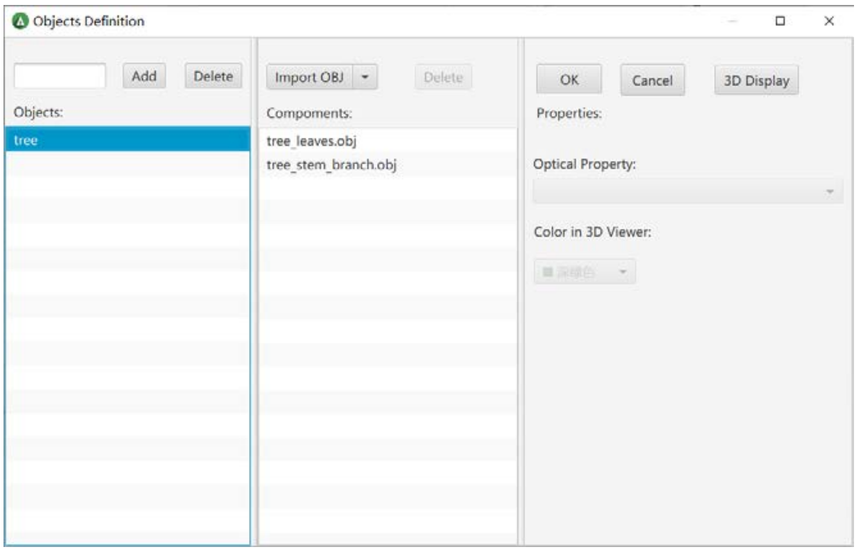
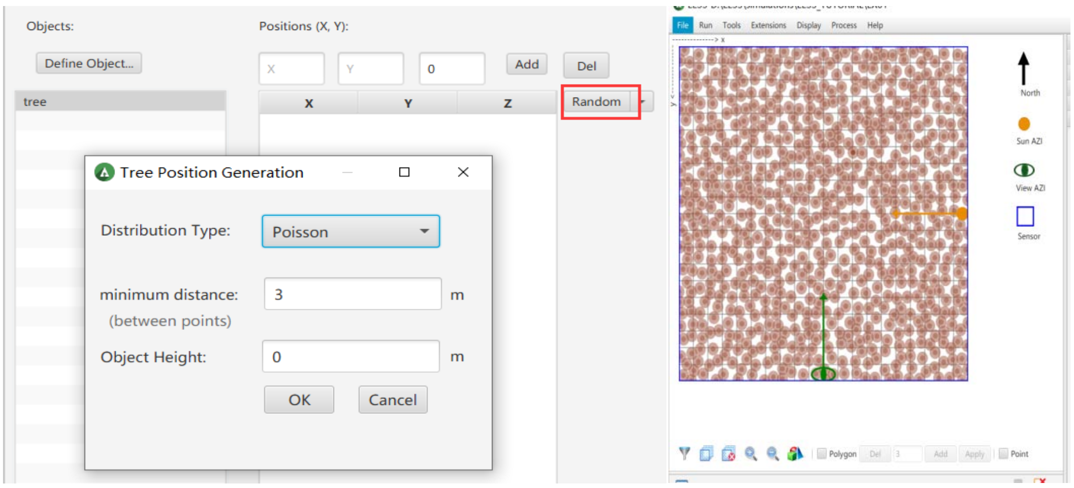


Figure 26. Define a scene Object

- **Define positions for scene elements**

Select the object (e.g. "tree") in the main window and click the **[Random]** button. In the pop-up window, define the minimum distance between any two trees (to avoid trees overlapping each other) and click **[OK]**. The "tree" object will then be randomly distributed in different positions in the scene.(**Figure 27**)



Distribution Type: Poisson

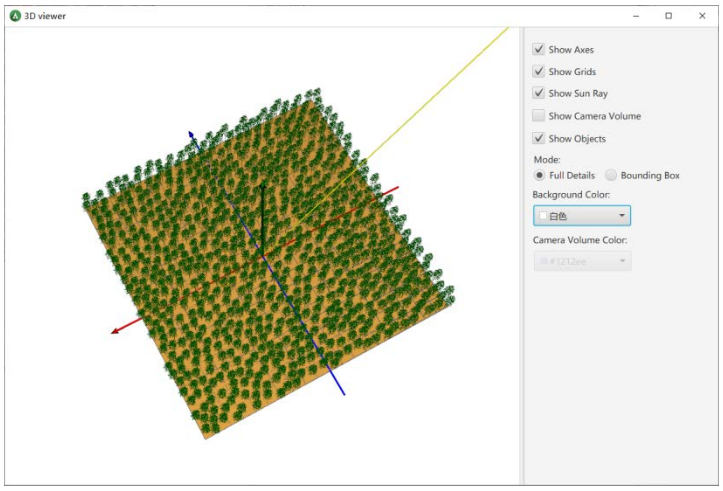
minimum distance: 3 m (between points)

Object Height: 0 m

Figure 27. Define the location of the scene Object

- **three-dimensional display**

Click the button[ ] on the main interface to display the 3D structure of the scene and check if the scene meets the expected goals.(**Figure 28**)



- [✓] Show Axes
- [✓] Show Grids
- [✓] Show Sun Ray
- [ ] Show Camera Volume
- [✓] Show Objects

Mode: [✓ Full Details] Bounding Box

Background Color: 白色

Camera Volume Color: #1212ee

Figure28. 3D Scene Visualization.

- **Image Simulation and Results Viewing:**

Click the menu **[Run] -> [Run All]** to start the simulation. After the simulation is complete, click the menu **[Tools] -> [Open Results Folder]** to open the simulation results directory. The file "spectral_VZ=0_VA=180" is the simulated radiance image. ENVI can be used to view this multispectral file.(**Figure 29**)

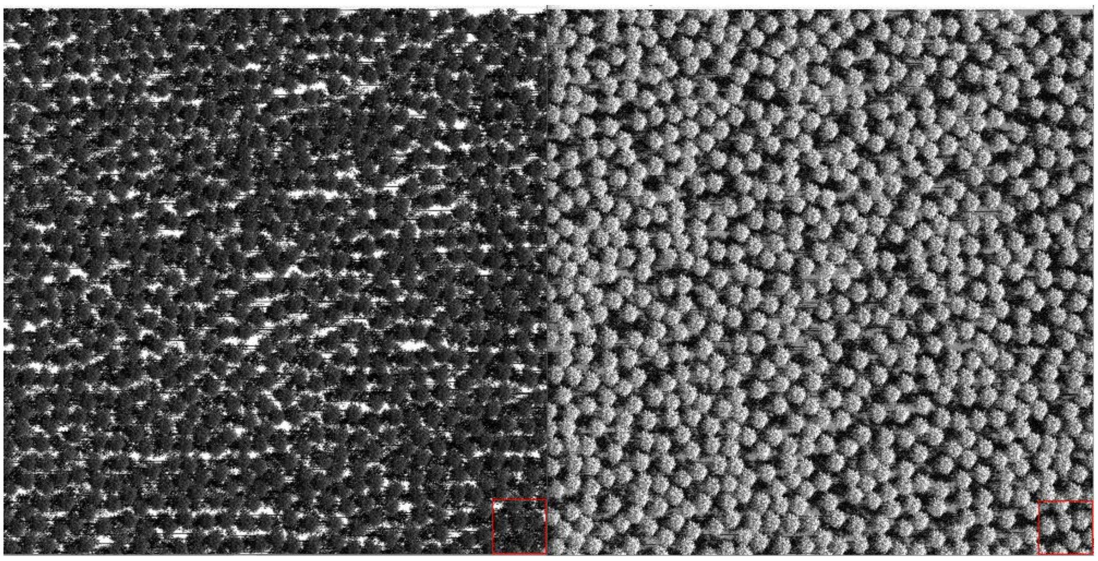


Figure 29. Analog images: Red (left) and near infrared (right)

- **Generating Reflectance Files**

Click the menu **[Process]-> [BRF Processing]** to convert the radiance image to a reflectance image. The result will be "spectralVZ=0VA=180_BRF".

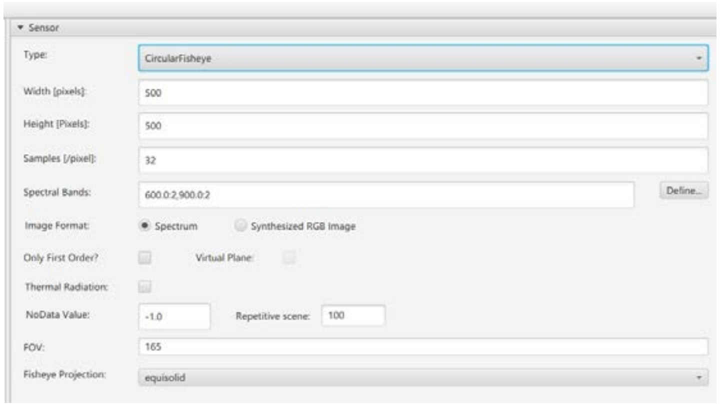
5.1.4 Fisheye camera simulation

- **Sensor Parameter Settings.**

To set the sensor parameters, select "CircularFisheye" in the **[Type]** menu. A new set of parameters will appear to define the characteristics of the fisheye camera. **[FOV]** represents the field of view of the fisheye camera, and **[Fisheye Projection]** represents the projection mode of the fisheye camera. There are four built-in projection modes in LESS:

- Equisolid: equal area projection
- Orthographic: orthographic projection
- Equidistant: equidistant projection
- Stereographic: stereographic projection

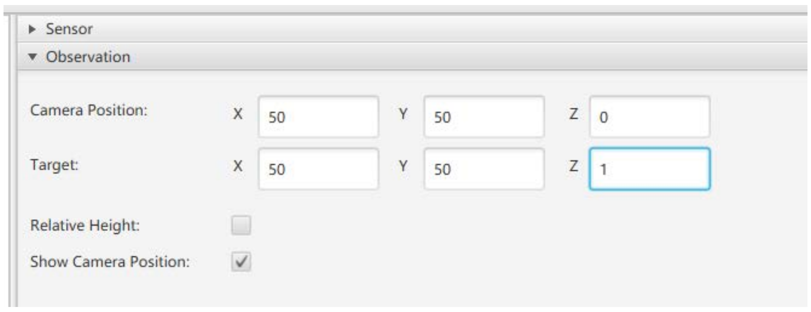
The commonly used projection mode is equisolid projection, which is also known as equal-area projection.(**Figure 30**)


▾ Sensor

Type:	CircularFisheye
Width [pixels]:	500
Height [Pixels]:	500
Samples [/pixel]:	32
Spectral Bands:	600:0.2,900:0.2 Define...
Image Format:	● Spectrum ○ Synthesized RGB Image
Only First Order?	☐ Virtual Plane: ☐
Thermal Radiation:	☐
NoData Value:	-1.0 Repetitive scene: 100
FOV:	165
Fisheye Projection:	equisolid

Figure30. Fisheye Camera Parameter Settings.

After selecting the fisheye camera, the observation mode of the camera has also changed. Therefore, it is necessary to reset the relevant parameters in **[Observation]**. **[Camera Position]** defines the position of the camera, while **[Target]** defines the observation point of the camera. The vector composed of these two points actually defines the observation direction of the camera. If **[Relative Height]** is checked, it means that the height Z of the AB point mentioned above is relative to the height above the ground (when there is terrain). In this exercise, A and B can be set to (50, 50, 0), (50, 50, 1), which means that the camera is observing towards the zenith direction from the center of the scene.(**Figure 31**)



Camera Position:	X	50	Y	50	Z	0
Target:	X	50	Y	50	Z	1
Relative Height:	☐					
Show Camera Position:	✓					

Figure31. Fisheye Camera Observation Parameter Settings

Setting **[NoData Value]** to 0 for simulation, the resulting fisheye camera photo at 900nm is shown in **Figure 32**

(file name "spectral_ox=50_00_oy=50_00_oz=0_00_tx=50_00_ty=50_00_tz=1_00").

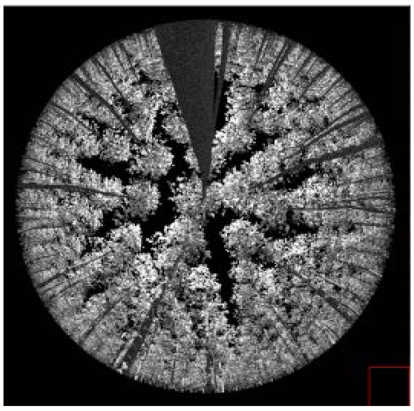


Figure32. Example of Fisheye Camera Photo.

5.1.5 Simulation of Multi-angle Reflectance.

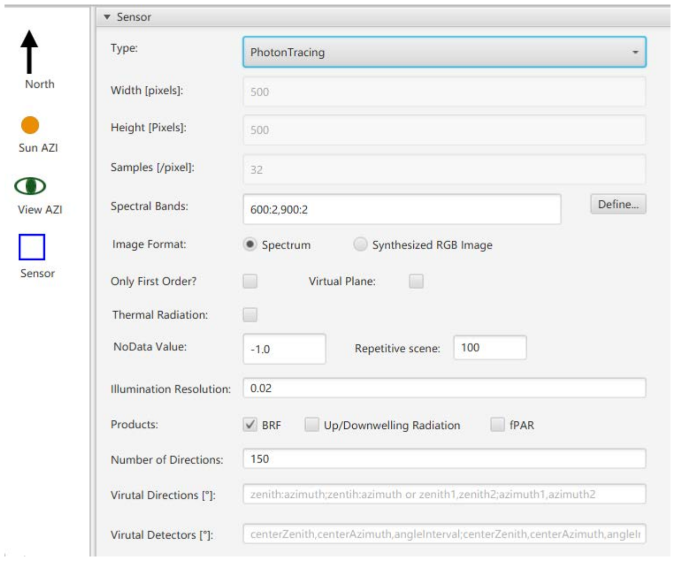
Although the simulated image can be converted to reflectance, this simulation method can only simulate one direction at a time. It is inefficient and inconvenient to simulate multiple angles of reflectance simultaneously. Therefore, the multi-angle reflectance simulation method in this exercise needs to be used.

- **Sensor parameter settings**

In the **[Type]** section, select "Photon Tracing" (**Figure 33**), which represents photon tracing simulation. In this mode, since no image is generated, the image width **[Width]**, height **[Height]**, and number of rays **[Samples]** do not need to be set. However, the [Illumination Resolution] needs to be set, which represents the density (spacing) of incident photons. The smaller the value, the denser the photons, and the more accurate the results, but the slower the calculation speed. In this exercise, it can be set to 0.05.

On the right side of **[Products]**, check "BRF", which will display the **[Number of Directions]**, **[Virtual Directions]**, and **[Virtual Detectors]**:

- **[Number of Directions]** represents the number of small surface elements (solid angles) that divide the upper hemisphere space. Each element will generate a corresponding reflectance. In this exercise, it can be set to 10.
- **[Virtual Directions]** represents the virtual directions used to calculate the directional reflectance factor BRF, expressed in zenith angle and azimuth angle. There are two ways to set this: one is "Zenith Angle 1: Azimuth Angle 1; Zenith Angle 2: Azimuth Angle 2", representing two virtual directions 11 and 22; the other is "Zenith Angle 1, Zenith Angle 2; Azimuth Angle 1, Azimuth Angle 2", representing four directions combined in pairs (11, 12, 21, 22). In this exercise, it can be set to "45:90;60:90".
- **[Virtual Detectors]** represents placing a virtual detector with a certain size in a certain direction to simulate directional reflectance with a certain field of view effect. This option is generally less frequently used and can be left blank.

