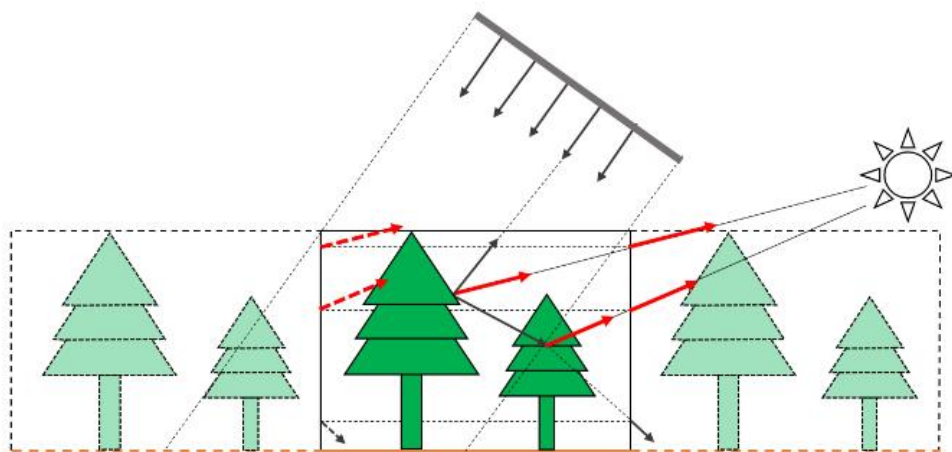


# 三维辐射传输模型 LESS 原理与实践

## ——简明教程

漆建波 编写



2019 年 7 月



## 序 言

LESS 是一个年轻的模型，它还有许多问题，但是经过 6 年的成长，我想它值得拥有一份属于它本身的教程，以便让大家更好地认识它，这也是我要编写这样一份教程的原始出发点。

《三维辐射传输模型 LESS 原理与实践（简明教程）》是专门为 LESS 辐射传输模型编写的学习教程，主要为初学者介绍 LESS 模型的基本原理，并且通过一组示例介绍 LESS 模型的基本操作。本教程力求简洁，希望能够利用最短的篇幅使初学者能够利用最短的时间掌握 LESS 模型的使用方法。对于想要了解 LESS 模型原理的用户，可以参考本教程的第一章，若不需要了解原理，可以直接从第二章开始学习。随着 LESS 应用的深入，本教程也在不断地修正、扩展和深化。

本教程在编写过程中受到了许多老师和同学的帮助与支持，感谢博士阶段的导师谢东辉老师和阎广建老师对 LESS 模型的大力支持，感谢课题组的许月对本教程的编写所付出的巨大努力，也感谢各位 LESS 模型的用户所提出的宝贵意见。

漆建波

2019 年 8 月 于北京林业大学



# 目 录

|                             |    |
|-----------------------------|----|
| 序 言 .....                   | I  |
| 目 录 .....                   | 1  |
| 第 1 章 遥感三维辐射传输模型 LESS ..... | 3  |
| 1.1 三维辐射传输模型发展现状 .....      | 3  |
| 1.1.1 基于辐射度算法的模型 .....      | 4  |
| 1.1.2 基于光线追踪算法的模型 .....     | 5  |
| 1.2 LESS 模型基本原理 .....       | 7  |
| 1.2.1 三维结构表示 .....          | 7  |
| 1.2.2 场景元素管理 .....          | 9  |
| 1.2.3 光线追踪基本原理 .....        | 10 |
| 1.2.4 前向光子追踪模型 .....        | 11 |
| 1.2.5 后向路径追踪模型 .....        | 15 |
| 1.2.6 热红外影像模拟 .....         | 18 |
| 1.2.7 植被冠层分层 FPAR 模拟 .....  | 19 |
| 1.2.8 复杂地形区上下行辐射模拟 .....    | 20 |
| 1.2.9 晴空大气辐射传输模拟 .....      | 21 |
| 1.2.10 土壤非朗伯反射模型 .....      | 26 |
| 第 2 章 LESS 模型实践练习 .....     | 28 |
| 2.1 操作案例 .....              | 28 |
| 2.1.1 多光谱影像模拟 .....         | 28 |
| 2.1.2 鱼眼相机模拟 .....          | 34 |
| 2.1.3 多角度反射率的模拟 .....       | 35 |
| 2.1.4 分层分组分 FPAR 模拟 .....   | 38 |
| 2.1.5 方向间隙率模拟 .....         | 40 |
| 2.1.6 热红外影像的模拟 .....        | 42 |

|                                          |    |
|------------------------------------------|----|
| 2.1.7 激光雷达数据模拟 .....                     | 42 |
| 2.1.8 日光诱导叶绿素荧光模拟 .....                  | 47 |
| 2.1.9 分层光照/阴影叶片面积比例* .....               | 49 |
| 2.1.10 TRAC 模拟* .....                    | 52 |
| 2.1.11 鱼眼影像透过率模拟* .....                  | 53 |
| 2.1.12 灯光模拟* .....                       | 53 |
| 2.1.13 通量模拟 (Flux) * .....               | 54 |
| 2.1.14 再碰撞概率模拟 .....                     | 55 |
| 2.1.15 浑浊介质表达叶片 .....                    | 56 |
| 2.2 实用工具 .....                           | 58 |
| 2.2.1 LAI 计算工具 .....                     | 58 |
| 2.2.2 批处理工具 .....                        | 58 |
| 2.2.3 影像和 LiDAR 波形查看器 .....              | 59 |
| 2.2.4 简单树冠生成工具 .....                     | 60 |
| 2.2.5 从机载点云重建三维场景 .....                  | 60 |
| 2.2.6 Python 代码自动生成工具 .....              | 62 |
| 2.2.7 Semi-LESS 快速模拟数据集 .....            | 62 |
| 2.3 LESS 文件格式 .....                      | 64 |
| 2.3.1 instance.txt .....                 | 64 |
| 2.4 SDK for Python .....                 | 65 |
| 2.4.1 利用 SDK for Python 实现 LESS 模拟 ..... | 65 |
| 2.4.2 树冠生成 .....                         | 75 |
| 第 3 章 LESS 扩展模块 .....                    | 79 |
| 3.1 LESS 1D 模型 .....                     | 79 |
| 3.2 LESS Plugins .....                   | 83 |
| 3.2.1 插件的使用 .....                        | 83 |
| 3.2.2 插件的开发 .....                        | 83 |
| 3.2.3 批量调用插件 .....                       | 84 |
| 参考文献 .....                               | 85 |
| 附录 .....                                 | 86 |
| 附录 1 SDK for Python 文档 .....             | 86 |
| SimulationHelper .....                   | 86 |

|                                |    |
|--------------------------------|----|
| Simulation .....               | 86 |
| Scene .....                    | 87 |
| Landscape .....                | 87 |
| Sensor (所有 Sensor 的基类) .....   | 87 |
| SensorOrthographic .....       | 88 |
| SensorPerspective .....        | 88 |
| SensorFisheye .....            | 88 |
| SensorPhotonTracing .....      | 88 |
| ObservationOrthographic .....  | 89 |
| ObservationPhotonTracing ..... | 89 |

## 第 1 章 遥感三维辐射传输模型 LESS

### 1.1 三维辐射传输模型发展现状

卫星遥感能够长时间、大范围对陆表植被进行监测，然而光学遥感卫星传感器往往只能获得单一（或几个）角度以及有限波段的影像数据，其所记录的辐射值也是经过地表反射以及大气作用后的综合结果，要从卫星观测中反演地表参数以实现地表特征的定量监测，就需要充分理解太阳辐射在大气以及地表的传输过程。对于植被冠层来说，其对太阳辐射在不同波段和不同角度都表现出了不同的反射特征(Zhang et al., 2017)，如何利用这些特征去推断植被冠层自身的参数（例如光合有效辐射）是植被遥感监测研究中的重要问题，也是植被遥感中的核心问题之一。

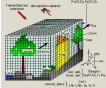
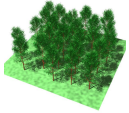
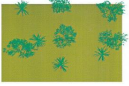
为了描述太阳辐射与植被冠层的相互作用，在植被遥感领域，先后出现了一维辐射传输模型、几何光学模型、混合模型以及基于植被三维结构的计算机模拟模型。一维辐射传输模型（例如 SAIL 模型）将植被冠层简化为垂直分层且水平均匀的场景，从而将植被冠层的反射辐射用一组解析的数学公式表达出来(Verhoef et al., 2007)。几何光学模型则是将树冠抽象为简单几何体（例如椭球体），从而可以利用几何投影特征来描述冠层的方向反射率，能够在一定程度上考虑像元内的异质性。三维辐射传输模型往往以真实场景结构为基础，可以模拟从叶片尺度到冠层尺度的反射特征。相较于其他模型，减少了对场景进行简化的要求，从而使得三维模型能够更加准确地考虑地表的异质性以及植被冠层的内部结