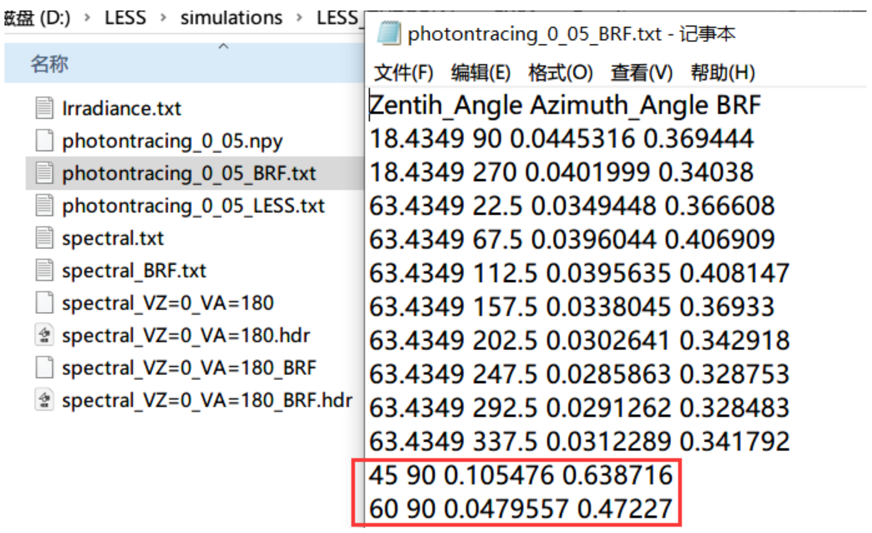


Type:	PhotonTracing
Width [pixels]:	500
Height [Pixels]:	500
Samples [/pixel]:	32
Spectral Bands:	600:2,900:2
Image Format:	[✓ Spectrum] Synthesized RGB Image
Only First Order?	☐ Virtual Plane: ☐
Thermal Radiation:	☐
NoData Value:	-1.0 Repetitive scene: 100
Illumination Resolution:	0.02
Products:	✓ BRF ☐ Up/Downwelling Radiation ☐ fPAR
Number of Directions:	150
Virutal Directions [°]:	zenith:azimuth;zentih:azimuth or zenith1,zenith2;azimuth1,azimuth2
Virutal Detectors [°]:	centerZenith,centerAzimuth,angleInterval;centerZenith,centerAzimuth,angleI

Figure33. BRF simulating sensor parameter Settings

- **BRF simulation**

Click on the menu **[Run] -> [Run All]** to start the simulation. After the simulation is completed, click on the menu **[Tools] -> [Open Results Folder]** to open the simulation result directory. The file "photontracing_0_05_BRF.txt" saves the simulated BRF results (**Figure 34**). The first column is the zenith angle, the second column is the azimuth angle, and starting from the third column, it corresponds to the reflectance factor BRF of each band. The first 10 rows of results correspond to the BRF of the number of elements set by **[Number of Directions]**, and the last two rows correspond to the BRF of the virtual directions set by **[Number of Directions]**. Generally speaking, BRF estimated from virtual directions are more accurate.



磁盘 (D:) › LESS › simulations › LESS
名称
Irradiance.txt
photontracing_0_05.npy
photontracing_0_05_BRF.txt
photontracing_0_05_LESS.txt
spectral.txt
spectral_BRF.txt
spectral_VZ=0_VA=180
spectral_VZ=0_VA=180.hdr
spectral_VZ=0_VA=180_BRF
spectral_VZ=0_VA=180_BRF.hdr
photontracing_0_05_BRF.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
Zentih_Angle Azimuth_Angle BRF
18.4349 90 0.0445316 0.369444
18.4349 270 0.0401999 0.34038
63.4349 22.5 0.0349448 0.366608
63.4349 67.5 0.0396044 0.406909
63.4349 112.5 0.0395635 0.408147
63.4349 157.5 0.0338045 0.36933
63.4349 202.5 0.0302641 0.342918
63.4349 247.5 0.0285863 0.328753
63.4349 292.5 0.0291262 0.328483
63.4349 337.5 0.0312289 0.341792
45 90 0.105476 0.638716
60 90 0.0479557 0.47227
```

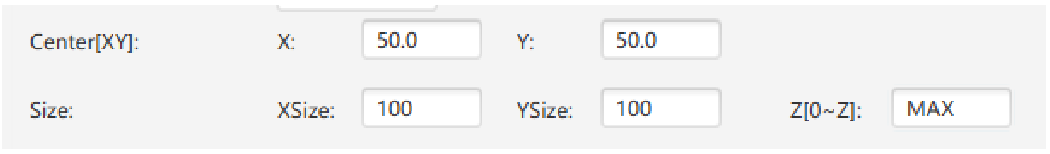
Figure34. BRF simulation results

- **Simulating reflectance**

In the result folder, "photontracing_0_05_LESS.txt" saves the reflectance values of the scene, which is the integral of the accumulated energy over all elements in the upper hemisphere of the space.

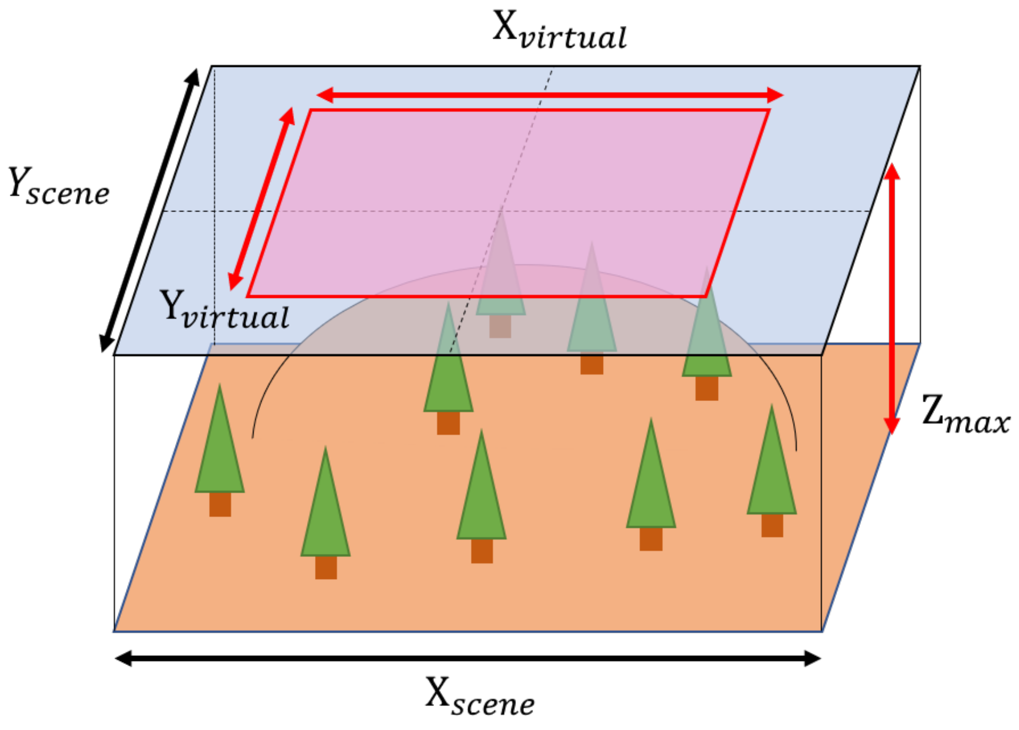
Virtual plane

In the **[Sensor]** menu, there is a **[Virtual Plane]** option. After checking it, a new set of parameters will appear, which can define a virtual plane located at the top of the scene with a certain range. When simulating the BRF of the scene, the total incident and outgoing energy will be calculated based on the range of the quadrilateral formed by the virtual plane (i.e., only photons passing through the quadrilateral will be collected). The virtual plane has a plane center position (**[X]** and **[Y]**) and size (**[XSize]** and **[YSize]**), and the default height is the highest point of the scene, so the value of **[Z]** here ("MAX") cannot be modified.

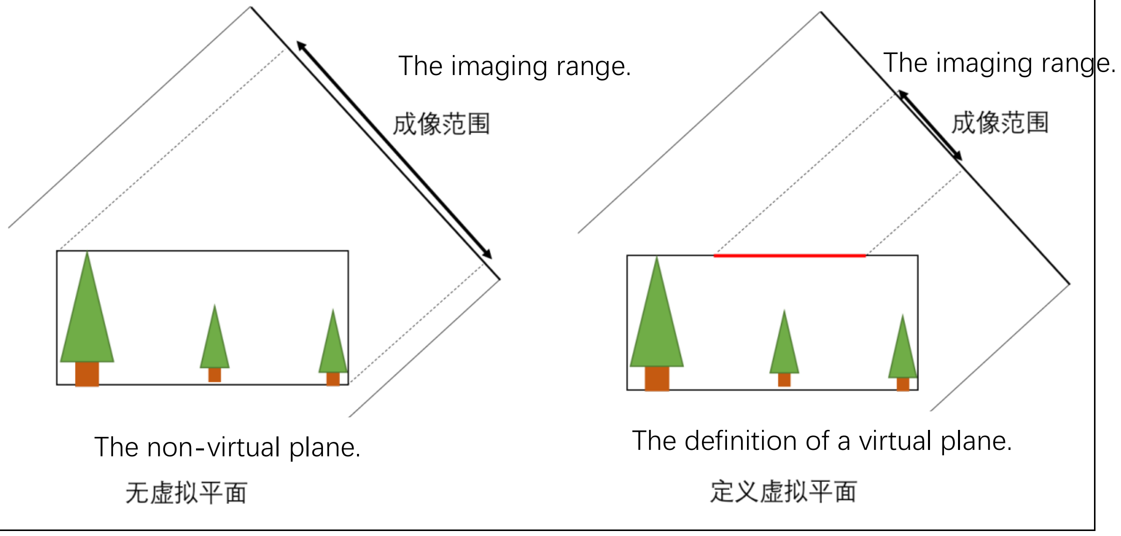


Center[XY]:	X:	50.0	Y:	50.0		
Size:	XSize:	100	YSize:	100	Z[0~Z]:	MAX

In Photontracing mode, if a virtual plane is not defined, the top plane of the scene's bounding box will be used as the default virtual plane.



In orthographic camera mode, if a virtual plane is not defined, there is no virtual plane by default. When a virtual plane is defined, the imaging range is the entire scene without a virtual plane, and the imaging range is the scene contained in the virtual plane when a virtual plane is present. This virtual plane plays a very important role in calculating the reflectivity of the canopy and the gap fraction between directions, which can avoid the impact of scene sidewalls.



5.1.6 Directional Gap Fraction simulation

The Gap Fraction is generally used to describe the probability of gaps in the vegetation canopy in a particular direction. In LESS, it can be obtained by simulating an image and then classifying it, but a more accurate way is to use four-component images to calculate the proportion of tree canopy gaps.

- **Parameter Settings**

Select “Orthographic” in **[Sensor] -> [Type]**; set the number of **[spectral bands]** to be greater than or equal to 5, for example, “630.0:2,690.0:2,750.0:2,810.0:2,870.0:2”; check **[Virtual Plane]**; check the “Four Components Product” on the right side of **[Products]**;

In **[Observation]**, set **[View Zenith]** and **[View Azimuth]** to the desired direction, for example, 0 and 180;

- **Four-Component Image Simulation**

Click on the menu **[Run]-> [Run All]** to start the simulation. After the simulation is completed, click on the menu **[Tools] -> [Open Results Folder]** to open the simulation result directory. The file “spectral_VZ=0_VA=180_4Components” is the four-component image (Figure 34). This file contains a total of five spectral bands:

 Band 1 is the four-component category to which each pixel belongs:

1-illuminated soil;

2-illuminated leaves;

3-shaded soil;

4-shaded leaves;

here, categories are determined based on the component with the largest proportion in the pixel.

 Band 2 indicates the proportion of illuminated soil in each pixel, with a value between 0 and 1;

 Band 3 indicates the proportion of illuminated leaves in each pixel;

 Band 4 indicates the proportion of shaded soil in each pixel;

 Band 5 indicates the proportion of shaded leaves in each pixel.

- **Calculation of Gap Fraction**

Therefore, the formula for calculating the Gap Fraction is mean (b2+b4), which is the sum of Band 2 and Band 4, and then averaged over all pixels.

For ease of calculation, LESS provides a Python script “DirectionalGapProb.py”, located in the directory “[Installation Path] \LESS\app\Python_script”. The script can be run using the Python interpreter provided with LESS, for example: [Installation Path]\LESS\app\bin\python [Installation Path]\LESS\app\Python_script\DirectionalGapProb.py -i[LESS output result path]\spectral_VZ=0_VA=180_4Components -o [Custom result save path]\gaps.txt. The final result is saved in the “gaps.txt” file. If there are multiple output files, the script can be called using wildcard characters.

For example, all files containing “4Components” can be processed simultaneously [InstallationPath]\LESS\app\bin\python[InstallationPath]\LESS\app\Python_script\DirectionalGapProb.py -i[LESS output result path]\spectral_VZ=_VA=_4Components -o [Custom result save path]\gaps.txt.(**Figure 35**)

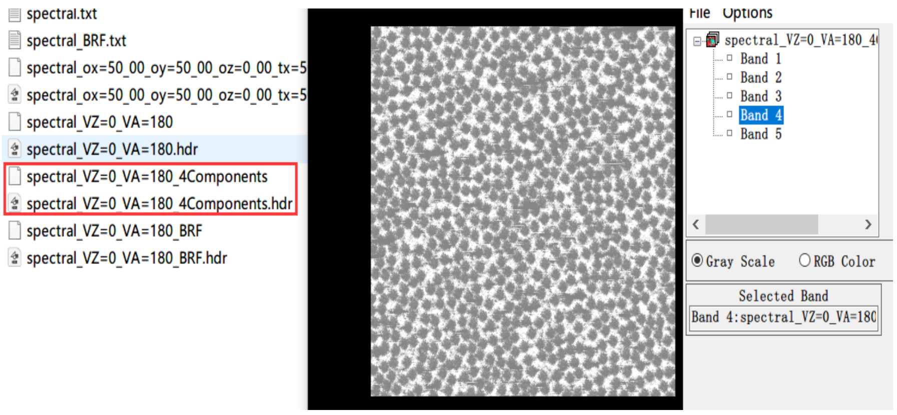

spectral.txt
spectral_BRF.txt
spectral_ox=50_00_oy=50_00_oz=0_00_tx=5
spectral_ox=50_00_oy=50_00_oz=0_00_tx=5
spectral_VZ=0_VA=180
spectral_VZ=0_VA=180.hdr
spectral_VZ=0_VA=180_4Components
spectral_VZ=0_VA=180_4Components.hdr
spectral_VZ=0_VA=180_BRF
spectral_VZ=0_VA=180_BRF.hdr
File Options
spectral_VZ=0_VA=180_4
Band 1
Band 2
Band 3
Band 4
Band 5
Gray Scale
RGB Color
Selected Band
Band 4:spectral_VZ=0_VA=180

Figure 35. Four component simulation results

5.1.7 Thermal Infrared Image Simulation

- **Parameter Settings**

To simulate thermal infrared images, select "Orthographic" in **[Sensor]**, check "Thermal Radiation", and set the **[spectral bands]** to 14um, for example, "14000:1".

In **[Optical Databse]**, a series of component temperatures can be defined. For example, "T300" is defined as "300:5", which means that the temperature of scene elements under sunlight is 302.5K, and the temperature in the shaded area is 197.5K. Similarly, another set of component temperatures "T290" can be defined with the value of "290:8".

At this point, the **[Temperature]** option will appear under **[Terrain]**, select "T290". Then open **[Define Objects]** under **[Object]** and set the temperature of "treeleaves.obj" to "T300" and the temperature of "treestem_branch.obj" to "T290".

- Image simulation

Click on the menu **[Run] -> [Run All]** to start the simulation. After the simulation is completed, click on the menu **[Tools] ->[Open Results Folder]** to open the simulation result directory. The file "thermalVZ=0VA=180" is the simulated thermal infrared image. Open the image in ENVI, and each pixel value represents the directional brightness temperature.(**Figure 36**)

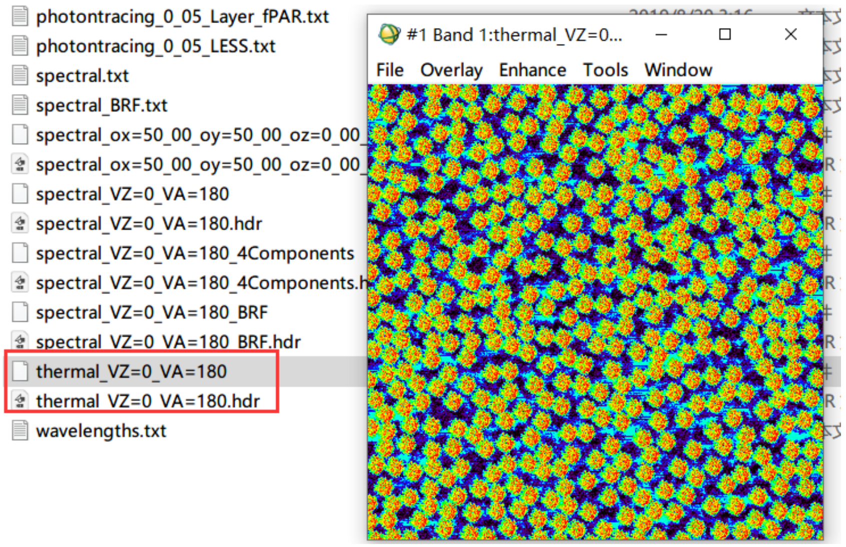

photontracing_0_05_Layer_fPAR.txt
photontracing_0_05_LESS.txt
spectral.txt
spectral_BRF.txt
spectral_ox=50_00_oy=50_00_oz=0_00
spectral_ox=50_00_oy=50_00_oz=0_00
spectral_VZ=0_VA=180
spectral_VZ=0_VA=180.hdr
spectral_VZ=0_VA=180_4Components
spectral_VZ=0_VA=180_4Components.h
spectral_VZ=0_VA=180_BRF
spectral_VZ=0_VA=180_BRF.hdr
thermal_VZ=0_VA=180
thermal_VZ=0_VA=180.hdr
wavelengths.txt
#1 Band 1:thermal_VZ=0...
File Overlay Enhance Tools Window

Figure36. Thermal Infrared Simulation Results

5.1.8 Layered Illumination/Shaded Leaf Area Ratio*1

Layered Illumination/Shaded Leaf Area Ratio refers to the ratio of illuminated/shaded leaf area in each horizontal layer of the canopy. The area here refers to the absolute area of the leaves, which is different from the ratio of the four components (observation directions) in the four-component simulation.(**Figure** 37)

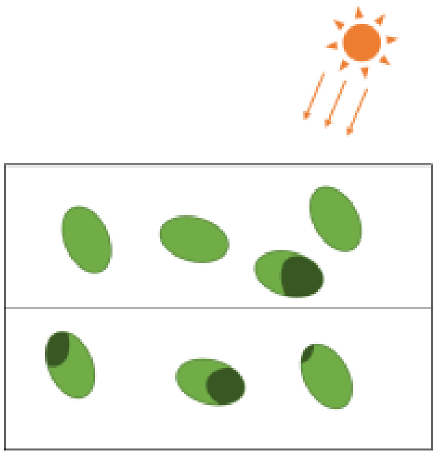
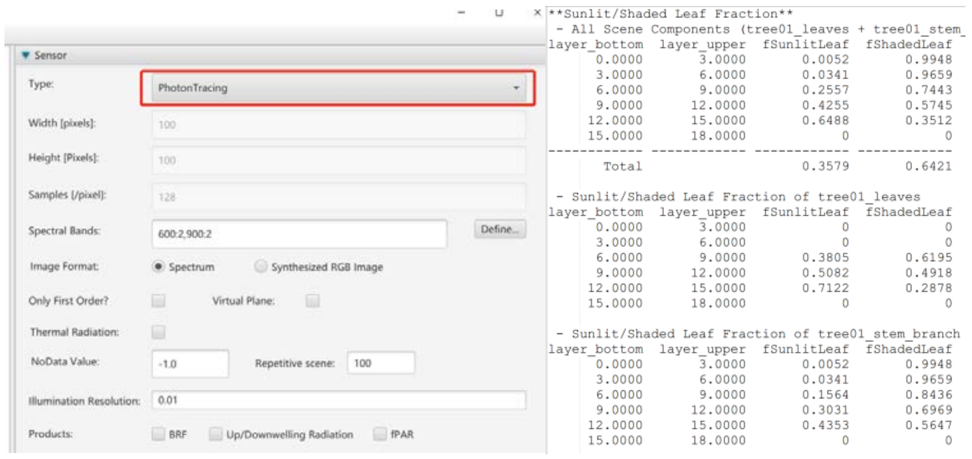


Figure37. Layered Illumination/Shaded Leaf Area Ratio

To enable the calculation of layered illumination/shaded leaf area ratio, create a file named “sunlit_leaf.conf” under the **[Parameters]** directory after creating a project and a scene, and select the PhotonTracing mode. The file content should be the number of layers separated by a colon, for example “0:1:20”, which means that the layers are separated every 1 meter from the altitude of 0 meters to 20 meters. The parameter that needs to be set is the **[Illumination Resolution]**. The smaller the value, the higher the calculation accuracy. Other products such as BRF and fPAR can be left unchecked. After executing the simulation, the “photontracing_*_*_Layer_fSunlitLeaf.txt” file will be generated under the **[Results]** directory, which records the layered illumination/shaded leaf area ratio. In the result file, “All Scene Components” represents the illumination/shaded ratio of all elements in the scene. The subsequent results are statistics for different scene components, such as leaves and branches in the scene. “Total” in the result represents the total illumination/shaded area ratio of the canopy, which needs to be noted that the total illumination/shaded ratio of the canopy is not a direct average of the ratio of each layer.（**Figure 38**）



Sensor

Type:	PhotonTracing
Width [pixels]:	100
Height [Pixels]:	100
Samples [/pixel]:	128
Spectral Bands:	600:2,900:2
Image Format:	[✓ Spectrum] Synthesized RGB Image
Only First Order?	☐
Virtual Plane:	☐
Thermal Radiation:	☐
NoData Value:	-1.0
Repetitive scene:	100
Illumination Resolution:	0.01
Products:	☐ BRF ☐ Up/Downwelling Radiation ☐ fPAR

```
**Sunlit/Shaded Leaf Fraction**
 - All Scene Components (tree01_leaves + tree01_stem_
layer_bottom  layer_upper  fSunlitLeaf  fShadedLeaf
      0.0000       3.0000       0.0052       0.9948
      3.0000       6.0000       0.0341       0.9659
      6.0000       9.0000       0.2557       0.7443
      9.0000      12.0000       0.4255       0.5745
     12.0000      15.0000       0.6488       0.3512
     15.0000      18.0000            0            0
------------ ------------ ------------ ------------
       Total                    0.3579       0.6421

 - Sunlit/Shaded Leaf Fraction of tree01_leaves
layer_bottom  layer_upper  fSunlitLeaf  fShadedLeaf
      0.0000       3.0000            0            0
      3.0000       6.0000            0            0
      6.0000       9.0000       0.3805       0.6195
      9.0000      12.0000       0.5082       0.4918
     12.0000      15.0000       0.7122       0.2878
     15.0000      18.0000            0            0

 - Sunlit/Shaded Leaf Fraction of tree01_stem_branch
layer_bottom  layer_upper  fSunlitLeaf  fShadedLeaf
      0.0000       3.0000       0.0052       0.9948
      3.0000       6.0000       0.0341       0.9659
      6.0000       9.0000       0.1564       0.8436
      9.0000      12.0000       0.3031       0.6969
     12.0000      15.0000       0.4353       0.5647
     15.0000      18.0000            0            0
```

Figure 38. Layered Illumination/Shaded Leaf Area Ratio calculation interface and results

[1] 带星号功能为实验功能：实验功能是指目前在 LESS 中已经实现，但是处于测试阶段或者使用人数较少，为简化程序界面，因此未加入到 GUI 中，可通过手工配置参数开启。

Case

The example result of the layered illumination/shaded leaf area ratio for a simple scene is shown in **Figure 39**. In this scene, there are two large leaves at a height of 1m and 2m from the ground, respectively, and their vertical projections completely overlap.

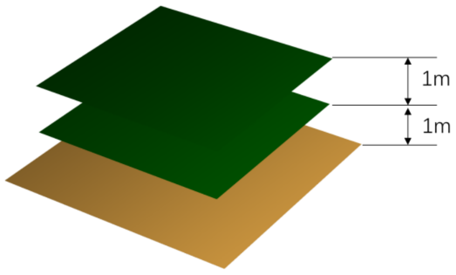


Figure 39. Example Result of Layered Illumination/Shaded Leaf Area Ratio

If the sun is vertically overhead, the calculated results are as follows:

```
**Sunlit/Shaded Leaf Fraction**
 - All Scene Components (plane_Group)
layer_bottom  layer_upper  fSunlitLeaf  fShadedLeaf
      0.0000       1.5000       0.0000       1.0000
      1.5000       3.0000       1.0000       0.0000
------------ ------------ ------------ ------------
       Total                    0.5000       0.5000
```

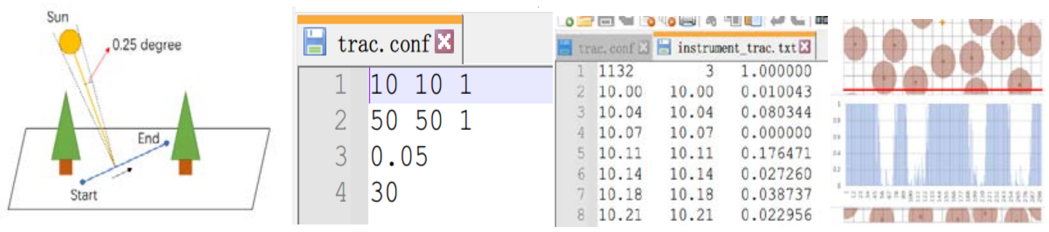
The lower leaves are completely shaded, while the upper leaves receive full illumination, resulting in a total canopy illumination ratio of 0.5. If the sun is at an elevation angle of 45° and azimuth angle of 90° (east direction), the calculated results would be as follows:

```
**Sunlit/Shaded Leaf Fraction**
 - All Scene Components (plane_Group)
layer_bottom  layer_upper  fSunlitLeaf  fShadedLeaf
      0.0000       1.5000       0.2500       0.7500
      1.5000       3.0000       1.0000       0.0000
------------ ------------ ------------ ------------
       Total                    0.6250       0.3750
```

5.1.9 TRAC simulation*

TRAC is a commonly used instrument for measuring Leaf Area Index (LAI), which records the direct solar radiation transmittance. The measurement process typically involves selecting several sample lines in a plot, and TRAC measures the canopy transmittance at a uniform time interval as the observer walks along the line. In LESS, the transmittance at each position on a simulated sample line can be calculated at equidistant intervals. The transmittance is determined by emitting rays in the direction of the sun, while taking into account the sun's apparent radius (0.25°).

To use TRAC, a configuration file named "trac.conf" should be created in the **[Parameters]** directory. The file content should include the starting point of the sample line (first line), the end point of the sample line (second line), the distance interval (third line), and the magnitude of the number of rays used to calculate the transmittance at each position along the line (fourth line). It should be noted that the number of rays here only represents a magnitude, and if the value is set to 30, the number of emitted rays will be approximately 900.(**Figure 40**)



```
10 10 1
50 50 1
0.05
30
```

```
1132            3    1.000000
10.00       10.00    0.010043
10.04       10.04    0.080344
10.07       10.07    0.000000
10.11       10.11    0.176471
10.14       10.14    0.027260
10.18       10.18    0.038737
10.21       10.21    0.022956
```

Figure 40. TRAC simulation examples

After the simulation is completed, the results are saved in the "instrument_trac.txt" file in the "Results" directory. The first number in the first row of the file represents the total number of points simulated, followed by the number of columns in the results (3 in this case), and the value "1.000000" represents the height (z-value) set in the trac.conf file. From the second row onwards, the first three columns represent the x-coordinate, y-coordinate, and transmittance value at each point.

5.1.10 Fisheye image transmittance simulation*

Fisheye image transmittance refers to simulating a fisheye image where each pixel represents the transmittance of light. In the image, a value of 1 represents the transmittance of the sky, a value of 0 represents complete pixel obstruction, and values between 0 and 1 represent partially obstructed pixels. It should be noted that this simulation also supports the turbid medium mode, where the transmittance of the turbid medium is calculated. This mode calculates the transmittance of direct sunlight in conjunction with LAD (Leaf Area Density), which is the unit of measure for the turbid medium. In the turbid medium mode, the radiative transfer equation is recalculated, and the transmittance values differ slightly from those obtained in the clear sky mode. During the fisheye image transmittance simulation, the transmission and scattering of light are calculated through three sections: the atmosphere, vegetation, and the ground, with transmittance calculated during this transmission process. When simulating at dawn or dusk, the scattering of light near the ground is greater, which affects the calculation of transmittance. Additionally, when a ray of light hits a pixel, it may be covered by multiple transmission paths, so the weighted average of transmittance needs to be considered. that is $e^{-G \cdot LAD \cdot t}$ (LAD is the leaf area density).

This simulation is set up exactly as the fisheye camera simulation in section 2.1.2, except that you need to create a dtrans.conf file under the Parameters folder to enable this mode, otherwise the simulation results are exactly as 5.1.4 (i.e., the simulation gets radiance information for each pixel).(**Figure 41**)

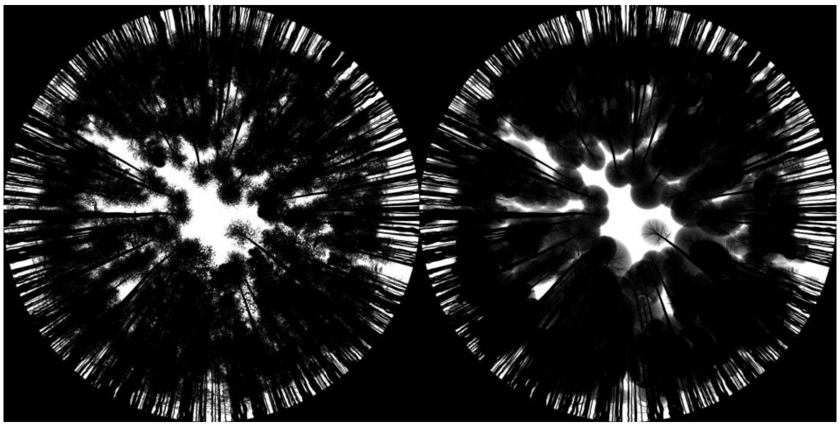
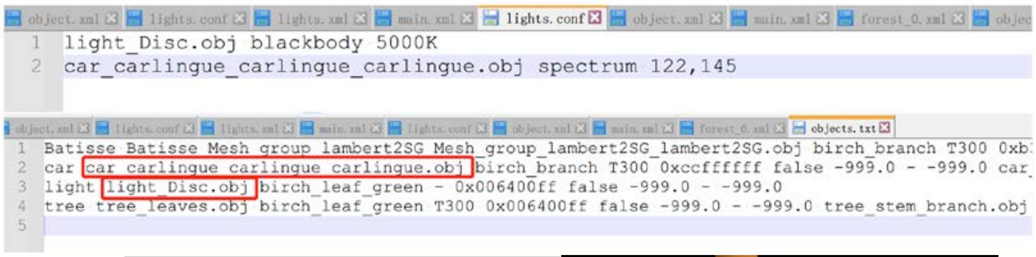


Figure 41. Examples of image simulation of transmittance of fisheye camera

5.1.11 Light simulation*

In LESS, certain objects can be set as light sources with a certain emission power. Therefore, scenes with night-time lighting can be simulated. To use the lighting simulation in LESS, you can construct a scene as normal and then create a file named “lights.conf” under the **[Parameters]** directory. The file contains the emission spectra for a component, which can be defined through blackbody emission or direct input spectra. For example, “lights.conf” contains two lines, with the first line defining the component “light_Disc.obj” in the scene as having a blackbody emission with a temperature of 5000K. LESS will automatically calculate the required emission spectra for the simulated bands based on the temperature and blackbody emission. The second line defines the spectrum using “spectrum,” with the spectral bands needing to match the number of bands defined in the sensor, and its units in w/m2/sr/nm. The name of the component can be found in the objects.txt file or GUI. By default, optics in LESS only emit from the front-facing side of the surface based on the normal vector, which is defined using the right-hand rule. Therefore, attention should be paid to the definition of the obj file’s normal vectors during the simulation.(**Figure 42**)