构。

近年来，三维辐射传输模型受到了越来越多的重视，逐渐成为研究异质地表植被冠层反射辐射特征的重要工具(Béland et al., 2014; Schneider et al., 2014; Widlowski et al., 2013)。利用三维辐射传输模型，可以定量地分析太阳辐射在植被冠层中的传输过程(Disney et al., 2006)，也可以用来发展一些参数化模型，并用于遥感产品的生产(Cahalan et al., 2005; Van Laake and Sanchez-Azofeifa, 2004; Zarco-Tejada et al., 2003)。除此之外，三维辐射传输模型也能够辅助传感器设计，用于卫星发射前的可行性研究(Goodenough and Brown, 2017; Kuester et al., 2014)。

目前，在遥感领域，已经发展了一系列的三维辐射传输模型(Kuusk, 2018)，这些模型大多数都是基于真实场景结构的模型。尽管各个模型的实现方法具有显著的差异(表 1-1)，但是目前的主流算法依然是辐射度和光线追踪算法。

表 1-1 典型的三维辐射传输模型

| 模型名称    | 模型原理         | 范围  | 场景形式  | 典型场景                                                                                | 参考文献                               |
|---------|--------------|-----|-------|-------------------------------------------------------------------------------------|------------------------------------|
| FLIGHT  | 光线跟踪         | 小场景 | 简化几何体 |   | (North, 1996).                     |
| DART    | 离散坐标         | 百米级 | 体元模型  |  | (Gastellu-Etchegorry et al., 1995) |
| Raytran | 光线跟踪<br>(前向) | 百米级 | 真实结构  |  | (Govaerts and Verstraete, 1998)    |
| RGM     | 辐射度          | 小场景 | 真实结构  |  | (Qin and Gerstl, 2000)             |
| RAPID   | 辐射度          | 公里级 | 简化结构  |  | (Huang et al., 2013)               |

### 1.1.1 基于辐射度算法的模型

辐射度方法来源于热辐射工程中的能量守恒原理。假定场景中所有面元均为朗伯体，每一个面元所发出的辐射等于其自身发射的辐射加上反射和透射的其他面元到达该面元的辐射能。这种辐射平衡状态能够使用一个方程组进行定量表达，从而求解每个面元的辐射分布。Gerstl 和 Borel 是将该方法运用于植被散射辐射特性分析的主要贡献者(Borel et al., 1994)。Goel 等基于该方法发展了 DIANA 模型(Goel et al., 1991)，该模型基于 L 系统在计

计算机上生成任意三维植被模型，并利用辐射度方法计算了可见光和近红外的二向反射因子，真正把辐射度模型发展到了真实冠层结构的 BRF 模拟上。同样基于辐射度方法，Qin 等人发展了基于真实结构的 RGM 模型(Qin and Gerstl, 2000)，并用于半干旱地区植被二向反射模拟，与实测数据相比，取得了较好的一致性。辐射度方法是一个全局求解方法，计算完所有面元的相互作用后，即能够同时获得多个观测角度的数据。但是辐射度算法需要计算任意两个面元之间的可视因子 (View factor)，因此当场景中面元较多时，计算效率很低，甚至对于具有大量面元的场景存在无法计算的情况。为了解决此问题，Huang 等人在 RGM 的基础上发展了 RAPID 模型(Huang et al., 2013)，该模型用多孔介质表示面元，从而大大地减少了场景中面元的个数，使得辐射度方法可以模拟大尺度的场景。

### 1.1.2 基于光线追踪算法的模型

与辐射度算法相比，光线追踪算法在三维辐射传输模型中的使用更为普遍，因为光线追踪算法对不同复杂程度的场景具有更好的适应性(Disney et al., 2000)。根据不同的研究目标，光线追踪算法有前向光线追踪和后向光线追踪两种实现方式。前向光线追踪是指光线从光源出发进入场景，经场景中元素反射后进入传感器的过程，而在后向光线追踪中，光线从传感器出发去计算光源沿该方向进入传感器的能量。前向光线追踪更适用于计算多角度 BRF 以及解决与能量平衡相关的问题。例如，DART 模型就是完全用前向光线追踪实现的三维辐射传输模型(Gastellu-Etchegorry et al., 2015)。在 DART 模型中，光线沿着固定的离散方向（离散坐标法）进行传播，并在场景顶部被搜集。目前，DART 模型可以模拟多角度反射率，能量收支以及光合有效辐射等多种遥感数据。除了 DART 模型之外，Raytran(Govaerts and Verstraete, 1998)、Rayspread(Widlowski et al., 2006)、FLiES(Kobayashi and Iwabuchi, 2008)以及 FLIGHT(North, 1996)模型等都是典型的前向光线追踪模型。在这些模型中，Raytran 是基于蒙特卡罗光线追踪算法实现的：光线从光源出发进入场景，当其离开场景时，被放置在场景顶部的半球所搜集，用于计算方向反射率。由于 Raytran 模型没有使用任何权重机制（即光线完全吸收或者完全散射），因此，Raytran 模型的计算效率很低，通常需要大量的光线才能够得到稳定的结果。那些被完全吸收的光线最终不会离开场景，对结果没有任何贡献，也会造成一定的效率损失。此外，由于叶片的反射率和吸收率随波段不同而不同，因此利用 Raytran 模型模拟多波段方向反射率 BRF 时，每个波段都需要模拟一次。为了解决 Raytran 模型效率低的问题，Widlowski 等对其进行了扩展并发展了 Rayspread 模型(Widlowski et al., 2006)，该模型引入了一种“副光线 (Secondary ray)”的机制，即在光线与场景元素的每一个交点处向传感器发射一条光线并计算入射光线沿传感器方向可能散射的能量，这种机制能够以较少的入射光线数量得到较精确的结果。为了能够同时模拟多个波段，Lewis 等人发展了 librat 模型(Lewis, 1999)，该模型使用了一种称

之为“光线束 (Ray bundle)”的概念, 即每条光线是包含了多个波段的能量束, 在每个碰撞点, 光线不会完全吸收, 而是根据光谱反射率及吸收率对光线所携带的各个波段的能量进行更新, 使得每条光线都能够对最终结果有贡献, 进一步提高了计算效率。

虽然前向光线跟踪算法能够在一次模拟中同时模拟多个角度的 BRF, 但是在模拟特定方向影像方面的效率却不高, 其主要原因是很多正向传播光线最终并没有进入传感器, 导致大量的冗余计算, 尤其是传感器视场只覆盖了一个大场景中很小的一部分时, 这种方式计算效率尤其低。然而, 前向光线追踪的弱点却是后向光线追踪的优点, 因为后向光线追踪只追踪那些进入传感器的光线, 而未被传感器视场覆盖的部分则不参与计算。这类模型的典型代表是 DIRSIG 模型(Goodenough and Brown, 2012), 这种特性使得 DIRSIG 模型尤其适合用来研究传感器硬件参数对记录的遥感信号的影响, 从而指导传感器硬件的设计。除此之外, VBRT(Li et al., 2018)也是基于后向光线追踪的辐射传输模型, 得益于后向模拟的高效率, 该模型能够模拟具有精细三维结构的场景的遥感信号。

因此, 能够同时进行前向光线跟踪和后向光线跟踪的辐射传输模型在模拟遥感数据方面具有更强的实用性, 而目前在遥感领域还不存在这样的模型。

在计算机图形学领域, 光线追踪已经成为了主流的渲染算法, 代表性的渲染器有 POV-ray (Plachetka, 1998)、PBRT (Pharr et al., 2016) 以及 Maxwell (maxwellrender.com) 等。这些渲染软件充分利用了光线追踪的可并行性以及最新的 CPU 加速优化方法 (SSE2 指令集等), 从而能够达到较高的执行效率。但是它们主要关注点仍然是图像模拟的真实感效果而不是物理量的准确性。因此如何充分利用现有光线追踪算法优势, 结合已有程序基础, 开发一个适用于遥感数据和影像模拟的三维辐射传输模型是一种更为经济的解决方案。例如劳彩莲基于 POV-Ray 开发了一个冠层三维辐射传输模型, 较早地将计算机图形学领域的技术应用于植被冠层辐射传输的模拟(劳彩莲, 2005)。同样基于 POV-Ray, Auer 等人发展了一个可以模拟微波后向散射信号的模型(Auer et al., 2016), 该模型能够模拟复杂的地面场景并且具有较高的计算效率。近期, DIRSIG 模型进行了一次代码重构, 其主要目的就是要引入目前计算图形学领域中高效的光线传播算法(Goodenough and Brown, 2017)。此外, DART 模型也集成了 Intel 开发的开源高性能光线追踪程序 Embree (Wald et al., 2014), 用于提高异质场景中的光线跟踪效率。