```
light_Disc.obj blackbody 5000K
car_carlingue_carlingue_carlingue.obj spectrum 122,145
```

```
Batisse Batisse Mesh_group_lambert2SG Mesh_group_lambert2SG_lambert2SG.obj birch_branch T300 0xb
car car_carlingue_carlingue_carlingue.obj birch_branch T300 0xccffffff false -999.0 - -999.0 car_
light light_Disc.obj birch_leaf_green - 0x006400ff false -999.0 - -999.0
tree tree_leaves.obj birch_leaf_green T300 0x006400ff false -999.0 - -999.0 tree_stem_branch.obj
```

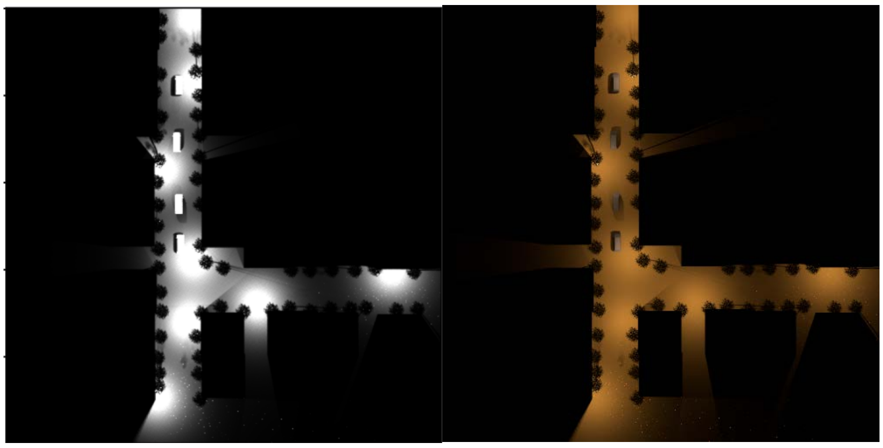
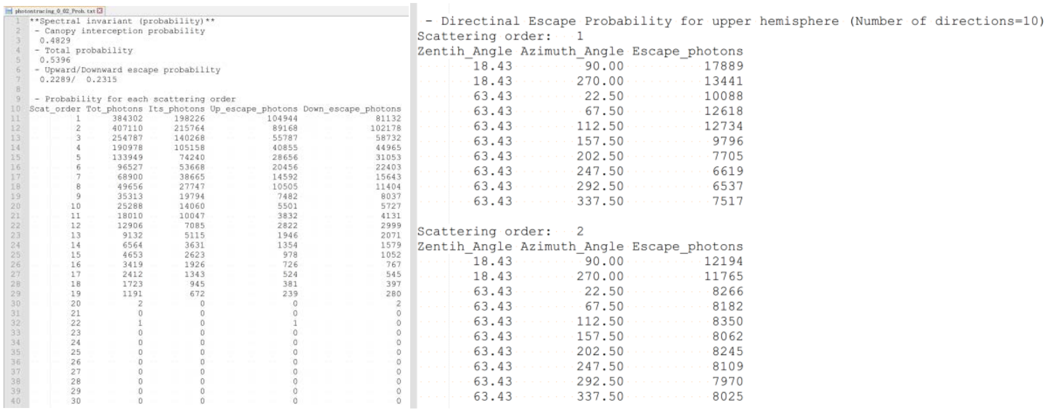


Figure 42. Lighting simulation parameter setting

5.1.12 Re-collision probability simulation

Recollision probability refers to the probability of the photon hitting the canopy again after scattering with the canopy, which is the core concept of spectral invariant theory. LESS provides parameters that are closely related to spectral invariant theory, such as simulated recollision probability, interception rate, hemisphere and directional escape probability. In LESS, the simulation of re-collision probability is carried out at the same time as the FPAR simulation, so after selecting **[PhotonTracing]** in **[Sensor]**, it is necessary to check **[fPAR & Recollision Prob.]** (If only the re-collision probability is simulated, Don't worry about the value of the Layer Definition). After the simulation, the *._prob. txt file will be generated in the result folder, and the result is shown in the figure below, where Total probability represents the total recollision probability of the canopy and is the sum of all scattering times. Canopy interception probability represents the total interception rate of incident light, that is, the probability that a scattering is intercepted by the canopy. Upward/Downward escape probability represents the probability of space escape in the upper and lower hemispheres of the canopy. Escape probability is defined as the probability of photon escaping from the canopy after scattering with the canopy, and the value is the sum of multiple scattering. Recollision probability + upper hemisphere + lower hemisphere escape probability =1. In addition to the total probability, the output also provides the number of photons scattered each time, from which the probability of the number of scatterings can be calculated or accumulated to get the total probability, Scat_order represents the number of scatterings, Tot_photons represents the total number of photons scattering from the canopy, Its represents the number of photons colliding with the canopy again, and Up_escape/Down_escape represents the number of photons escaping from the upper and lower hemispheres.

The upper hemisphere escape probability is further subdivided into the directional escape probability, which provides the number of escaped photons in each direction under each scattering number, according to which the probability can be calculated (right of the figure below). The Escape_photons represent the total number of escaped photons in this solid Angle, where the sum of all directions equals the total hemispheres escaped. Number of directions (N) indicates that the upper hemisphere space is evenly divided into equal solid Angle directions, the size of each solid Angle is $\frac{2\pi}{N}$, Zenith_Angle/Azimuth_Angle indicates the central direction of each solid Angle, Azimuth_Angle is the range clockwise from north [0° ,360°]. According to the central direction provided, the azimuth range of zenith Angle of each solid Angle can be inversely calculated. For example, as shown in the figure below, the upper hemisphere space is divided into 10 directions, and zenith Angle and azimuth Angle both start from 0, then 18.43 represents the zenith Angle range of this day [0, 36.86]. From this we can calculate the next zenith Angle range as [36.86, 63.43+(63.43-36.86)]=[36.86, 90.00], and so on. The number of directions for the escape probability of the upper hemisphere direction can be set in the following way (the default value is 1). In the **[Parameters]** directory, create a file named p_prob.conf with the number of directions in the file, for example, "100", then the corresponding number of directions will be displayed in the output file.(**Figure 43**)



```
**Spectral invariant (probability)**
 - Canopy interception probability
  0.4829
 - Total probability
  0.5396
 - Upward/Downward escape probability
  0.2289/  0.2315

 - Probability for each scattering order
Scat_order Tot_photons Its_photons Up_escape_photons Down_escape_photons
         1      384302      198226            104944               81132
         2      407110      215764             89168              102178
         3      254787      140268             55787               58732
         4      190978      105158             40855               44965
         5      133949       74240             28656               31053
         6       96527       53668             20456               22403
         7       68900       38665             14592               15643
         8       49656       27747             10505               11404
         9       35313       19794              7482                8037
        10       25288       14060              5501                5727
        11       18010       10047              3832                4131
        12       12906        7085              2822                2999
        13        9132        5115              1946                2071
        14        6564        3631              1354                1579
        15        4653        2623               978                1052
        16        3419        1926               726                 767
        17        2412        1343               524                 545
        18        1723         945               381                 397
        19        1191         672               239                 280
        20           2           0                 0                   2
        21           0           0                 0                   0
        22           1           0                 1                   0
        23           0           0                 0                   0
        24           0           0                 0                   0
        25           0           0                 0                   0
        26           0           0                 0                   0
        27           0           0                 0                   0
        28           0           0                 0                   0
        29           0           0                 0                   0
        30           0           0                 0                   0
```

```
 - Directional Escape Probability for upper hemisphere (Number of directions=10)
Scattering order:   1
Zentih_Angle Azimuth_Angle Escape_photons
        18.43        90.00          17889
        18.43       270.00          13441
        63.43        22.50          10088
        63.43        67.50          12618
        63.43       112.50          12734
        63.43       157.50           9796
        63.43       202.50           7705
        63.43       247.50           6619
        63.43       292.50           6537
        63.43       337.50           7517

Scattering order:   2
Zentih_Angle Azimuth_Angle Escape_photons
        18.43        90.00          12194
        18.43       270.00          11765
        63.43        22.50           8266
        63.43        67.50           8182
        63.43       112.50           8350
        63.43       157.50           8062
        63.43       202.50           8245
        63.43       247.50           8109
        63.43       292.50           7970
        63.43       337.50           8025
```

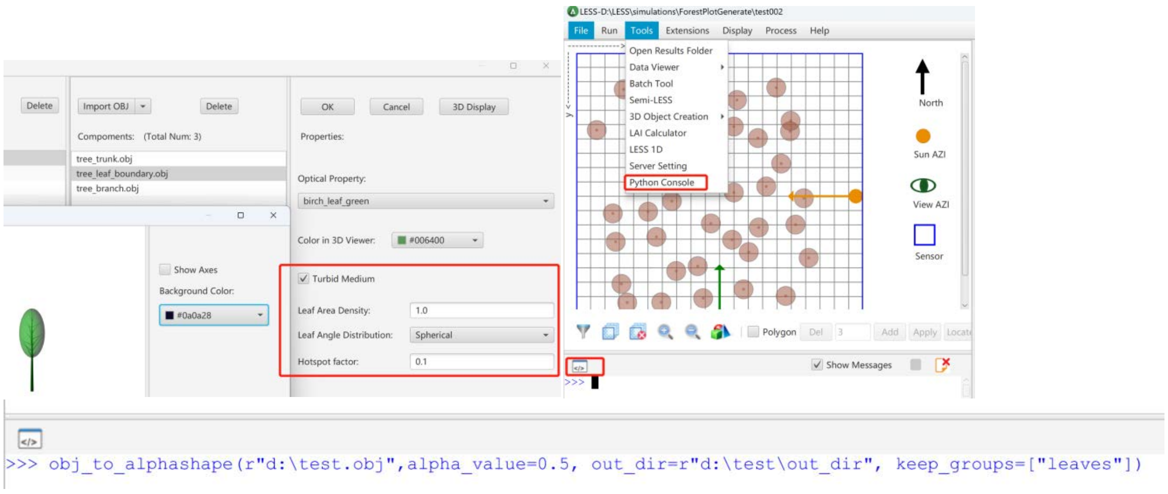
Figure 43. Re-collision probability simulation

5.1.13 Cloudy medium expressed leaf

In LESS, apart from using triangular surfaces to describe each leaf in detail, a cluster of leaves can also be statistically described using turbidity media. The advantage of this method is that it does not need detailed information of each leaf, but only needs to set its leaf area density, leaf inclination distribution and hot spot factor. The premise of using this expression is to create the outer envelope of the tree crown, which can be simple geometry or regular shapes such as volume elements. In LESS, the commonly used expression is alphashape, which can create a tight outer envelope based on the three-dimensional point cloud or surface element model to simulate the appearance of the tree crown, and the parameters of the turbidity medium can be set inside. The outer envelope of the tree Crown can be created using LESS **[Tools] [Simple Crown Generator]** or from an existing 3D OBJ model (sheet). LESS provides a tool to create an envelope from the existing OBJ model. In order to facilitate batch processing, this tool is provided in the form of Python functions. You can directly enter the Python terminal from the icon of the LESS main interface (the right picture in the following figure) or **[Tools] [Python Console]**, where you can directly run the code. The function for building the outer envelope is:

```
obj_to_alphashape(obj_path, alpha_value=0.5, out_dir=None,
keep_groups=None)
```

obj_path indicates the input obj path. alpha_value represents the radius parameter when constructing alphashape. The larger the value, the closer the outer envelope constructed is to the convex envelope, and the smaller the value is to the concave envelope. However, if the value is too small, it may fail, and the default value is 0.5); out_dir indicates the directory of the output file. If this parameter is not set, the output file will be saved to the same directory as the input obj file by default. keep_groups indicates that only some groups are calculated, that is, the original input obj file may have multiple groups, such as both branches and leaves. Usually, we only need to build the outer envelope for leaves, so only leaves can be included here. If this parameter is not set, all components are included by default. An example of a typical call is shown below. At the same time as the output envelope, a txt file with the same name will be output, which records the surface area of all components, as well as the calculated leaf area density, which can be used as the input of LESS.(**Figure 44**)



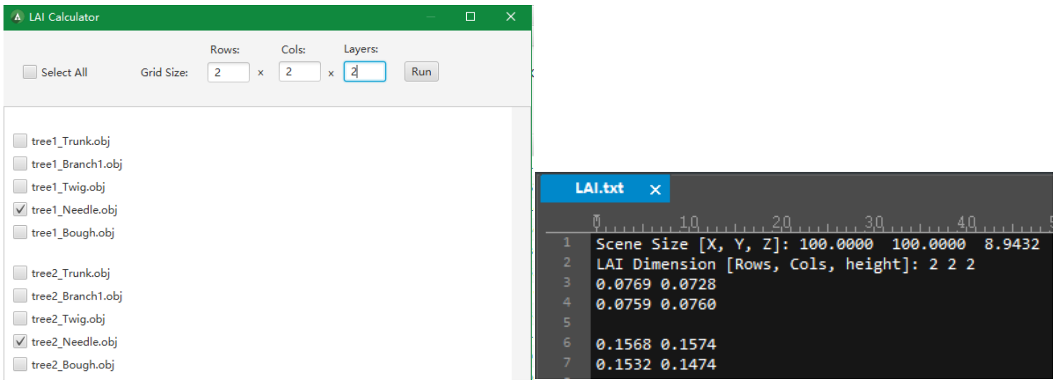
```
>>> obj_to_alphashape(r"d:\test.obj",alpha_value=0.5, out_dir=r"d:\test\out_dir", keep_groups=["leaves"])
```

Figure 44. Creation and setting of turbid media crown

5.2 Utility

5.2.1 LAI calculation tool

Click the menu **[Tools]-> [LAI Calculator]** to open the LAI calculation interface. Rows indicates the number of grids in the X direction, Cols indicates the number of grids in the Y direction, and Layers indicates the number of grids in the Z direction. For example, the value of **[Grid Size]** is 1x1x1, indicating that an LAI value is calculated for the entire scene. If the value of **[Grid Size]** is 2x2x2, it means that the whole scene will be divided into 2 regions in the X direction, 2 regions in the Y direction and 2 regions in the Z direction, and the final result will be 8 LAI values. It should be noted that **[LAI Calculator]** could calculate not only LAI, and the component names to be calculated should be checked below. If only leaf components were selected, the result would be LAI. If all components are checked, the result is PAI. As shown in the figure, the scene is divided into 8 blocks and LAI values are calculated respectively. After parameters are set, click **[Run]**. Open the result file LAI.txt. In the first line, "Scene Size" indicates the scene size. The second line "LAI Dimension" is the number of grids set for dividing the scene. LAI values start from the third row. The arrangement of LAI values corresponds to the spatial position of the block. The first four values are the corresponding values when the height is 1.(**Figure 45**)



Grid Size: Rows: 2 × Cols: 2 × Layers: 2

- ☐ tree1_Trunk.obj
- ☐ tree1_Branch1.obj
- ☐ tree1_Twig.obj
- ☑ tree1_Needle.obj
- ☐ tree1_Bough.obj
- ☐ tree2_Trunk.obj
- ☐ tree2_Branch1.obj
- ☐ tree2_Twig.obj
- ☑ tree2_Needle.obj
- ☐ tree2_Bough.obj

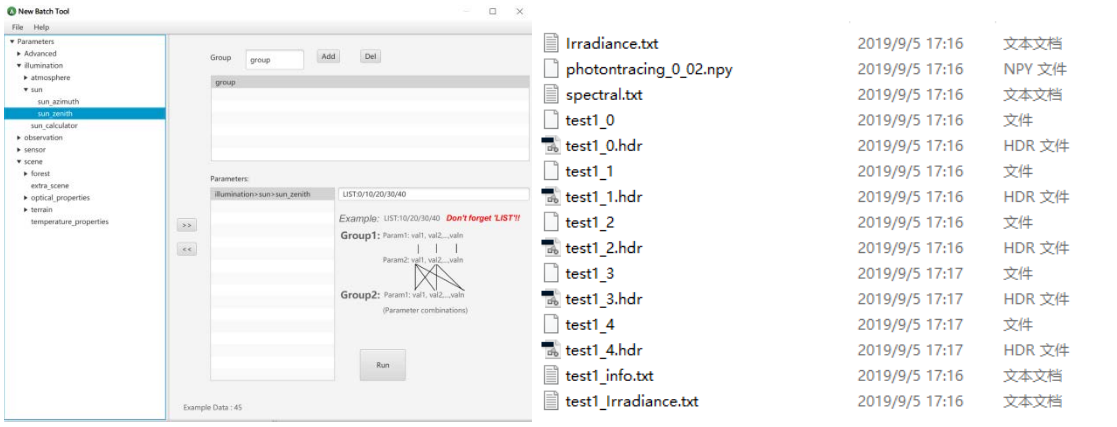
```
Scene Size [X, Y, Z]: 100.0000  100.0000  8.9432
LAI Dimension [Rows, Cols, height]: 2 2 2
0.0769 0.0728
0.0759 0.0760

0.1568 0.1574
0.1532 0.1474
```

Figure 45. LAI calculation tool

5.2.2 Batch processing tool

When the same scene needs to be simulated and only a few parameters are changed, for example, to study the change of the reflectance of the earth and the sky when the sun zenith Angle is increasing in the same scene, using batch processing tools will greatly reduce the amount of work required for simulation. Click the menu **[Tools] -> [Batch Tool]** to open the batch processing interface. Enter any name in **[Group]**, select the newly created **[Group],** find the Parameter to be changed in **[Parameter]** on the left and select it, such as "sun_zenith", click **[>>]**, and add it to **[Paramter]** on the right. After the changed parameter is added, enter a series of values that the parameter wants to set. For example, after sun_zenith is added, enter LIST:0/10/20/30/40. (Note that LIST is mandatory, and enter commas when the input method is English.) After parameter setting, click **[Run]** to save the file in the pop-up window, and then the program will run automatically. Open the results folder and the BRF files with solar zenith angles of 0°, 10°, 20°, 30°, 40° are saved in test0 to test4 respectively.(**Figure 46**)



Group

group

group

Parameters:

illumination>sun>sun_zenith

LIST:0/10/20/30/40

Example: LIST:10/20/30/40 Don't forget 'LIST'!!

Group1: Param1: val1, val2,...,valn

Param2: val1, val2,...,valn

Group2: Param1: val1, val2,...,valn

(Parameter combinations)

Example Data : 45

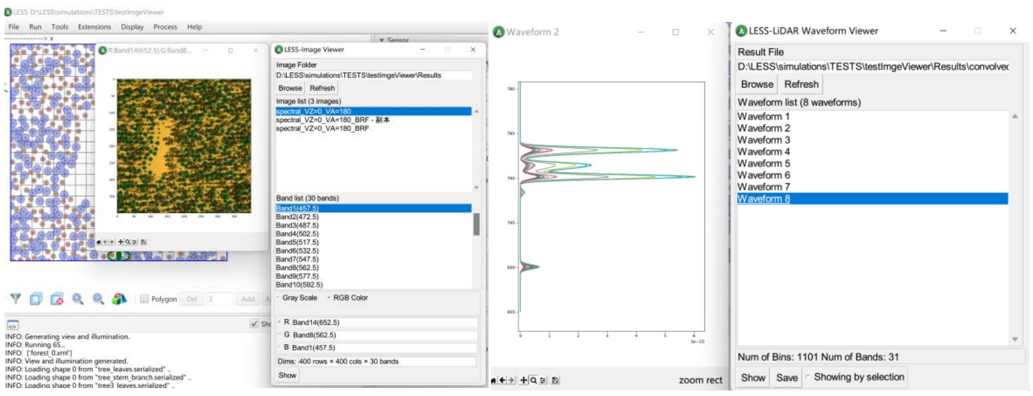
Irradiance.txt	2019/9/5 17:16	文本文档
photontracing_0_02.npy	2019/9/5 17:16	NPY 文件
spectral.txt	2019/9/5 17:16	文本文档
test1_0	2019/9/5 17:16	文件
test1_0.hdr	2019/9/5 17:16	HDR 文件
test1_1	2019/9/5 17:16	文件
test1_1.hdr	2019/9/5 17:16	HDR 文件
test1_2	2019/9/5 17:16	文件
test1_2.hdr	2019/9/5 17:16	HDR 文件
test1_3	2019/9/5 17:17	文件
test1_3.hdr	2019/9/5 17:17	HDR 文件
test1_4	2019/9/5 17:17	文件
test1_4.hdr	2019/9/5 17:17	HDR 文件
test1_info.txt	2019/9/5 17:16	文本文档
test1_Irradiance.txt	2019/9/5 17:16	文本文档

Figure 46. Visual batch processing tool

5.2.3 Image and LiDAR waveform viewer

LESS The default image format of simulated production is ENVI format (that is, a pair of files, one of which is an hdr header file), which is saved in the Results directory by default. In order to view simulated images easily and quickly, LESS provides a multi-band Image Viewer at the location of **[Tools] [Data Viewer] [Image Viewer]**, which will open all ENVI files in the Results directory by default. You can select one of them. All the bands of the file are listed below, and on the basis of the bands, you can select grayscale or RGB composition to display the image .

In order to view LESS simulated Waveform Data more conveniently, LESS also provided a waveform Viewer at the location of Tools **[Data Viewer][LiDAR Waveform Viewer]**. The waveform viewer selected 0.txt from the convolved_waveform directory in the Results directory by default, or it could switch waveform files by itself. After selecting one, all waveforms in the file would be listed. When selecting one, you could click **[Show]** to display the waveform. At the same time, if "Showing by selection" is selected, the waveform will be displayed immediately after a waveform is selected, without clicking "Show". This function can quickly browse the waveform.(**Figure 47**)



LESS-D:\LESS\simulations\TESTS\testImgeViewer
File Run Tools Extensions Display Process Help
R:Band14(652.5),G:Band8...
Polygon Del Add
Waveform 2
zoom rect

LESS-Image Viewer
Image Folder
D:\LESS\simulations\TESTS\testImgeViewer\Results
Browse Refresh
Image list (3 images)
spectral_VZ=0_VA=180
spectral_VZ=0_VA=180_BRF - 副本
spectral_VZ=0_VA=180_BRF
Band list (30 bands)
Band1(457.5)
Band2(472.5)
Band3(487.5)
Band4(502.5)
Band5(517.5)
Band6(532.5)
Band7(547.5)
Band8(562.5)
Band9(577.5)
Band10(592.5)
Gray Scale · RGB Color
R Band14(652.5)
G Band8(562.5)
B Band1(457.5)
Dims: 400 rows × 400 cols × 30 bands
Show

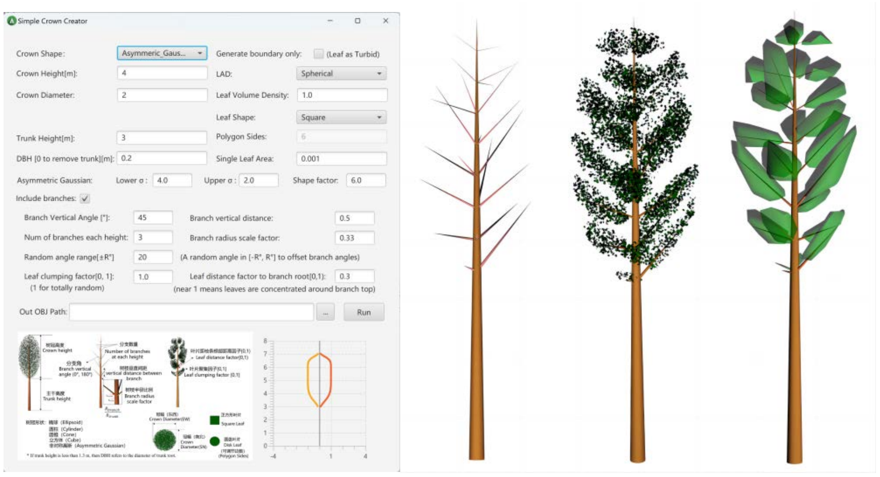
```
INFO: Generating view and illumination.
INFO: Running 6S...
INFO: ['forest_0.xml']
INFO: View and illumination generated.
INFO: Loading shape 0 from "tree_leaves.serialized" ..
INFO: Loading shape 0 from "tree_stem_branch.serialized" ..
INFO: Loading shape 0 from "tree3_leaves.serialized" ..
```

LESS-LiDAR Waveform Viewer
Result File
D:\LESS\simulations\TESTS\testImgeViewer\Results\convolvec
Browse Refresh
Waveform list (8 waveforms)
Waveform 1
Waveform 2
Waveform 3
Waveform 4
Waveform 5
Waveform 6
Waveform 7
Waveform 8
Num of Bins: 1101 Num of Bands: 31
Show Save Showing by selection

Figure47. Image viewer and waveform viewer

5.2.4 Simple crown generation tool

The graphical interface of LESS provides a tool for creating a Simple Crown in OBJ format, **[Tools] [3D Object Creation] [Simple Crown Creator]** (as shown below). In addition to the four simple tree crowns (cube, ellipsoid, cone and cylinder), LESS provides a tree crown described by asymmetric Gaussian distribution function, which uses three parameters to control its shape. Lower sigma is used to control the shape of the lower part of the tree crown, the smaller the value, the flatter the lower part of the tree crown. Upper sigma controls the top Shape of the tree crown, and the larger the value, the sharper the crown; Shape factor controls the overall shape of the tree crown, and the larger the value, the closer the crown is to the cylinder. If Generate boundary only is checked, only the outer outline of the tree crown will be generated. This tree crown model is mainly used to express the leaf in cloudy media. After LESS is imported, the leaf area density and leaf inclination distribution can be set. If you check the upper **[Include branches]**, in addition to setting the parameters of branches, you can also use branches to control Leaf clumping. Leaf clumping factor determines the degree of leaf clumping. When it is 1, the leaves are distributed randomly and evenly, and when it is close to 0, the leaves are clustered more in branches. Leaf distance factor represents the distance between the leaf and the root of the branch. The closer the value is to 1, the more the leaf gathers at the tip of the branch.(**Figure 48**)



Simple Crown Creator
Run
Crown height
Branch vertical angle (0°, 180°)
Trunk height
Number of branches at each height
vertical distance between branch
Branch radius scale factor
Leaf distance factor[0,1]
Leaf clumping factor [0,1]
Crown Diameter(EW)
Square Leaf

Crown Shape: Asymmeric_Gaus...
Generate boundary only: ☐ (Leaf as Turbid)
Crown Height[m]: 4
LAD: Spherical
Crown Diameter: 2
Leaf Volume Density: 1.0
Leaf Shape: Square
Trunk Height[m]: 3
Polygon Sides: 6
DBH [0 to remove trunk][m]: 0.2
Single Leaf Area: 0.001
Asymmetric Gaussian: Lower σ : 4.0 Upper σ : 2.0 Shape factor: 6.0
Include branches: ✓
Branch Vertical Angle [°]: 45
Branch vertical distance: 0.5
Num of branches each height: 3
Branch radius scale factor: 0.33
Random angle range[±R°] 20 (A random angle in [-R°, R°] to offset branch angles)
Leaf clumping factor[0, 1]: 1.0 (1 for totally random)
Leaf distance factor to branch root[0,1]: 0.3 (near 1 means leaves are concentrated around branch top)
Out OBJ Path:

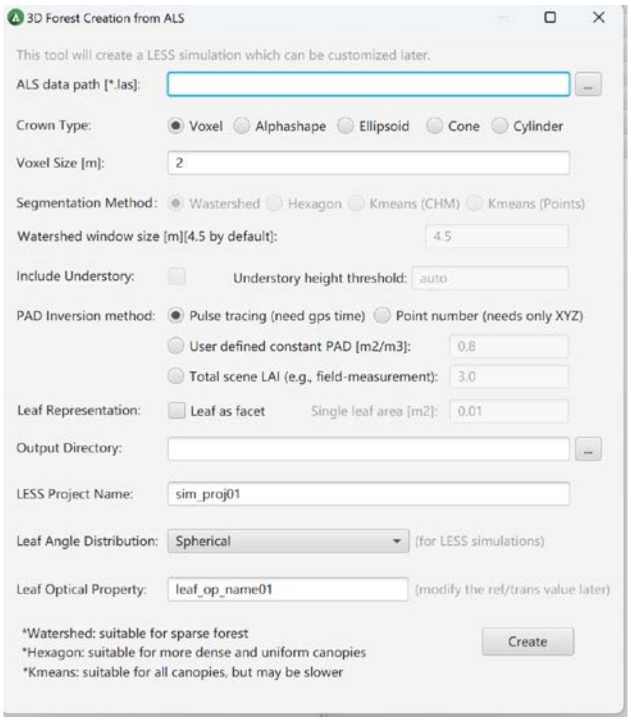
Figure48. Simple crown generation tool

5.2.5 Reconstruction of 3D scene from airborne point cloud

LESS's graphical user interface provides tools for reconstructing 3D scenes from airborne point clouds, including "Tools," "3D Object Creation," and "3D Forest from LiDAR (ALS)," as shown in the figure below. LESS provides five methods for representing tree crowns in turbid media: voxel, alpha shape, ellipsoid, cylinder, and cone. Voxel divides space into a uniform grid and calculates its radiative scattering for each unit. Alpha shape is a lightweight tree crown representation that balances accuracy in structure reconstruction and computational efficiency. Ellipsoid, cylinder, and cone are more efficient computation methods but have limited precision due to their overly simplified models.

This tool offers four methods for creating tree crowns (single-tree segmentation) in alpha shape, ellipsoid, cylinder, and cone: watershed algorithm, hexagonal algorithm, and clustering algorithms based on canopy height models (CHMs) or points. The watershed algorithm is suitable for sparse tree crowns but performs poorly in continuously dense scenes, resulting in some trees being too large or too small. LESS's default watershed window size is 4.5m, but users can set it themselves. The hexagonal algorithm is suitable for dense and uniform tree crowns, partitioning space into hexagons as the basic unit. The Kmeans clustering algorithm is suitable for all types of tree crowns and offers two ways of clustering. One is based on 2D CHM image clustering, which is efficient, and the other is direct 3D point cloud clustering, which has better results but may be less efficient. Additionally, users can set a height threshold to define the range of understory crowns. If the "Include Understory" box is checked, this tool will also model the understory. The tool calculates plant area density (PAD) for each reconstructed envelope using one of four methods: Pulse tracking, which requires the GPS time of point clouds to recover pulses and calculate transmittance to ultimately calculate PAD; a point-based method using only 3D coordinates of ground and non-ground points to calculate transmittance; a constant value for PAD or PAI set for the entire scene, with PAD calculated automatically based on tree crown volume.

The distribution of leaf inclination angle has six types: spherical, uniform, planar, vertical, inclined, and extreme, with the default in LESS being spherical. The default LESS project file name is "simproj01," and the default leaf spectral attribute name is "leafopname01." After setting the parameters, click "Create." After completion, a LESS project named "simproj01" will be generated under the "Output Directory," which can be opened using either LESS's GUI or SDK. By default, all tree crowns' optical attributes are preset to "leafopname01," so after opening the project, users need to define the specific reflectance/transmittance values of "leafopname01" in the "Optical Databse" before simulation.(**Figure 49**)



3D Forest Creation from ALS

This tool will create a LESS simulation which can be customized later.

ALS data path [*.las]:	...
Crown Type:	[✓ Voxel] Alphashape Ellipsoid Cone Cylinder
Voxel Size [m]:	2
Segmentation Method:	[✓ Wastershed] Hexagon Kmeans (CHM) Kmeans (Points)
Watershed window size [m][4.5 by default]:	4.5
Include Understory:	Understory height threshold: auto
PAD Inversion method:	[✓ Pulse tracing (need gps time)] Point number (needs only XYZ)
	User defined constant PAD [m2/m3]: 0.8
	Total scene LAI (e.g., field-measurement): 3.0
Leaf Representation:	Leaf as facet Single leaf area [m2]: 0.01
Output Directory:	...
LESS Project Name:	sim_proj01
Leaf Angle Distribution:	Spherical (for LESS simulations)
Leaf Optical Property:	leaf_op_name01 (modify the ref/trans value later)

*Watershed: suitable for sparse forest
*Hexagon: suitable for more dense and uniform canopies
*Kmeans: suitable for all canopies, but may be slower

Create

Figure 49. Simple crown generation tool

5.2.6 Python code automatic generation tool

At present, LESS has two ways to run, one is by manipulating the GUI, and the other is through the Python SDK script to perform simulation. The advantage of the former is that the operation is intuitive and simple, and the advantage of the latter is that it is easy to control the simulation process, such as adding elements, modifying the spectrum, and then conducting batch simulation. However, simulation through SDK first needs to be familiar with the function interface of SDK, which has a certain learning cost. In order to reduce the difficulty of using code, LESS provides the ability to automatically generate corresponding Python scripts based on GUI parameters. The method of use is: First, open the GUI interface, then set the corresponding simulation parameters, click **[Tools] [Generate Python Code]**, and select the save path. Then a py script file will be generated, the content of which is the script corresponding to the simulation project created by automatically generating GUI using Python SDK. If the script is modified to some extent, the parameters can be easily modified or batch simulated.

5.3 LESS file format

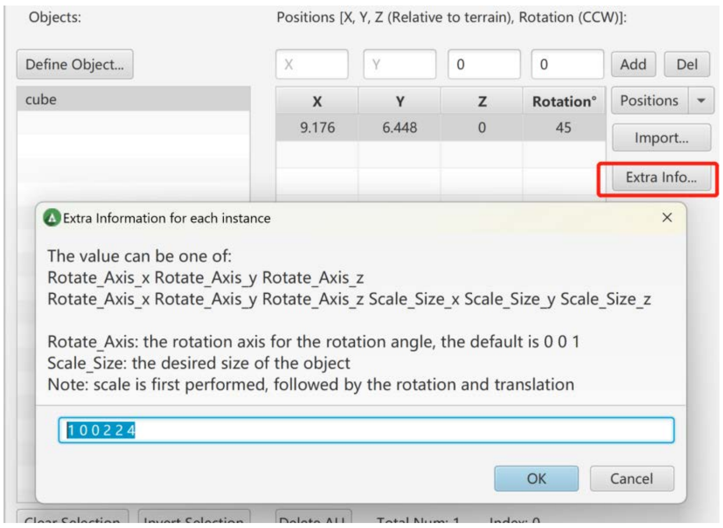
5.3.1 instance.txt

In the Parameters directory of the simulation project, the instance.txt file stores the position information of each object. The file has three different formats, representing different meanings:

- 5-column mode: object_name X Y Z Rotate, where the 5 columns respectively represent the object's name, its XYZ position, and a rotation angle. The default rotation angle is counterclockwise in the horizontal plane, in degrees.

- 8-column mode: objectname X Y Z Rotate AxisX AxisY AxisZ, with three additional columns representing the rotation center axis when rotating. The right-hand rotation rule is adopted, and the rotation center is the origin of the obj file. The rotation is performed first before moving to the XYZ position.
- 11-column mode: objectname X Y Z Rotate AxisX AxisY AxisZ Scale_X ScaleY ScaleZ, which includes three additional columns defining the object's size (not scaling factor) in the order of scaling, rotation, and translation.

To simplify the GUI interface, LESS only provides direct editing for the 5-column mode. Other modes can be edited directly in the instance.txt file and then the project needs to be reopened; alternatively, the instance.txt file can be imported from the GUI interface. All coordinate definitions in the file are the same as in the GUI, that is, X moves to the right, Y moves downward, and Z is the vertical direction. In addition, for each selected position information in LESS's GUI, clicking the "Extra Info⋯" button brings up a dialog box where additional information beyond the 5 columns can be edited.(**Figure 50**)


Objects:
Positions [X, Y, Z (Relative to terrain), Rotation (CCW)]:
Define Object...
cube
X
Y
0
0
Add
Del
Positions
Import...
Extra Info...
Extra Information for each instance
OK
Cancel

X	Y	Z	Rotation°
9.176	6.448	0	45

The value can be one of:
Rotate_Axis_x Rotate_Axis_y Rotate_Axis_z
Rotate_Axis_x Rotate_Axis_y Rotate_Axis_z Scale_Size_x Scale_Size_y Scale_Size_z

Rotate_Axis: the rotation axis for the rotation angle, the default is 0 0 1
Scale_Size: the desired size of the object
Note: scale is first performed, followed by the rotation and translation

1 0 0 2 2 4

Figure 50. Simple crown generation tool

5.4 SDK for Python

5.4.1 Implement LESS simulation using SDK for Python

In addition to using the GUI program for simulation, LESS also provides a Python SDK that enables users to create new simulation projects, develop 3D scenes, modify simulation parameters, and conduct data simulations using Python scripts. The SDK is located in the pyLessSDK folder of the LESS installation directory (e.g., D:\LESS\app\Python_script\pyLessSDK). The main classes and their relations in the SDK are shown in **Figure 51**.

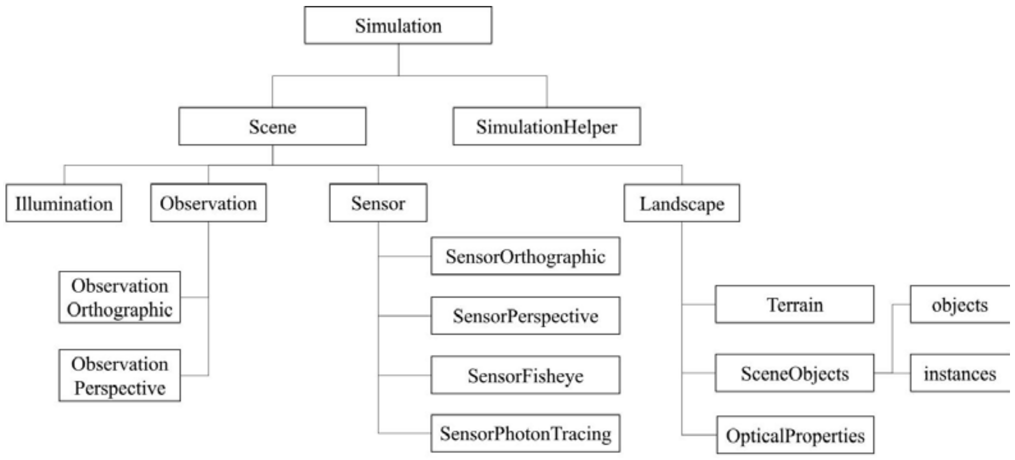


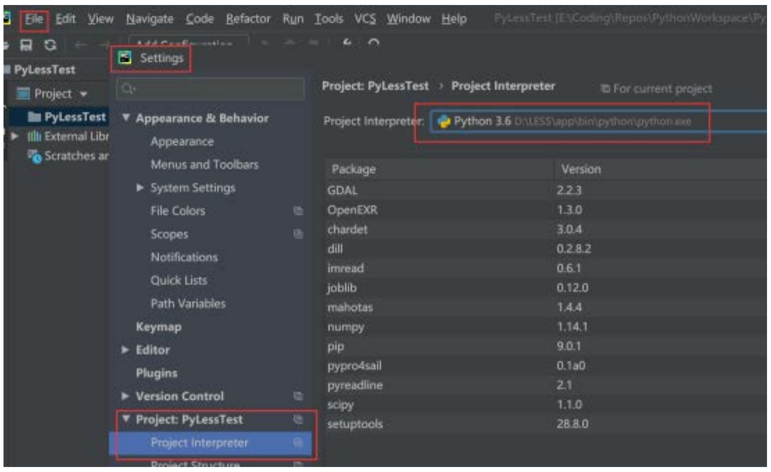
Figure 51. LESS Main classes in the SDK and their relationships

To use the LESS SDK, an appropriate Python interpreter is required. The LESS installation package includes a full Python 3.6 interpreter (other Python interpreters can also be used) located at "LESS installation directory\app\bin\python\python.exe". It is recommended to use the PyCharm Integrated Development Environment (IDE) for code completion and program debugging.

After creating a new project (e.g., PyLessTest), select the appropriate Python version in **[File] [Settings] [Project Interpreter]**, as shown in Figure 51. Then create a new .py file to start writing the program. To enable the program to find the LESS SDK, there are two ways, first by adding the path through the sys module:

```python
import sys
sys.path.append(r"D:\LESS\app\Python_script\pyLessSDK")
from SimulationHelper import SimulationHelper
```

Although this method allows the program to run, it is not conducive to program development, as there is no code hinting. In PyCharm, there is another way, as shown in **Figure 52**. Click the dropdown arrow at **[File] [Settings] [Project Interpreter]**, select **[Show All]**, and then choose the Python version being used. Add the location of the LESS SDK to the path, and the module can be imported directly.



Project: PyLessTest › Project Interpreter

Project Interpreter: Python 3.6 D:\LESS\app\bin\python\python.exe

Package	Version
GDAL	2.2.3
OpenEXR	1.3.0
chardet	3.0.4
dill	0.2.8.2
imread	0.6.1
joblib	0.12.0
mahotas	1.4.4
numpy	1.14.1
pip	9.0.1
pypro4sail	0.1a0
pyreadline	2.1
scipy	1.1.0
setuptools	28.8.0

Figure 52. PyCharm configuration

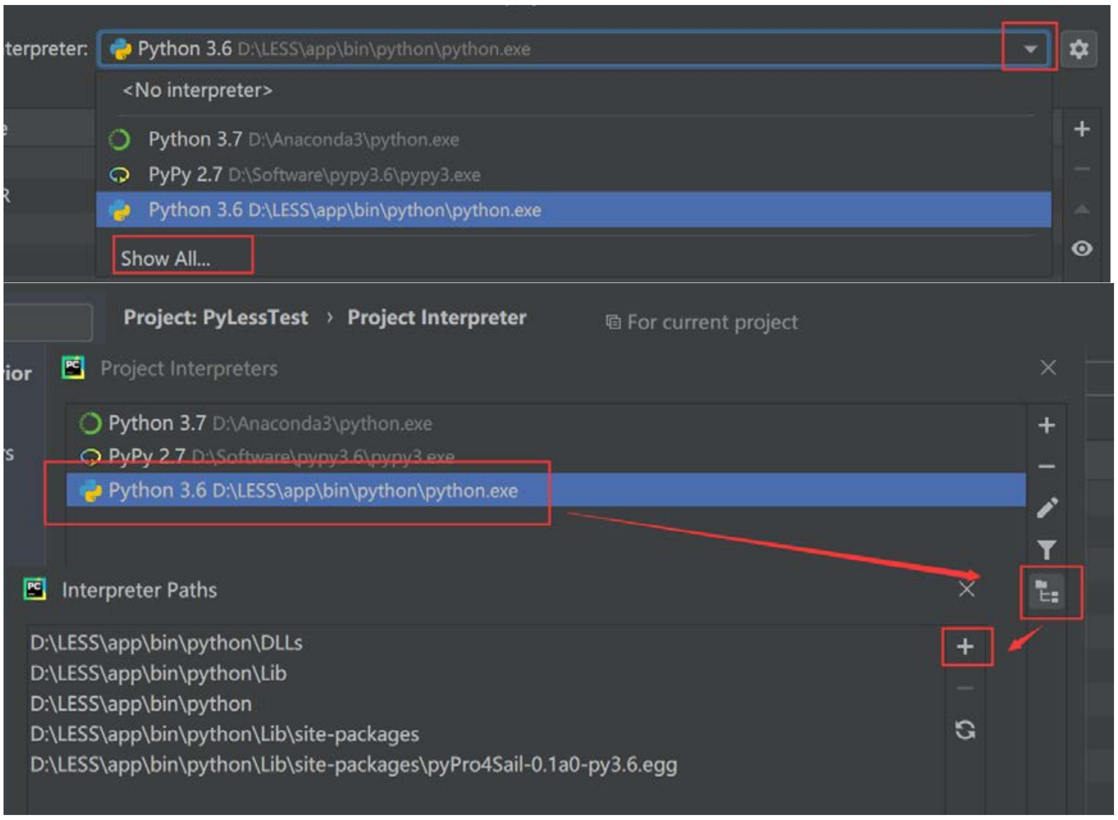

terpreter:
Python 3.6 D:\LESS\app\bin\python\python.exe
<No interpreter>
Python 3.7 D:\Anaconda3\python.exe
PyPy 2.7 D:\Software\pypy3.6\pypy3.exe
Python 3.6 D:\LESS\app\bin\python\python.exe
Show All...
Project: PyLessTest › Project Interpreter
For current project
Project Interpreters
Python 3.7 D:\Anaconda3\python.exe
PyPy 2.7 D:\Software\pypy3.6\pypy3.exe
Python 3.6 D:\LESS\app\bin\python\python.exe
Interpreter Paths
D:\LESS\app\bin\python\DLLs
D:\LESS\app\bin\python\Lib
D:\LESS\app\bin\python
D:\LESS\app\bin\python\Lib\site-packages
D:\LESS\app\bin\python\Lib\site-packages\pyPro4Sail-0.1a0-py3.6.egg

Figure 52. Configure the LESS SDK in PyCharm

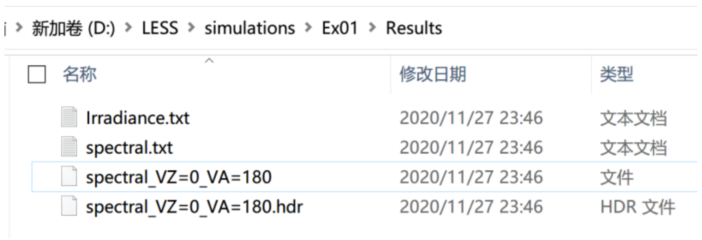
- **Ex01 Basic simulation**

The following code demonstrates an example of creating a default simulation project using Python and conducting simulation directly. Firstly, import the SimulationHelper and Simulation classes; the former is an auxiliary class for a simulation project, and the latter represents the entire simulation project. Simulation includes a SimulationHelper and a Scene member variable. The SimulatinHelper mainly provides functions such as creating a new simulation project and obtaining some system parameters (such as the path of Python). For instance, the createnewsim function creates a new simulation project based on the input directory path and initializes the project with a standard template, setting default parameters, etc. If the simulation project already exists, it will be skipped. The Simulation instance is initialized from a simulation project path and a SimulatinHelper instance. After obtaining the Simulation instance (i.e., sim), readsimproject function reads the simulation project's content into the sim object. Then, the savesimproject() function can be called to save the project (no parameters are modified here, so it can also be omitted), and finally, the start() function is called to conduct simulation.

```python
# coding: utf-8
from SimulationHelper import SimulationHelper
from Simulation import Simulation
from PostProcessing import PostProcessing
import os
```

```
sim_dir = r"D:\LESS\simulations\Ex01"
sim_helper = SimulationHelper(r"D:\LESS")  # Create a
SimulationHelper instance with the root directory of LESS
installation as the parameter
sim_helper.create_new_sim(sim_dir)  # Create a new simulation
project
sim = Simulation(sim_dir, sim_helper)  # Initialize the Simulation
object
sim.read_sim_project()  # Read the contents of the simulation
project
sim.save_sim_project()  # Save the project
sim.start()  # Start the simulation
```

The above code is the simplest simulation process, which can be implemented without opening the GUI program. The final simulation result is an image (spectralVZ=0VA=180), which is saved in D:\LESS\simulations\Ex01\Results\ directory in ENVI format, with Radiance as the pixel value type.(**Figure 53**)



› 新加卷 (D:) › LESS › simulations › Ex01 › Results

名称	修改日期	类型
Irradiance.txt	2020/11/27 23:46	文本文档
spectral.txt	2020/11/27 23:46	文本文档
spectral_VZ=0_VA=180	2020/11/27 23:46	文件
spectral_VZ=0_VA=180.hdr	2020/11/27 23:46	HDR 文件

Figure 53. The radiance image is obtained by simulation

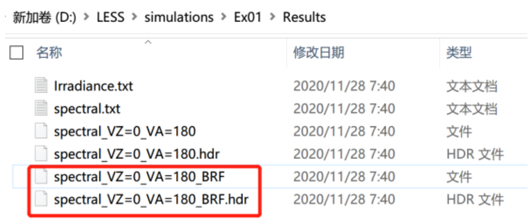
To convert radiance to reflectance, the radiance2brf function in PostProcessing can be used. This function has three parameters, the first being the path of the simulation project, used to obtain information such as irradiance. The second parameter is the input file path, which is the radiance file obtained from the previous step of the simulation. The third parameter is the output file path - the BRF file. After the simulation is completed, the resulting BRF file can be obtained (the highlighted and bolded code in the code below is the newly added code).