近年来, 随着计算机硬件的发展, 基于 GPU (Graphics Processing Unit) 的加速方法得到越来越多的关注(Bailey et al., 2014; Efremenko et al., 2014; Malik et al., 2017)。GPU 本身是为图形处理专门设计的, 但是由于其往往具有较强的并行能力, 因此目前已经大量应用于科学计算之中。为了解决复杂冠层的辐射传输问题, Bailey et al. (2016) 开发了一个基于 GPU 加速的植被辐射传输模型, 能够模拟几十万棵单木组成的场景, 但是这些单木是经过高度简化的 (体元模型), 无法表示冠层内部细节。与 CPU 相比, 虽然 GPU 并行能力较强, 但是 GPU 往往只有有限的内存, 并且 GPU 不能独立于 CPU 运行, 所以实际计算过程

中首先需要将数据从 CPU 传输至 GPU，计算完成后再传回 CPU，对于数据量比较大的应用，这个过程是相当耗费时间的。除此之外，对于具有复杂结构的真实场景，其数据量往往超过很大，无法在 GPU 上直接运行。因此，目前三维辐射传输模型仍以 CPU 为主。

综上所述，目前已有的三维辐射传输模型大部分以简化场景或者只模拟小范围的真实场景为主，难以满足模拟大尺度复杂场景的需求，因此发展一套既能够处理真实结构场景又能够模拟大尺度遥感数据的三维辐射传输模型显得尤为必要。这也是发展三维辐射传输模型 LESS 的初衷。

## 1.2 LESS 模型基本原理

LESS (Large-Scale remote sensing data and image Simulation framework) 是基于光线追踪三维真实结构辐射传输模型，它可以模拟入射光在场景中的传输过程（吸收、反射以及透射），并输出相应的模拟数据（例如反射率，反照率，fPAR 等）。所谓真实结构是指通过三角面片表示的具有任意复杂结构的场景元素（例如树木、房屋等），这里的真实结构是相对于传统遥感模型中将树冠假想成椭球体以及将树叶表达为连续均匀介质等简化方式而言的，因此 LESS 模型本身对场景的复杂程度没有任何限制。

LESS 模型的基本输入是场景的三维结构和组分光谱、观测几何、光照参数等，通过光线追踪的方式模拟得到所需要的遥感数据。

### 1.2.1 三维结构表示

在 LESS 模型中，除简单几何体外（例如平面、椭球体等），场景主要以三角面片的形式来表示，这种形式可以较为准确地描述物体真实的表面结构，通常而言，三角面元的个数越多，拟合的表面越接近真实表面，但是相应地需要耗费更多的计算和存储资源。图 1-1 展示了用三角网表示的一个森林场景以及局部放大图。从图中可以看出，每个叶片是用一系列的三角形来表示的。

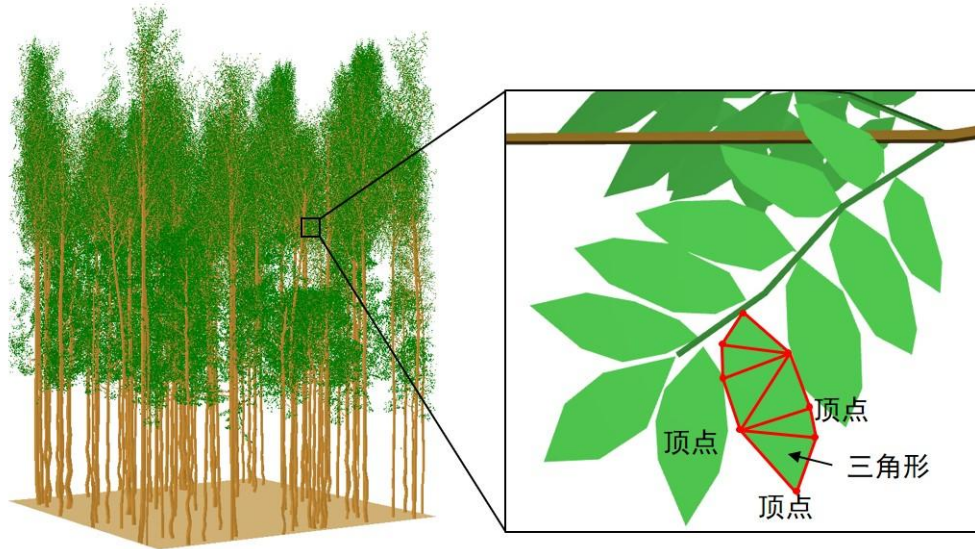


图 1-1 LESS 中用面元描述三维结构

一般而言，三角网中的每个三角形有三个顶点（Vertex），那么对三角网最自然的储存方式就是对每个三角形（Triangle）分别储存其对应的顶点，因此  $N$  个三角形将储存  $3N$  个顶点，但是这种方式可能造成大量的冗余信息，因为一个顶点可能同时属于多个三角形。因此一种更为经济的储存方式将三角网的所有顶点保存为一个数组，然后只保存每个三角形三个顶点在该数组中的索引。图 1-2 展示了一种保存三角网的数据结构：所有顶点的 XYZ 坐标均保存在一个顶点数组中（Vertex array），三角形顶点索引列表（Triangle list）则保存每个三角形的三个顶点在顶点数组中的位置信息，如 Triangle 0 的三个顶点坐标分别为  $(x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)$ 。

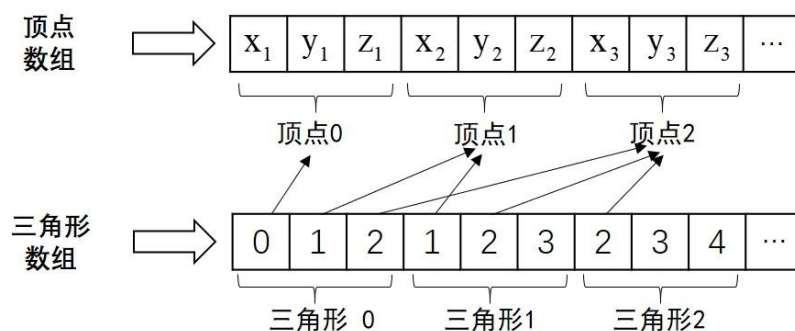


图 1-2 三角网的储存形式

描述这种三维结构的一种典型文件格式是 Wavefront OBJ 文件，它是一种标准的 3D 模型格式文件，被许多三维建模软件所支持。本质上，OBJ 文件是一个文本文件，并且不需要任何头文件，可以直接使用文本编辑器进行编辑。OBJ 文件主要用于描述多边形，并且支持纹理及法线。在 LESS 模型种，主要用到的主要是 OBJ 文件所描述的空间位置关系，因此无需纹理及法线信息。OBJ 文件由一行行文本组成，每一行由一个关键词开始以说明

该行的属性，常用的关键字是 **g**、**v**、**f** 等。**G** 表示一个组、**b** 表示顶点、**f** 表示面元。典型的 OBJ 文件如下所示：

```
v 1 0 1
v 0 0 1
v 1 0 0
v 0 0 0
g t1
f 1 2 3
g t2
f 2 3 4
```

该文件表示了一个在水平面内的两个三角形，前四行表示了四个顶点，分别为  $(1, 0, 1)$ 、 $(0, 0, 1)$ 、 $(1, 0, 0)$  和  $(0, 0, 0)$ ，第五和第七行表示分组，第五行和第七行之间的表示第一个分组，其名称为“t1”。第七行之后的表示第二个分组，其名称为“t2”。**f** 表示面元，即用第 1、2 和 3（或者 2、3 和 4）个顶点组成一个三角形（**f** 也支持 4 个顶点组成的多边形）。需要注意的是，在 LESS 中导入的 OBJ 文件为左手坐标系，Y 轴朝上、X 轴朝左、Z 轴垂直纸面朝里。

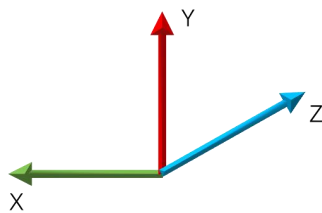


图 1-3 LESS 中 OBJ 文件的坐标系

### 1.2.2 场景元素管理

在 LESS 模型中，基本的场景元素单位是 **Object**，每一个 **Object** 代表一个或者一组对象，例如一棵树、一间房子。**Object** 用三角面片来表示，可以是导入的一个或多个 OBJ 文件。如图 1-4 所示，每一个 **Object** 可能有多个组分（**components**），每个组分可以定义自己的光学属性，因此在 LESS 中光学属性的定义是以组分为基本单元的。

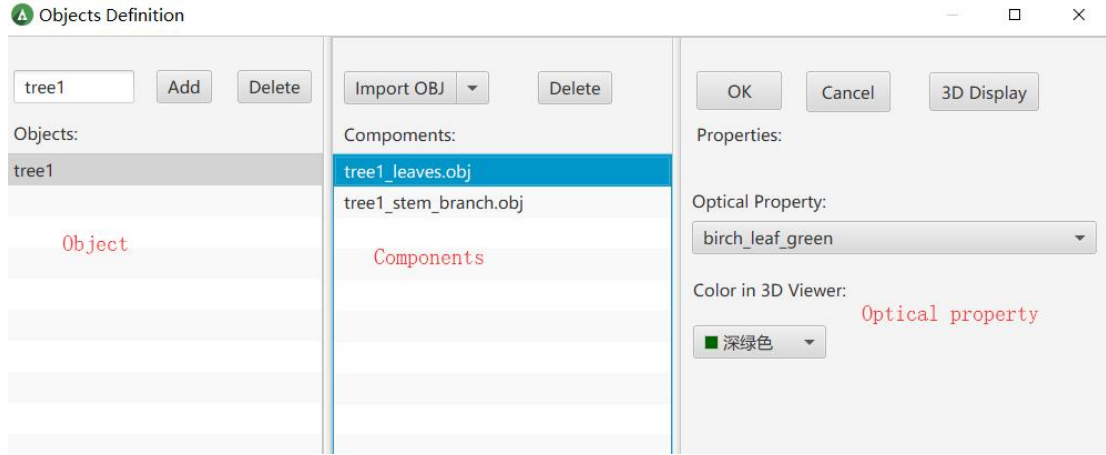


图 1-4 在 LESS 中定义 Object

需要注意的是，在 LESS 中定义的 Object 并没有直接放置在场景中，而是需要将其分配到不同位置，每一个 Object 可以被分配到多个位置且只用保存几何变换矩阵，而无需复制整个 OBJ 文件的所有三角面片，这种技术被称之为实例化（Instancing）技术，能够显著地减少内存的使用。

### 1.2.3 光线追踪基本原理

在计算机图形学领域，光线追踪（Ray tracing）是指通过发射光线并追踪其传输路径以从三维场景生成二维图像的方法，就其本质而言，光线追踪就是计算光源发射的光线在所设定的虚拟场景中的反射、透射以及折射的过程，并最终得到传感器能够接收到的辐亮度信息。该过程可以用一个渲染方程来表示(Kajiya, 1986):

$$L_o(q, \omega_o) = L_e(q, \omega_o) + \int_{4\pi} f(q, \omega_i, \omega_o) L_i(q, \omega_i) |\cos \theta_i| d\omega_i \quad (1-1)$$

式中 $L_o(q, \omega_o)$ 为场景中从 $q$ 点出发沿 $\omega_o$ 方向出射的辐亮度； $L_e(q, \omega_o)$ 为 $q$ 点沿 $\omega_o$ 方向发射的辐亮度（例如热辐射）； $f(q, \omega_i, \omega_o)$ 表示 $q$ 附近的双向散射分布函数（Bidirectional Scattering Distribution Function, BSDF）； $L_i(q, \omega_i)$ 为 $q$ 点附近沿 $\omega_i$ 方向的入射辐亮度； $\theta_i$ 表示入射光线与局部表面法向量之间的夹角。整个积分式表示从 $q$ 点沿 $\omega_o$ 方向出射的辐亮度等于该点处物体自身发射的辐亮度以及周围所有入射辐亮度经过该点散射到出射方向的辐亮度之和。

因此，一个完整的光线追踪过程至少应当包含以下几个步骤（如图 1-5）：

- **三维场景的几何描述**，即整个虚拟场景结构应当用一定的方法在计算机中进行完整地描述，场景中的元素一般可以用简单几何体（如球体）、三角网（Triangle mesh）以及浑浊介质（Turbid medium）等进行描述，用这些元素进行描述的目的在于计算光线与场景元素之间的交点。

- **光线的产生。**光线一般从场景中的辐射源出发而进入场景，自然界中典型的辐射源有太阳以及大气等。在后向光线追踪模式下，光线不是从辐射源本身出发，而是从传感器产生并进入场景。
- **光线与场景交点的计算。**求交计算是光线追踪中最为核心也是最耗时的部分。因为光线需要与每一个场景元素进行求交测试，进而确定其与某个场景元素是否有交点。为了提高效率，往往将场景元素用一些空间加速数据结构进行管理（如层次包围盒 BVH: Bounding Volume Hierarchy），以提高求交计算的效率。
- **场景元素光学属性的确定。**对于每一个光线与场景的交点，根据该点处的光学属性确定光线能量的改变以及散射光线的方向。一般而言，物体表面的光学属性用双向散射分布函数 BSDF 来表述，它表示入射光被散射到特定方向的能量大小，这里的散射主要指光线与场景元素发生碰撞之后的反射、透射以及折射现象。
- **辐射信息的计算。**通过跟踪光线在场景中每一次碰撞之后的能量变化，可以计算光线最终逃逸出场景之后的能量，以得到传感器所记录的辐亮度信息。除此之外，还可以根据需要从光线传播的过程中提取多种多样的辐射信息，比如计算特定场景元素所接收的能量。

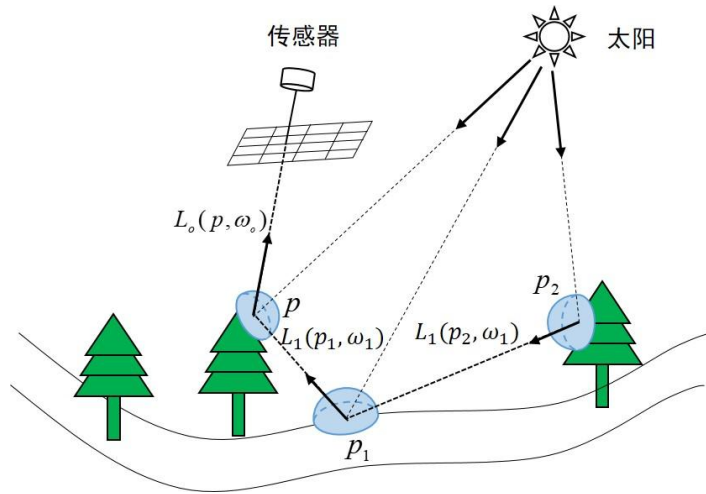


图 1-5 光线追踪基本原理

$L_o(p, \omega_o)$ 表示从 $p$ 沿 $\omega_o$ 方向出射并进入传感器的辐亮度， $L_1(p_1, \omega_o)$ 为 $p_1$ 点对 $p$ 贡献的辐亮度。

在 LESS 中，按照实现方式，光线追踪可以分为前向光子追踪和后向路径追踪。

#### 1.2.4 前向光子追踪模型

前向光子追踪模型是从光源发射光子，跟踪光子在场景中的散射路径，并计算最终遥感传感器接收能量的过程。换句话说，前向光子追踪模型就是模拟了真实自然界太阳辐射

入射并被散射的过程。

#### 1.2.4.1 光子的产生

在 LESS 中，主要考虑的光源有太阳直射入射和天空散射光入射，一般将太阳直射入射作为平行光处理，而天空散射光则作为各向同性入射光处理。

对于直射光，LESS 将其表示为一个投影后能够覆盖整个场景的圆盘，该圆盘的大小和位置由场景外接球决定。如图 1-6，光子的起始位置位于外接球的外接圆柱上一定距离的圆盘上，假设沿太阳入射方向的辐照度为  $E_{dir}$  ( $\text{w}/\text{m}^2$ )，则整个光源的发射功率为  $L_{dir} = \pi r^2 E_{dir}$ 。

对于散射光而言，光子的起始位置为包围整个场景的一个半球之上，该半球的半径通常是场景外接球半径的 2~3 倍。光子的方向在半球起始位置局部的半球空间中随机采样，即光子方向以局部表面为准以朗伯余弦的形式分布（如图 1-6）。假设天空散射光的辐亮度为  $I_{diff}$ ，那么天空的发射功率为  $L_{diff} = 2\pi^2 r^2 I_{diff}$ 。