```
# coding: utf-8
from SimulationHelper import SimulationHelper
from Simulation import Simulation
from PostProcessing import PostProcessing
import os

sim_dir = r"D:\LESS\simulations\Ex01"
sim_helper = SimulationHelper(r"D:\LESS")  # Create a
SimulationHelper instance with the root directory of LESS
installation as the parameter
sim_helper.create_new_sim(sim_dir)  # Create a new simulation
project
sim = Simulation(sim_dir, sim_helper)  # Initialize the Simulation
object
```

```
sim.read_sim_project()  # Read the contents of the simulation
project
# Output file from the simulation
sim.save_sim_project()  # Save the project
sim.start()  # Start the simulation
PostProcessing.radiance2brf(sim.get_sim_dir(), sim.get_dist_file(),
sim.get_dist_file()+"_BRF")
```

After the simulation is complete, the resulting BRF file will be displayed as shown in **Figure 54**.



新加卷 (D:) > LESS > simulations > Ex01 > Results

名称	修改日期	类型
Irradiance.txt	2020/11/28 7:40	文本文档
spectral.txt	2020/11/28 7:40	文本文档
spectral_VZ=0_VA=180	2020/11/28 7:40	文件
spectral_VZ=0_VA=180.hdr	2020/11/28 7:40	HDR 文件
spectral_VZ=0_VA=180_BRF	2020/11/28 7:40	文件
spectral_VZ=0_VA=180_BRF.hdr	2020/11/28 7:40	HDR 文件

Figure 54. The reflectivity image is obtained by simulation

The code above uses the default output file name spectral_VZ=0_VA=180 (obtained through the function sim.get_dist_file). However, in reality, for convenience, we can set our own output file name during the simulation by modifying the parameters between read_sim_project and save_sim_project, as shown in the following code:

```
sim.read_sim_project()  # Read the contents of the simulation project
sim.set_dist_file(os.path.join(sim.get_sim_dir(), "Results",
"output_radiance_file"))
# Output file from the simulation
sim.save_sim_project()  # Save the project
```

- **Ex02 Add scene elements**

In Ex01, although we have implemented the entire simulation process, the default scene only contains a simple flat terrain of 100m * 100m. In this example, we will add some trees to the scene and set their spectra.

A. Importing Scene OBJ Files

In LESS, all scene elements (trees, houses, etc.) are input as OBJ format files consisting of triangular faces, and tree models can be generated using third-party software such as OnyxTree.(**Figure 55**)

All scene elements are managed within the Scene class. After creating a Simulation, you can obtain an instance of Scene through sim.get_scene() and then access or modify the elements in the scene. The example code is as follows:

```
# coding: utf-8
from SimulationHelper import SimulationHelper
from Simulation import Simulation
from PostProcessing import PostProcessing
import os
```

```
from SceneObjects import SceneObject
from Utility import OBJHelper
import random

sim_dir = r"D:\LESS\simulations\Ex02"
sim_helper = SimulationHelper(r"D:\LESS")  # Create a SimulationHelper,
the parameter is the root directory of the LESS installation.
sim_helper.create_new_sim(sim_dir)  # Create a new simulation project
sim = Simulation(sim_dir, sim_helper)  # Initialize a Simulation object
sim.read_sim_project()  # Read the content of the simulation project
sim.set_dist_file(os.path.join(sim.get_sim_dir(), "Results",
"output_radiance_file"))

#################Add scene elements start ####################
scene = sim.get_scene()  # Get the Scene
landscape = scene.get_landscape()  # Get the LandScapeobject
landscape.clear_landscape_elements()  # # Clear the scene elements that
already exist in the project (must be executed when the program is run
repeatedly)
obj_tree01 = SceneObject("Tree01")  #  Define a scene object named Tree01
# Add an obj file as a component of tree1. For obj files that contain multiple
groups, the paths for each group must be separated and imported separately, and
each group must be given a different optical property
comp_list =
OBJHelper.seperate_obj(r"D:\LESS\app\Database\3D_Objects\Trees\RAMI
\FREX_FROM_HET09_JBS_SUM.obj")
# FREX_FROM_HET09_JBS_SUM.obj contains two components, the first
one is leaves, and the second one is branch, so they are imported
separately and given different optical properties.
# birch_leaf_green and birch_branch are default built-in optical
properties. Custom optical properties will be explained in the next
example.
obj_tree01.add_component_from_file(comp_list[0],
"birch_leaf_green")
obj_tree01.add_component_from_file(comp_list[1], "birch_branch")
landscape.add_object(obj_tree01)


# When an obj file is defined, it also needs to be placed in different
positions, which is called an instance in LESS
# Randomly place in the scene, the default size of the scene is 100m * 100m.
for i in range(1000):
    x = random.random()*100
    y = random.random()*100
    landscape.place_object("Tree01", x=x, y=y)
############### Add scene elements end ##############
# Output file of the simulation
sim.save_sim_project()  #  Save the project
sim.start()  # Start the simulation
```

```
PostProcessing.radiance2brf(sim.get_sim_dir(), sim.get_dist_file(),
sim.get_dist_file()+"_BRF")
```

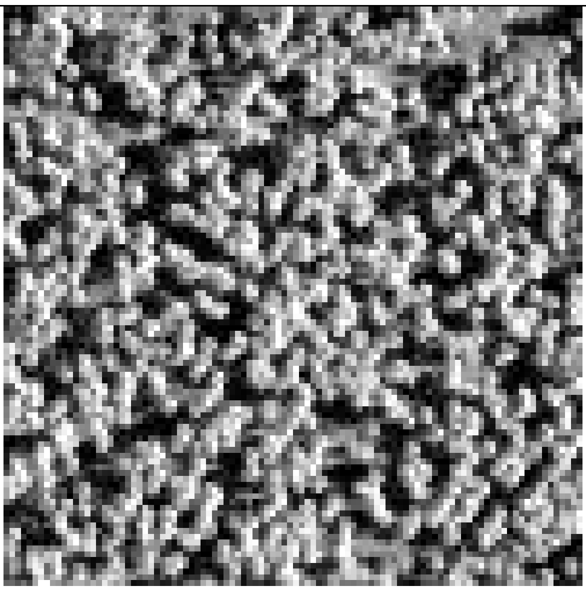


Figure 55. Add a tree model to the scene

B. Custom Optical Properties

In LESS, an optical property typically refers to a pair of reflectance and transmittance for both the front and back sides of an object (assuming the transmittance is the same for both sides). The class for defining optical properties is called OpticalItem. Based on the example above, the code that needs to be modified is:

```
scene = sim.get_scene()  # Get Scene
landscape = scene.get_landscape()  # Get Landscape object
landscape.clear_landscape_elements()  # # Clear existing landscape
elements in the project (must be executed when running the program repeatedly)
# Define an optical property (front reflectance; back reflectance;
transmittance)
landscape.clear_user_defined_optical_properties()  # Clear user-
defined optical properties
# LESS initially has two bands
op_item01 = OpticalItem("op_leaves", "0.05,0.4;0.05,0.4;0.05,0.4")
landscape.add_op_item(op_item01)  # Add to the scene library
op_item02 = OpticalItem("op_branch", "0.2,0.45;0,0;0,0") #LESS
initially has two bands
landscape.add_op_item(op_item02)  # Add to the scene library


obj_tree01 = SceneObject("Tree01")  # Define a scene object called Tree01
# Add an obj file as a component of tree1. For obj files containing multiple
groups, each group needs to be separated and imported separately with different
optical properties set for each group
# different optical properties set for each group
comp_list =
OBJHelper.seperate_obj(r"D:\LESS\app\Database\3D_Objects\Trees\RAMI
\FREX_FROM_HET09_JBS_SUM.obj")
```

```
# FREX_FROM_HET09_JBS_SUM.obj contains two components, the first
one is leaves and the second one is branch, so they are imported
separately
# birch_leaf_green and birch_branch are default pre-defined
internal optical properties, custom optical properties will be
explained in the next example
obj_tree01.add_component_from_file(comp_list[0], "op_leaves")
obj_tree01.add_component_from_file(comp_list[1], "op_branch")
landscape.add_object(obj_tree01)
```

C. Custom Sensor

In examples Ex02-A and Ex02-B, we used the default sensor, which is an orthographic projection with an image size of 100 pixels by 100 pixels. In this example, we will modify the sensor parameters using the SDK. The sample code is as follows:

```
from Observation import ObservationOrthographic
…
# After defining an obj file, it needs to be placed at different locations,
which is called an instance in LESS
# Randomly place the object in the scene, the default scene size is 100m*100m
for i in range(1000):
    x = random.random()*100
    y = random.random()*100
    landscape.place_object("Tree01", x=x, y=y)
################ Adding Landscape Elements end #####################

# Modify sensor parameters
sensor = scene.get_sensor()  # Get Sensor first
sensor.set_film_type("rgb")  # Set to rgb, which can directly output a png
image. If set to spectrum, it outputs a multi-band image in Envi format.
sensor.set_image_width(300)
sensor.set_image_height(300)
sensor.set_sample_per_pixel(16)

# Create a new Observation, then set it to the scene. It can also be read
directly and then modified.
obs = ObservationOrthographic()  # Default zenith angle is 0 and azimuth
angle is 180
scene.set_observation(obs)

# The output file for simulation
sim.save_sim_project()  # Save project
sim.start()  # Start simulation
```

Using the above code to simulate an image, as shown in **Figure 56**, it can be seen that the resolution of the simulated image has increased and the outline of the tree canopy is clearer.

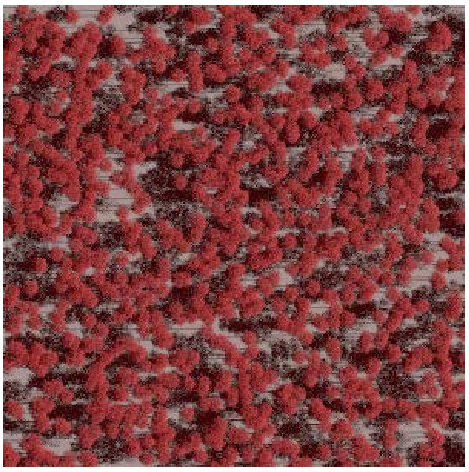


Figure 56. Modifying Sensor parameters

The previous modifications only adjusted the parameters of the default SensorOrthographic (e.g. image size). If other types of sensors need to be set (SensorPerspective, SensorFisheye, SensorPhotonTracing), they need to be created and added to the scene. An example code for SensorPerspective is shown below:

```
# Add a new sensor
sensor = SensorPerspective()
sensor.set_fov_x(40)  # X direction field of view (unit: degree)
sensor.set_fov_y(30)  # Y direction field of view (unit: degree)
sensor.set_film_type("rgb")
sensor.set_image_width(300)
sensor.set_image_height(300)
sensor.set_sample_per_pixel(16)
scene.set_sensor(sensor)

# Add a new observation, which corresponds to ObservationPerspective for
SensorPerspective
obs = ObservationPerspective()
obs.set_origin((50, 50, 150))  # Set the position of the camera
obs.set_target((50, 50, 0))  # Set the observation target of the camera.
The vector between the target and the origin determines the direction of the
camera observation.

scene.set_observation(obs)
```

SensorPhotonTracing simulates forward ray tracing and does not have an actual camera entity. Therefore, ObservationPhotonTracing corresponding to SensorPhotonTracing does not need to set specific parameters, and an empty instance needs to be created. Example code is shown below:

```
# Add a forward ray tracing sensor to simulate multi-angle reflectivity
sensor = SensorPhotonTracing()
sensor.enable_brf_product(True)
sensor.set_sun_ray_resolution(5)
sensor.set_virtual_directions("10:90;20:90")
scene.set_sensor(sensor)
```

```
obs = ObservationPhotonTracing()
scene.set_observation(obs)
```

- **Ex03 Lidar simulation**

Simulation of LiDAR is similar to that of image simulation, but there are differences in the parameters of the sensor. To enable LiDAR simulation, call scene.enable_lidar(True). Currently, there are two built-in LiDAR sensors, ALSLiDAR and TLSLiDAR. Each sensor has three types of parameters: Beam, Device and Platform. Beam includes Axial Division and Maximum Scattering Order, which are called using liarSensor.beam.axialDivision and lidarSensor.beam.maxOrder (lidarSensor being the created LiDAR object, for example, lidarSensor = ALSLiDAR()). Device is the hardware parameter of the instrument, which includes spot energy, etc., called using lidarSensor.device(). Platform is the platform parameter, called using lidarSensor.platform. The function liarSensor.set_lidar_sim_mode(LiDARSimMode.MultiRayPointCloud) sets the simulation mode, there are three types: Single Ray Point Cloud, Multi Ray Point Cloud, and Multi Ray Waveform (corresponding to GUI one by one). After creating the sensor, it needs to be set to the scene using scene.set_lidar_sensor(liarSensor), and then save the project and run the simulation.

```
# coding: utf-8
from SimulationHelper import SimulationHelper
from Simulation import Simulation
import os
from LiDAR import LiDAR, ALSLiDAR, LiDARSimMode

sim_dir = r"D:\LESS\simulations\Ex03_lidar"
sim_helper = SimulationHelper(r"D:\LESS")  # Create a SimulationHelper with the installation root directory of LESS
sim_helper.create_new_sim(sim_dir)  # New construction simulation project
sim = Simulation(sim_dir, sim_helper)  # Initialize the Simulation object
sim.read_sim_project()  # Read the contents of the simulation project
scene = sim.get_scene()  # Get Scene

scene.enable_lidar(True)
liarSensor = ALSLiDAR()
liarSensor.beam.axialDivision = 30
liarSensor.beam.maxOrder = 1
liarSensor.platform.altitude = 800
liarSensor.platform.startX = 5
liarSensor.platform.startY = 50
liarSensor.platform.endX = 95
liarSensor.platform.endY = 50
liarSensor.platform.swathWidth = 90
liarSensor.platform.rangeResolution = 0.5
liarSensor.platform.yawResolution = 0.5
liarSensor.platform.minRange = 770
liarSensor.platform.maxRange = 805

liarSensor.set_lidar_sim_mode(LiDARSimMode.MultiRayPointCloud)
```

```
scene.set_lidar_sensor(liarSensor)

sim.save_sim_project()  # Save the project
sim.start()  # Start simulation
```

5.4.2 Crown formation

The SDK for Python provides a set of tools for generating tree crowns of various shapes (with randomly distributed leaves inside the crown). The fundamental parameters of the tree crown are shown in the figure below.(**Figure 57**)

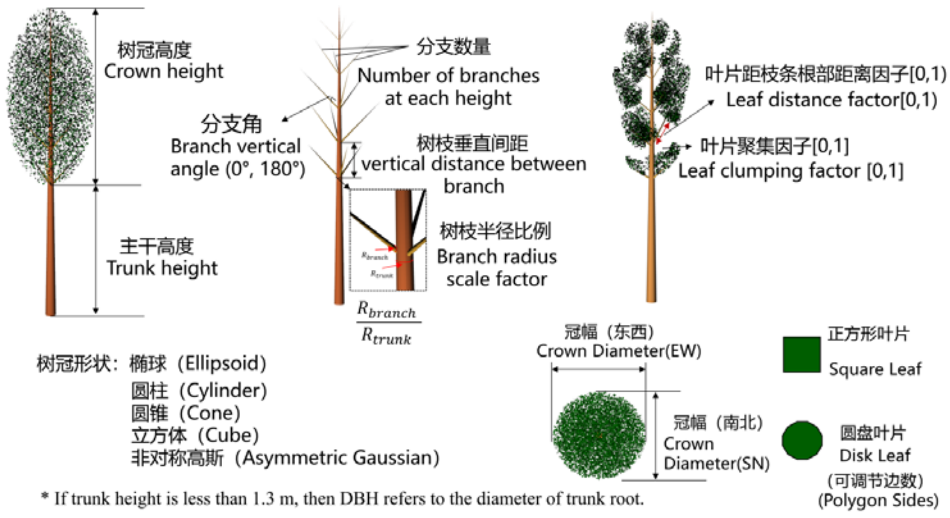

树冠高度
Crown height
主干高度
Trunk height
分支角
Branch vertical angle (0°, 180°)
分支数量
Number of branches at each height
树枝垂直间距
vertical distance between branch
树枝半径比例
Branch radius scale factor
$\frac{R_{branch}}{R_{trunk}}$
叶片距枝条根部距离因子[0,1)
Leaf distance factor[0,1)
叶片聚集因子[0,1]
Leaf clumping factor [0,1]
树冠形状：椭球（Ellipsoid）
圆柱（Cylinder）
圆锥（Cone）
立方体（Cube）
非对称高斯（Asymmetric Gaussian）
冠幅（东西）
Crown Diameter(EW)
冠幅（南北）
Crown Diameter(SN)
正方形叶片
Square Leaf
圆盘叶片
Disk Leaf
（可调节边数）
(Polygon Sides)
* If trunk height is less than 1.3 m, then DBH refers to the diameter of trunk root.

Figure57. Basic parameters of tree crown

Here is the code sample:

```
from ObjectGenerator import CrownGenerator, CrownShape, LeafShape, LAD
cg = CrownGenerator()
cg.lvd = 10000   # Leaf Area Index, LAI
cg.crown_diameter_EW = 2   # East-west crown
cg.crown_diameter_SN = 2   # North-south crown width
cg.crown_height = 4       # Crown height
cg.crown_shape = CrownShape.ELLIPSOID      # Crown shape
cg.leaf_angle_dist = LAD.SPHERICAL       # leaf angle distribution
cg.single_leaf_area = 0.001        # Individual blade area
cg.leaf_shape = LeafShape.SQUARE       # Blade shape. If DISK is set, the number of triangles on the blade can be set（leaf_number_triangles）
cg.has_trunk = True   # Include trunk or not
cg.trunk_height = 3   # If yes, set the trunk height
cg.dbh = 0.2   # Main DBH
cg.generate_crown(r"D:\crown.obj")   # Save the obj file
```

6. LESS extension modules

6.1 LESS1D

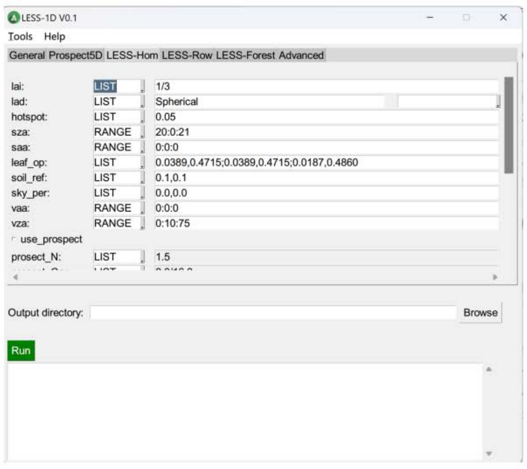
The canopy radiation transfer model is the foundation for vegetation quantitative remote sensing inversion. The current commonly used one-dimensional model is widely applied due to its high computational efficiency. Although three-dimensional models can obtain more accurate simulation results, the complexity of input parameters and less convenient operations limit their application. To fully utilize the accuracy feature of the three-dimensional model, this study developed the LESS1D module by compromising between one and three dimensions, which is a one-dimensional simplified version of the LESS three-dimensional radiation transfer model. LESS1D provides input parameters similar to a one-dimensional model (such as leaf area index, leaf inclination angle distribution, and plant density), but with the calculation accuracy of a three-dimensional model (no need for approximate processing of hotspots). The LESS1D currently provides three scene simulations: LESS-Hom, LESS-Row, and LESS-Forest, which represent horizontally uniform distribution scenes, ridge row structure scenes, and discrete forest canopy scenes, respectively. The radiation transfer calculation process of LESS1D is consistent with the LESS model, which performs radiation transfer process entirely by ray tracing, therefore, it has high computational accuracy. LESS1D provides a user-friendly graphical interface that supports batch input of parameters and batch simulation, making it an ideal tool for simulating a large number of parameter combinations.

To open the LESS main program, open the LESS1D main interface under **[Tools]**, click **[Browse]** to select the simulation result output path, and create a new simulation project. You can also load an existing simulation project through the main interface by selecting **[Tools]** and then **[Load]**. After setting all the parameters, click **[Tools]** and then **[Save]** to save the simulation parameters.

1.Basic parameter settings: In **[General]** set the spectral **[wavelengths]** for simulation, and set the number of cores according to the computer's performance in **[num_of_cores]**.

2.Leaf reflectance simulation: For convenience, LESS1D provides an additional PROSPECT model that can simulate leaf scale reflectance and transmittance by inputting leaf parameters. The data type of each parameter has two options, **[LIST]** and **[RANGE]**, with **[LIST]** separating different data with a "/," and **[RANGE]** setting the lower limit, step size, and upper limit. The final simulation result is the combination of each parameter, making it easy to perform batch processing. In **[Prospect5D]**, set the leaf parameters, click **[Browse]** to set the output directory, and click **[Run]** to obtain the leaf spectral reflectance and transmittance simulation.

3.Canopy spectral simulation: LESS1D provides three canopy scenarios, LESS-Hom, LESS-Row, and LESS-Forest, which represent horizontally uniform distribution scenes, ridge row structure scenes, and discrete forest canopy scenes, respectively.(**Figure 58**)



```
lai:          LIST    1/3
lad:          LIST    Spherical
hotspot:      LIST    0.05
sza:          RANGE   20:0:21
saa:          RANGE   0:0:0
leaf_op:      LIST    0.0389,0.4715;0.0389,0.4715;0.0187,0.4860
soil_ref:     LIST    0.1,0.1
sky_per:      LIST    0.0,0.0
vaa:          RANGE   0:0:0
vza:          RANGE   0:10:75
use_prospect
prosect_N:    LIST    1.5
Output directory:
```

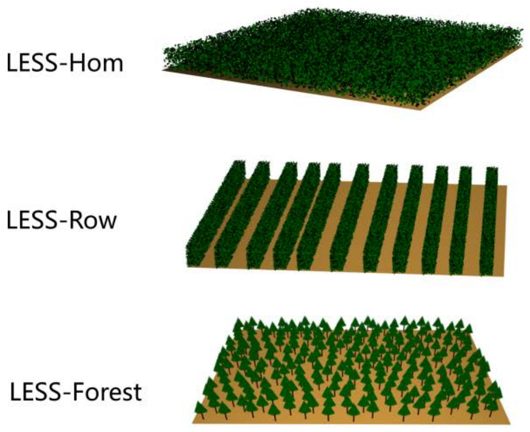


Figure 58. LESS1D Main interface and tree crown parameters

(1). Horizontal homogeneous scene 【LESS-Hom】

Parameters	Description
lai	The LAI value of the scene in LESS-Hom.
lad	The leaf angle distribution type: Erectophile vertical type, Exemophile extreme type, Plagiophile slanted type, Planophile plane type; Spherical spherical type, Uniform uniform type.
hotspot	The hot spot calculation formula is the ratio of leaf length to canopy height.
leaf_op	Leaf spectrum. When “use_prospect” is checked, LESS1D will automatically calculate the leaf spectral parameters based on the “prospect” parameter.
soil_op	Soil spectrum (the number of bands needs to be consistent with vegetation, etc.).
sky_per	Sky light proportion (the number of bands needs to be consistent with soil and vegetation, etc.).
sza/saa，vza/vaa	The solar zenith angle/azimuth angle, the viewing zenith angle/azimuth angle.

(2). Ridge row structure scene 【LESS-Row】

Parameters	Description
row_width	Ridge width
plant_height	Ridge height
ridge_spacing	Ridge spacing

(3). Ridge row structure scene 【LESS-Forest】

Parameters	Description
lai_single_tree	The LAI of LESS-Forest is the single tree LAI value.
crown_shape	Crown shape: Ellipsoid, Cylinder, Corn, Cube.
stem_density	The number of trees in the scene, i.e., tree density. (Scene area is 100*100m)
tree_height	Tree height
crown_diameter	Crown diameter
crown_length	Crown length
dbh	Diameter breast height

4.【Advanced】settings

If **[has_fpar]** is not checked, the simulation results will only output BRF values; if **[has_fpar]** is checked, the simulation results will output both BRF and FPAR values.

In the **[fpar_layers]** setting, the scene is divided into layers to obtain the fPAR of each layer. The format is “starting height: step length: ending height”.

In the **[sun_spec]** and **[sky_spec]** fields, the direct and scattered radiation values of the scene are entered, respectively.

5.Simulation results

After setting the parameters, set the storage and output path in **[Browse]**, click **[Run]** in the lower-left corner of the simulated scene page to start the simulation, and view the results in the [Result] path after the simulation is complete.

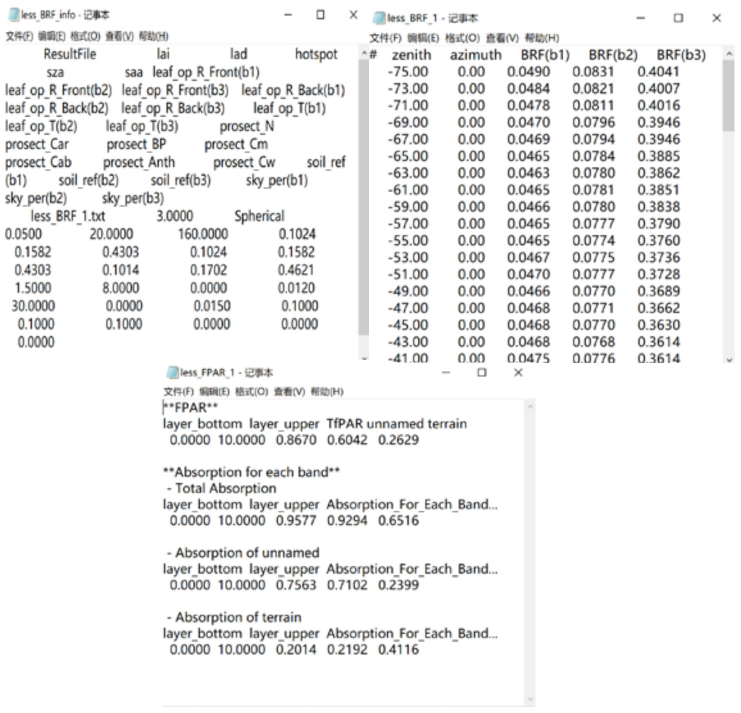
(1) BRF results

Open the **[less_BRF_info]** txt file in **[Result]** to view the input parameters. The BRF simulation results are stored in the **[less_BRF]** txt file. In this exercise, three bands (600nm, 700nm, 800nm) are set, and the observation angle range is “-75:2:75”. The BRF values of each band at different observation angles can be obtained from the graph.

(2) FPAR results

Open the **[less_FPAR]** txt file in **[Result]** to view the FPAR simulation results.(**Figure 59**)

Simulation results	Description
TfPAR	Total FPAR value of the scene, including total leaf FPAR and total soil FPAR.
Unnamed （component name）	Total leaf FPAR
terrain	Total soil absorption
Total Absorption	FPAR value of each band in the scene
Absorption of unnamed/terrain	FPAR value of each band for leaves/soil.


less_BRF_info - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
less_BRF_1 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
less_FPAR_1 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
     ResultFile          lai          lad          hotspot
     sza          saa  leaf_op_R_Front(b1)
leaf_op_R_Front(b2)  leaf_op_R_Front(b3)  leaf_op_R_Back(b1)
leaf_op_R_Back(b2)  leaf_op_R_Back(b3)      leaf_op_T(b1)
leaf_op_T(b2)      leaf_op_T(b3)        prosect_N
prosect_Car       prosect_BP        prosect_Cm
prosect_Cab       prosect_Anth        prosect_Cw       soil_ref
(b1)       soil_ref(b2)       soil_ref(b3)        sky_per(b1)
sky_per(b2)        sky_per(b3)
    less_BRF_1.txt           3.0000          Spherical
0.0500           20.0000           160.0000            0.1024
  0.1582           0.4303            0.1024            0.1582
  0.4303           0.1014            0.1702            0.4621
  1.5000           8.0000            0.0000            0.0120
 30.0000            0.0000            0.0150            0.1000
  0.1000           0.1000            0.0000            0.0000
  0.0000
```

```
#  zenith   azimuth   BRF(b1)   BRF(b2)   BRF(b3)
   -75.00     0.00    0.0490    0.0831    0.4041
   -73.00     0.00    0.0484    0.0821    0.4007
   -71.00     0.00    0.0478    0.0811    0.4016
   -69.00     0.00    0.0470    0.0796    0.3946
   -67.00     0.00    0.0469    0.0794    0.3946
   -65.00     0.00    0.0465    0.0784    0.3885
   -63.00     0.00    0.0463    0.0780    0.3862
   -61.00     0.00    0.0465    0.0781    0.3851
   -59.00     0.00    0.0466    0.0780    0.3838
   -57.00     0.00    0.0465    0.0777    0.3790
   -55.00     0.00    0.0465    0.0774    0.3760
   -53.00     0.00    0.0467    0.0775    0.3736
   -51.00     0.00    0.0470    0.0777    0.3728
   -49.00     0.00    0.0466    0.0770    0.3689
   -47.00     0.00    0.0468    0.0771    0.3662
   -45.00     0.00    0.0468    0.0770    0.3630
   -43.00     0.00    0.0468    0.0768    0.3614
   -41.00     0.00    0.0475    0.0776    0.3614
```

```
**FPAR**
layer_bottom  layer_upper  TfPAR unnamed terrain
  0.0000  10.0000   0.8670   0.6042   0.2629

**Absorption for each band**
 - Total Absorption
layer_bottom  layer_upper  Absorption_For_Each_Band...
  0.0000  10.0000   0.9577   0.9294   0.6516

 - Absorption of unnamed
layer_bottom  layer_upper  Absorption_For_Each_Band...
  0.0000  10.0000   0.7563   0.7102   0.2399

 - Absorption of terrain
layer_bottom  layer_upper  Absorption_For_Each_Band...
  0.0000  10.0000   0.2014   0.2192   0.4116
```

Figure 59. LESS1D simulation results

References

[1] Y. M. Govaerts and M. M. Verstraete, “Raytran: A Monte Carlo ray-tracing model to compute light scattering in three-dimensional heterogeneous media,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 36, no. 2, pp. 493–505, 1998.

[2] B. Beckers and P. Beckers, “A general rule for disk and hemisphere partition into equal-area cells,” *Computational Geometry*, vol. 45, no. 7, pp. 275–283, Aug. 2012.

[3] M. I. Disney, P. Lewis, and P. North, “Monte Carlo ray tracing in optical canopy reflectance modelling,” *Remote Sensing Reviews*, vol. 18, no. 2–4, pp. 163–196, 2000.

[4] T. Yin, N. Lauret, and J.-P. Gastellu-Etchegorry, “Simulating images of passive sensors with finite field of view by coupling 3-D radiative transfer model and sensor perspective projection,” *Remote Sensing of Environment*, vol. 162, pp. 169–185, Jun. 2015.

[5] J.-L. Widlowski, T. Lavergne, B. Pinty, M. Verstraete, and N. Gobron, “Rayspread: A virtual laboratory for rapid BRF simulations over 3-D plant canopies,” *Computational methods in transport*, pp. 211–231, 2006.

[6] R. L. Thompson and N. S. Goel, “Two models for rapidly calculating bidirectional reflectance of complex vegetation scenes: Photon spread (PS) model and statistical photon spread (SPS) model,” *Remote Sensing Reviews*, vol. 16, no. 3, pp. 157–207, Mar. 1998.

[7] J. T. Kajiya, “The rendering equation,” in *ACM Siggraph Computer Graphics*, 1986, vol. 20, pp. 143–150.

[8] H. Kobayashi and H. Iwabuchi, “A coupled 1-D atmosphere and 3-D canopy radiative transfer model for canopy reflectance, light environment, and photosynthesis simulation in a heterogeneous landscape,” *Remote Sensing of Environment*, vol. 112, no. 1, pp. 173–185, Jan. 2008.