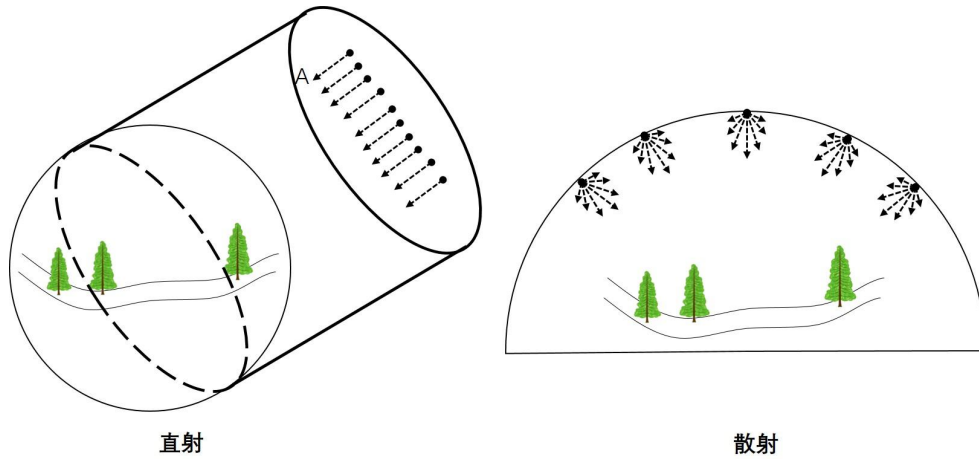


图 1-6 直射光和天空散射光

假设在一个模拟过程中产生的总的光子数为  $N$ ，那么每个光子的能量  $P^0(\lambda)$  (瓦特) 可以表示为

$$P^0(\lambda) = \frac{L_{dir} + L_{diff}}{N} \quad (1-2)$$

当产生这  $N$  个光子时，在直射光源和天空散射光源上产生的光子数是按照一定权重进行分配的，即  $w_k = \frac{L_k(\lambda)}{\sum_{k=1}^K L_k(\lambda)}$ ，式中  $K$  为光源数量。需要注意的是，此处的光子是一个能量包的概念，即包含了不同波段的能量，而且该光子的能量是可变的，在传输过程中，可能根据反射或者透射过程按不同波段的反射率对不同波段的能量进行修改。当光子的起始位置和方向确定之后，就可以计算该光子与场景的交点，进而跟踪其传输路径。

### 1.2.4.2 真实光子跟踪算法

当光子进入场景并与场景中的元素发生碰撞之后，其能量会发生改变，能量的改变量与发生碰撞的次数以及碰撞点的 BSDF 有关，对于一个发生了  $Q$  次碰撞的光子，其能量将变为

$$P^Q(\lambda) = P^0(\lambda) \cdot \prod_{q=1}^Q [\pi f(q, \omega_i, \omega_o, \lambda) / p_q] \quad (1-3)$$

其中  $f(q, \omega_i, \omega_o, \lambda)$  表示在第  $q$  个交点处的 BSDF 函数； $p_q$  为光子被反射或者透射的概率，在 LESS 中，每一次碰撞后光子可能会被反射，也可能发生透射，一般将反射和透射的概率都设置为 0.5。当发生碰撞后，光子的出射方向由 BRDF 或者 BTDF 决定，由于在 LESS 中采用了基于朗伯表面的假设，因此光子的方向按照朗伯余弦定律的形式在出射的半球空间进行随机采样。此外，由于光子的传播路径与波长无关，只根据不同波段的反射率/透过率对其波段能量进行修改，因此之后的章节将省略公式中的  $\lambda$  参数。

当光子从场景顶部逃逸离开场景之后，可能会被传感器所捕获。如果光子从场景侧面逃逸出场景，往往需要考虑场景的边缘效应，对于边缘效应的处理，通常将场景在水平方向进行复制，以模拟水平无限场景，但是更为简单且高效的方式是场景固定而改变光子的位置以达到同样的效果。如图 1-7 所示，当光子从场景边缘逃逸时，将会被重新移动到场景的对立面并以相同的方向进入场景，直到光子从场景顶部离开场景或者散射次数达到了预设值（例如 5 次）(Kobayashi and Iwabuchi, 2008)。为了避免人为截断光子传播而造成的能量损失，光子会按照一定的概率  $p$  终止其传输，典型的概率  $p$  为 0.05。如果光子没有成功被终止，那么该光子的能量会乘以一个概率值  $\frac{1}{1-p}$ 。

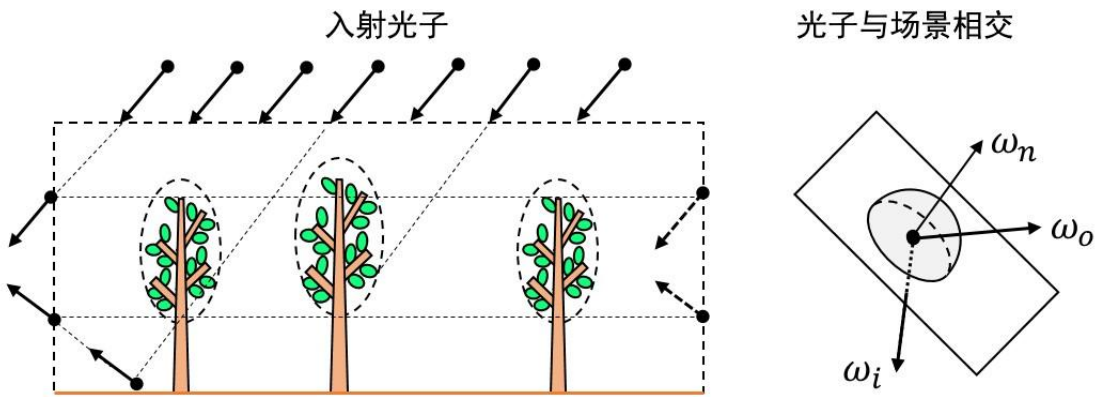


图 1-7 前向光子追踪算法示意图。

对于那些从场景顶部逃逸的光子，将会被放置于场景顶部的一个虚拟半球所搜集 (Govaerts and Verstraete, 1998)。为了计算多角度反射率，该半球被等分为  $N_p$  个面积相等的

小面元, 每个小面元的面积为  $\Delta S = \frac{2\pi}{N_p}$ 。如图 1-8 所示, 该剖分方法主要利用等面积投影法, 即将半球表面等面积投影到水平面形成一个圆盘, 由此可知  $\Delta\Omega = \Delta S$ 。在天顶方向 (圆环), 每个划分点的计算公式为

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

其中  $(\theta_i, \theta_{i-1})$  定义了半球上的一个圆环, 且  $\theta_0 = \frac{\pi}{2}$ ;  $k_i$  是天顶角  $\theta_i$  范围内包含的所有的面元数量, 且  $k_0 = N_p$ ;  $r_i$  为投影的圆环半径, 且  $r_i = 2 \sin \frac{\theta_i}{2}$ ;  $a_{aspect}$  是每个面元的宽高比, 一般被强制设定为 1; 每个圆环区域包含了  $k_{i-1} - k_i$  个面元。

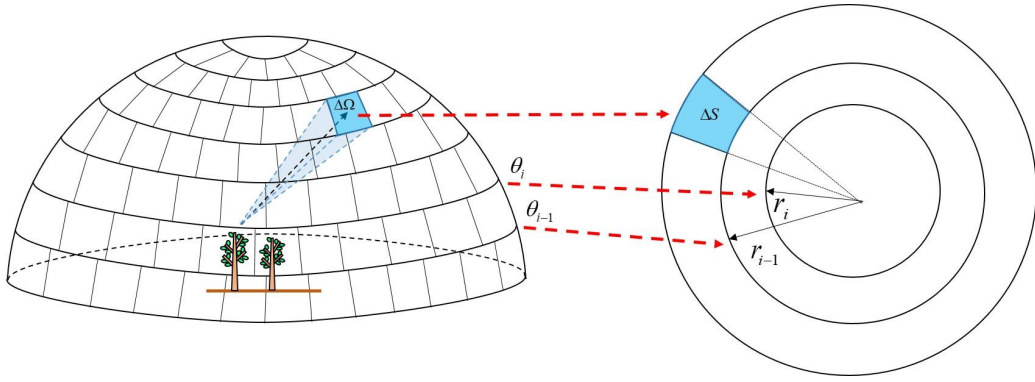


图 1-8 半球表面的等面积剖分方法

当光子离开场景, 其对应的面元由光子的方向决定, 与光子离开场景时最后一次碰撞位置无关, 这相当于把接收光子的半球放置到了无穷远处。对每一个面元, 可以计算一个对应的反射率, 即

$$f_{BRF_i} = \frac{\pi P_i^A}{\Delta\Omega_i \cdot \cos \theta_i^c \cdot P_{scene}} \quad (1-5)$$

式中  $P_i^A$  是面元  $i$  搜集到的总的光子能量, 即  $P_i^A = \sum_{P^Q \in \Delta\Omega_i} P^Q$ ;  $\Delta\Omega_i = \frac{2\pi}{N_p}$  为每一个面元对应的立体角;  $\theta_i^c$  是面元  $i$  的中心天顶角;  $P_{scene}$  是场景顶部的参考平面 (与场景大小一致) 所搜集到的能量。当每个面元的能量都搜集完成后, 整个场景的反照率可表示为

$$\omega_{albedo} = \frac{\sum_{i=1}^{N_p} P_i^A}{P_{scene}} \quad (1-6)$$

#### 1.2.4.3 虚拟光子跟踪算法

1.2.4.2 节中所描述的真实光子跟踪算法依赖于一个放置于场景顶部的半球, 并且 BRF 由等分小面元的立体角计算得到, 当小面元的面积越小时, 需要更多的光子才能得到较为

稳定的值。尽管如此，在一个面元内光子的方向仍然有一定的差异，因此通过该方法计算的 BRDF 并不是场景真正意义上的方向反射率(Disney et al., 2000)。为了解决该问题，在 LESS 中提出了虚拟光子追踪算法，该方法类似于 DART 模型中的“虚拟方向 (Virtual direction)” (Yin et al., 2015)，Rayspread 模型中的“副光线 (Secondary ray)” (Widlowski et al., 2006) 以及其他大气模型中的“局部估计 (Local estimates)” 法(Iwabuchi, 2006)。当一个光子与场景元素发生碰撞，并且没有被完全吸收的情况下，将会沿着虚拟方向（即需要计算 BRDF 的方向）发射一个虚拟光子，并且计算入射光子经过碰撞点可能沿虚拟方向所散射的能量 ( $W \cdot sr^{-1}$ )，即

$$I = V \cdot P^{q-1} \cdot f(q, \omega_i, \omega_v) \cdot \cos \langle \omega_v, \omega_n \rangle \quad (1-7)$$

式中  $P^{q-1}$  表示在第  $q$  个碰撞点处光子入射的能量； $\omega_v$  是需要计算 BRDF 的虚拟方向； $V$  是可视因子，如果虚拟光子沿虚拟方向传播路径被场景元素遮挡则  $V = 0$ ，否则  $V = 1$ 。当发射虚拟光子时，也需要考虑场景的边缘效应（如图 1-9）。最终的 BRDF 计算公式为

$$f_{BRF_v} = \frac{\pi I_v^A}{\cos \theta_v \cdot P_{scene}} \quad (1-8)$$

式中  $I_v^A$  为虚拟方向  $v$  的辐射强度 ( $W \cdot sr^{-1}$ )； $\theta_v$  是虚拟方向的天顶角。利用虚拟方向计算方向反射率的优势在于计算 BRDF 的立体角是无限小的(Thompson and Goel, 1998)，也更容易与其他辐射传输模型进行比较。

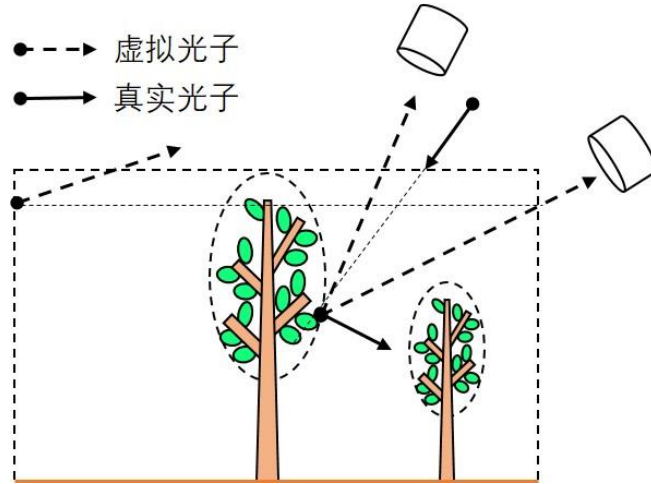


图 1-9 虚拟光子跟踪算法计算 BRDF

### 1.2.5 后向路径追踪模型

与前向光线追踪不同的是，后向光线追踪从传感器发射光线，并计算沿该光线传播到传感器的辐亮度值。光线的起始位置和方向由传感器的特征所决定，例如视场角、位置以

及观测方向等。后向光线追踪的主要原理是建立发射源和传感器之间的一种联系，从而计算发射源到传感器之间所传播的能量，这种联系称之为“路径（Path）”。因此在本研究中，称这种后向光线追踪为后向路径追踪（BPT: Backward Path Tracing）。后向路径追踪主要用于遥感影像的模拟，对于每一个像素都将发射多条光线，并将返回能量的平均值作为最终的像元值，因此每平米光线数（或者每像素光线数）对模拟影像的成像质量具有较大的影响，这也是后向路径跟踪算法中比较重要的参数之一。

#### 1.2.5.1 一次散射的计算

当光线从传感器出发并找到与场景的第一个交点时，需要计算发射源（即太阳或者大气）经该点散射到传感器的能量，这一过程称之为一次散射。由于场景中可能同时存在太阳与大气，因此在计算一次散射过程中需要随机地选择发射源。如图 1-10 所示，当光线到达 $q$ 时，将根据一定的概率在太阳和大气这两个辐射源中随机选择一个，通常太阳和大气的选择概率（ $p_{dir}$ 和 $p_{diff}$ ）可以设置为 0.5，或者根据发射功率权重进行设定，即 $w_k =$

$\frac{L_k(\lambda)}{\sum_{k=1}^K L_k(\lambda)}$ 。根据选择的辐射源不同，其处理方式也有所不同：

- 如果随机选择了太阳，那么将从 $q$ 点出发沿太阳方向在光源上找到一点 $q'$ ，如果点 $q$ 与 $q'$ 之间有场景元素遮挡，那么此时太阳对 $q$ 点的辐射贡献为 0，如果未发生遮挡，则太阳的辐照度为 $E_{dir}/p_{direction}^{sun}$ ，式中的 $p_{direction}^{sun}$ 表示太阳产生该方向光线的概率，由于其方向是固定的，因此 $p_{direction}^{sun} = p_{dir} \cdot 1$ 。
- 如果随机选择了大气，将在 $q$ 点所在的上半球空间随机产生一个方向，并计算该光线与大气半球的交点 $q'$ 。如果 $q$ 与 $q'$ 之间无遮挡，则大气的辐亮度为 $I_{diff}/p_{direction}^{atm}$ ，式中的 $p_{direction}^{atm}$ 表示在 $q$ 处产生该方向光线的概率，其计算公式为 $p_{direction}^{atm} = p_{diff} \frac{\cos\theta_{atm}}{\pi}$ ， $\theta_{atm}$ 为光线与 $q$ 点平面法线之间的夹角。

无论是太阳入射还是大气辐射入射，都会经过 $q$ 点散射到传感器光线方向 $\omega_o$ 。由于在 LESS 中假设了物体为朗伯表面，因此对于 $q$ 点产生特定入射角度的光线的概率为 $p_{bsdf}^{atm} = \frac{\cos\theta_{atm}}{\pi}$ （或者 $p_{bsdf}^{sun} = \frac{\cos\theta_{sun}}{\pi}$ ），这相当于同时对辐射源以及 BSDF 进行了采样，这种采样方式常常被称之为多重重要性采样（Multiple importance sampling），将他们组合可以得到总概率(Veach, 1997)

$$p_{weight}^{sun} = \frac{p_{direction}^{sun^2}}{p_{direction}^{sun^2} + p_{bsdf}^{sun^2}}, p_{weight}^{atm} = \frac{p_{direction}^{atm^2}}{p_{direction}^{atm^2} + p_{bsdf}^{atm^2}} \quad (1-9)$$

最终太阳对于传感器的辐亮度贡献为

$$\begin{aligned}
L_{sun} &= \frac{E_{dir}}{p_{direction}^{sun}} \cdot p_{weight}^{sun} \cdot f(q, \omega_i, \omega_o) \cdot \cos \theta_{sun} \\
&= \frac{E_{dir} \cos \theta_{sun}}{p_{dir}} \cdot p_{weight}^{sun} \cdot f(q, \omega_i, \omega_o)
\end{aligned} \tag{1-10}$$

大气对传感器的辐亮度贡献为

$$\begin{aligned}
L_{atm} &= \frac{I_{diff}}{p_{direction}^{atm}} \cdot p_{weight}^{atm} \cdot f(q, \omega_i, \omega_o) \cdot \cos \theta_{atm} \\
&= \frac{\pi I_{diff}}{p_{diff}} \cdot p_{weight}^{atm} \cdot f(q, \omega_i, \omega_o)
\end{aligned} \tag{1-11}$$

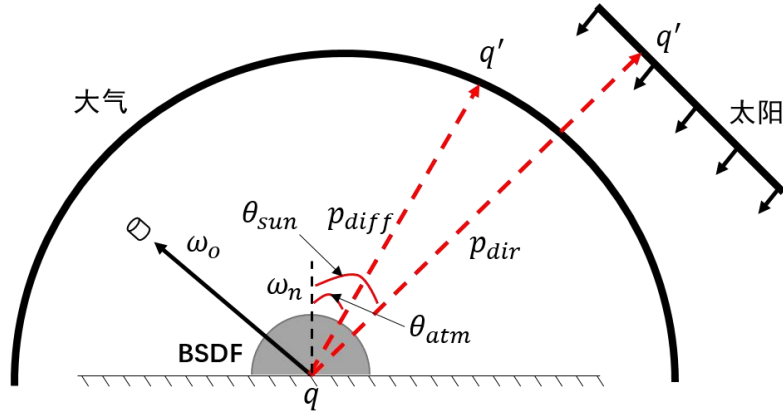


图 1-10 表面散射计算过程

#### 1.2.5.2 多次散射的计算

为了计算多次散射，将从 $q$ 点根据 BSDF 随机产生一条新的光线，如果该光线与场景相交于 $q_1$ （图 1-11），那么在 $q_1$ 点也会计算太阳以及大气的直接一次散射，从 $q_1$ 散射的辐亮度沿着 $\omega_1$ 方向将到达 $q$ ，最终经过 $q$ 返回到传感器，这一部分能量就是二次散射项。该过程可以一直重复直到光线离开场景，或者达到用户设定的最大的散射次数。为了避免因为人为的截断造成能量的损失，在此也使用了与前向光子跟踪模型中相同的基于概率的截断机制，即以一定的概率 $p$ 终止光线的传播，对于未被终止的光线乘以概率 $\frac{1}{1-p}$ 。

当模拟水平无限场景时，对于传感器发射的光线（Sensor ray）以及计算辐射源贡献时的光照光线（Illumination ray），都需要考虑边缘效应。当他们从场景侧边缘离开场景时，需要从另一个对称侧边重新进入场景（镜像光线），以保持在场空间内的能量守恒。

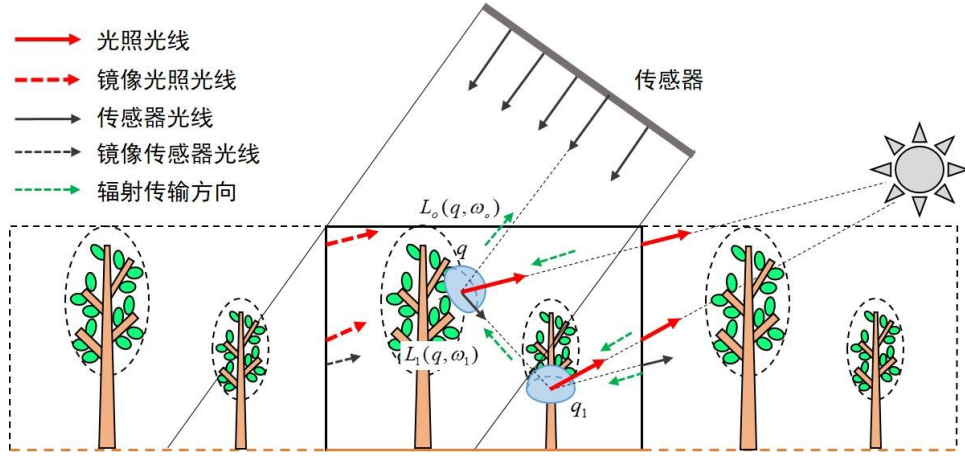


图 1-11 后向路径跟踪算法多次散射计算示意图

### 1.2.6 热红外影像模拟

在热红外波段，物体本身也会发射一定的能量，其发射的能量由温度以及发射率 $\epsilon_\lambda$ （Emissivity）所决定。对于黑体来说，可以用普朗克定律（Planck's law）来描述物体发射辐亮度  $L$  与温度  $T$  以及波长 $\lambda$ 关系

$$L(T, \lambda) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1} \quad (1-12)$$

式中 $k_B = 1.38 \times 10^{-23}$  J/K 为玻尔兹曼常数（Boltzmann constant）； $h = 6.63 \times 10^{-34}$  J·s 为普朗克常数（Planck constant）； $c = 2.99 \times 10^8$  m/s 为光速； $\lambda$ 为波长（单位：m）。最终得到的辐亮度单位是  $\text{W} \cdot \text{sr}^{-1} \cdot \text{m}^{-2}$ 。对于非黑体而言，其发射辐亮度等于同温度下的黑体辐亮度乘以发射率，即 $\epsilon_\lambda \cdot L(T, \lambda)$ 。

在  $8 \sim 14 \mu\text{m}$  波段，太阳辐射与地物热辐射相比明显微弱得多，因此在模拟过程中可以忽略太阳对热红外辐射的贡献。但是太阳入射光形成的光照和阴影分布会极大地影响物体表面的温度分布，进而影响整个场景的热红外辐射特性，因此在模拟过程中需要首先确定阴影以及光照的分布。一般而言，在植被遥感中，可以将场景元素划分为四类：光照土壤，阴影土壤，光照叶片以及阴影叶片。传统的正向模型（如 DART）在模拟热红外辐射时首先需要模拟并保存这四个分量，并赋予相应的组分温度，在此基础上再继续模拟热红外的模拟。而在 LESS 中，采用了实时计算四分量的方法，从而避免了储存四分量信息，减少了内存使用，尤其是对于有大量三角面元的场景，该方法具有显著的优势。

图 1-12 展示了利用后向路径追踪模拟热红外影像的过程。当光线从传感器出发找到场景中的第一个交点 $q$ 时，首先从以 $q$ 点为起点，沿着太阳入射反方向发射一条遮挡测试光线（Occlusion ray），如果 $q$ 没有被其他场景元素所遮挡，则利用光照元素（光照叶片或者光照土壤）的组分温度根据普朗克定律以及发射率计算 $q$ 点的发射辐亮度，反之，则利用阴

影元素（阴影叶片或者阴影土壤）计算发射辐亮度。 $q$ 点的发射辐亮度就是公式(1-1)中的发射项 $L_e(q, \omega_o)$ ，除了发射项以外，周围场景元素发射能量以及天空入射能量经过 $q$ 点反射也会进入传感器。为了考虑这一部分能量，将场景元素都作为发射源来处理，每次随机选择一个并计算其对 $q$ 点辐射量的贡献，该计算过程与1.2.5节中计算方法一致，重复此过程可以计算多次散射的影响。

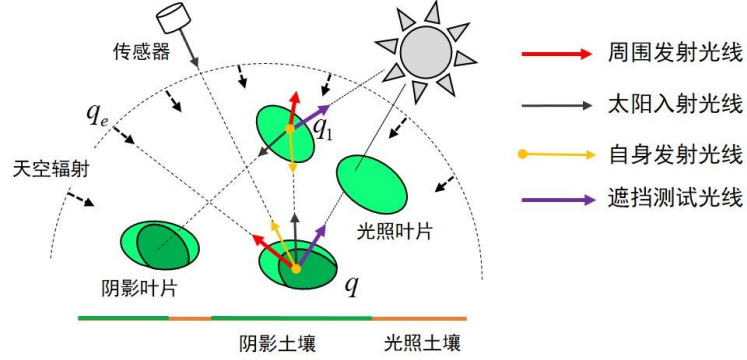


图 1-12 利用后向路径跟踪算法模拟热红外信号过程示意图

### 1.2.7 植被冠层分层 FPAR 模拟

在前向光子跟踪过程中，根据光子碰撞点以及场景元素局部光学属性，可以计算冠层的分层分组分 FPAR（Fraction of Photosynthetically Active Radiation）。如图 1-13 所示，当光子与场景中元素（如叶片）发生碰撞时，可以根据叶片的吸收率来计算叶片吸收能量的大小。在朗伯假设下，叶片的吸收率可以表示为  $1 - \rho_\lambda - \tau_\lambda$ ，其中  $\tau_\lambda$  为透过率， $\rho_\lambda$  为反射率，光子在经过  $Q - 1$  次散射后剩余能量为

$$P^{Q-1}(\lambda) = P^0(\lambda) \cdot \prod_{q=1}^{Q-1} (\rho_{q,\lambda} + \tau_{q,\lambda}) \quad (1-13)$$

上式在实质上等价于式(1-3)，不同的是，前者从反射透射角度来描述散射过程，而后者从能量平衡的角度来描述散射过程。

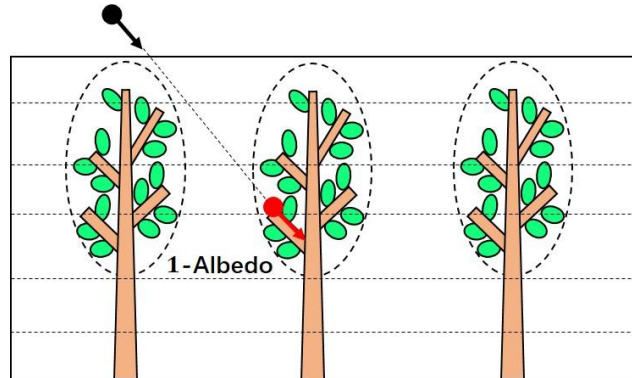


图 1-13 冠层分层分组分 FPAR 模拟

为计算 FPAR，在每一个碰撞点，可以记录光子的吸收能量为  $(1 - \rho_\lambda - \tau_\lambda) \cdot P^{Q-1}(\lambda)$ 。此外，根据光子碰撞点的位置以及碰撞元素的类别，可以分别记录得到冠层的分层分组吸收能量。如果对 380 nm ~ 710 nm 之间的 PAR 进行波段积分再除于入射辐射可得到最终 FPAR 的值。

### 1.2.8 复杂地形区上下行辐射模拟

地形的起伏对于局部太阳下行辐射具有较大的影响(Yan et al., 2016)，该影响主要体现在起伏的地形造成坡面与太阳入射方向夹角的改变，周围地形对于太阳和天空光的遮挡以及坡面之间的相互散射(Wen et al., 2018)。为了定量刻画这种地形对于上下行辐射的影响，目前已经建立了多种多样的与地形有关的模型(Dozier and Frew, 1990; Li et al., 1999; Wang et al., 2005)，在所有的这些模型中，基于蒙特卡罗以及光线跟踪的方法具有较高的精度，而且可以模拟任意复杂地形下的下行辐射，LESS 是这一类模型中的代表之一。与其他模型相比，LESS 不仅能够模拟任意复杂地形，而且在地形之上可以添加任何复杂的植被冠层，可以定量考虑植被冠层对于上下行辐射的影响。

图 1-14 展示了 LESS 中上下行辐射模拟的基本原理。首先根据模拟需要将场景划分成基本的网格，每个网格为一个像素 (Pixel)，其目的就是要模拟每个网格中的上行和下行辐射。假设一个光子从  $P_1$  点反射并与场景相交于  $P_2$  点，则需要根据  $P_1$  和  $P_2$  的位置关系来确定上下行辐射的计算：

- 若  $P_1$  与  $P_2$  属于不同的像元 (Pixel4 和 Pixel2)，则对出射像元 (Pixel4) 记录上行辐射，对入射像元 (Pixel2) 记录下行辐射；
- 如果  $P_1$  与  $P_2$  属于同一个像元，则不记录上行辐射也不记录下行辐射，即该散射属于像元内部散射；
- 如果只存在  $P_1$  交点，即光子从  $P_1$  出发后离开了场景，则只记录出射像元 (Pixel4) 的上行辐射

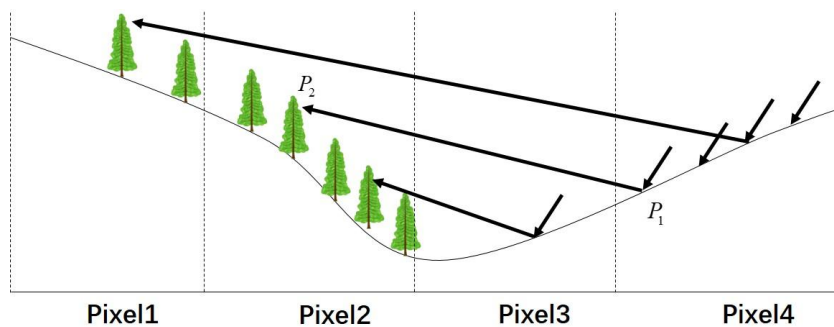


图 1-14 利用前向光子跟踪算法模拟复杂地形区上下行辐射

在光子与地形的碰撞点，需要考虑地物的反射率，然而现实中往往每个像元的反射率是不同的，例如有冰雪覆盖的地面反射率明显高于无冰雪覆盖的部分。为了能够模拟这种空间异质条件下的上下行辐射，LESS 采用了基于纹理的反射率输入方法，即可以在 LESS 中输入一幅反射率影像，每个像素保存了对应网格的反射率信息。图 1-15 展示了在空间均匀反射率以及空间变化反射率影响下，地形之间多次散射的模拟结果。从图中可以看出，均匀反射率情况下（图 1-15c），多次散射较高的区域主要在山谷地区，因为山谷两侧山坡距离较近，多次散射较强。而当把真实反照率影像作为输入（图 1-15a）来模拟多次散射，多次散射较强的区域主要位于冰雪覆盖区域（图 1-15c）。

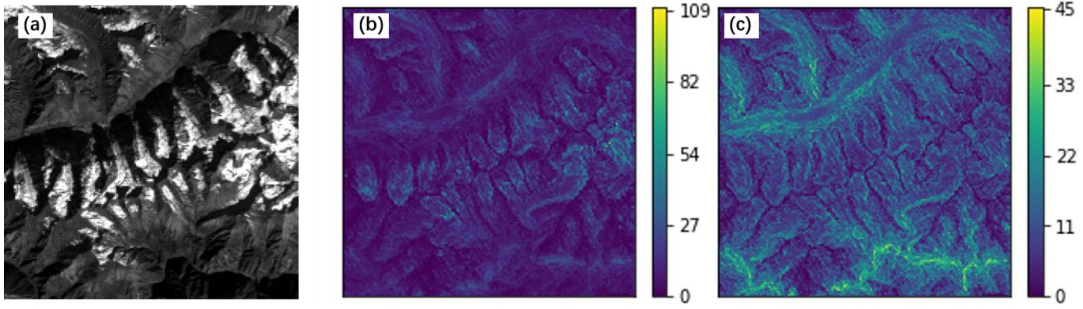


图 1-15 空间变化反射率及均匀反射率下地形多次散射下行辐射模拟

(a) 真实反照率数据作为输入数据；(b) 空间变化反射率情况下多次散射模拟结果；(c) 均一反射率情况下多次散射模拟结果

### 1.2.9 晴空大气辐射传输模拟

在 LESS 模拟中，所有的辐射源（太阳和大气辐射）都是直接输入的参数，这些参数可以来自实际测量的太阳下行辐射，也可以来自于其他大气模型的输出结果。为了能够直接使用大气参数进行模拟，LESS 模型中增加了大气辐射传输模拟模型。

#### 1.2.9.1 水平分层大气模型

由于大气具有垂直分层结构，而且每一层内部的性质相对均一，因此在 LESS 中，大气被抽象为具有水平层的结构。如图 1-16 所示，每一水平层的光学属性由六个参数进行描述：气体分子消光系数 $\beta_{gas}^{ext}$ ，气体分子单次散射反照率 $\omega_{gas}$ ，气体分子散射相函数 $P_{gas}$ ，气溶胶消光系数 $\beta_{aerosol}^{ext}$ ，气溶胶单次散射反照率 $\omega_{aerosol}$ 以及气溶胶散射相函数 $P_{aerosol}$ 。当用 LESS 进行大气辐射传输模拟时，这些参数将进行加权平均以得到每层大气的平均光学属性

$$\beta_{atm}^{ext} = \beta_{gas}^{ext} + \beta_{aerosol}^{ext} \quad (1-14)$$

$$\omega_{atm} = \frac{\beta_{gas}^{ext} \cdot \omega_{gas} + \beta_{aerosol}^{ext} \cdot \omega_{aerosol}}{\beta_{gas}^{ext} + \beta_{aerosol}^{ext}} \quad (1-15)$$

$$P_{atm}^{ext} = \frac{\beta_{gas}^{ext} \cdot \omega_{gas} \cdot P_{gas} + \beta_{aerosol}^{ext} \cdot \omega_{aerosol} \cdot P_{aerosol}}{\beta_{gas}^{ext} \cdot \omega_{gas} + \beta_{aerosol}^{ext} \cdot \omega_{aerosol}} \quad (1-16)$$

对于  $P_{gas}$ ，一般采用瑞利散射相函数进行描述；对于  $P_{aerosol}$ ，一般采用 Henyey-Greenstein 散射相函数进行描述 (Henyey and Greenstein, 1941)，即

$$P_{gas}(\cos \theta) = \frac{3}{16\pi} (1 + \cos^2 \theta) \quad (1-17)$$

$$P_{aerosol}(\cos \theta) = \frac{1}{4\pi} \frac{1 - g^2}{(1 + g^2 + 2g \cos \theta)^{3/2}} \quad (1-18)$$

式中  $\theta$  为散射相位角，即散射方向与入射方向的夹角； $g$  为非对称参数，当  $g > 0$  时，表示前向散射占优，当  $g < 0$  时，表示后向散射占优。

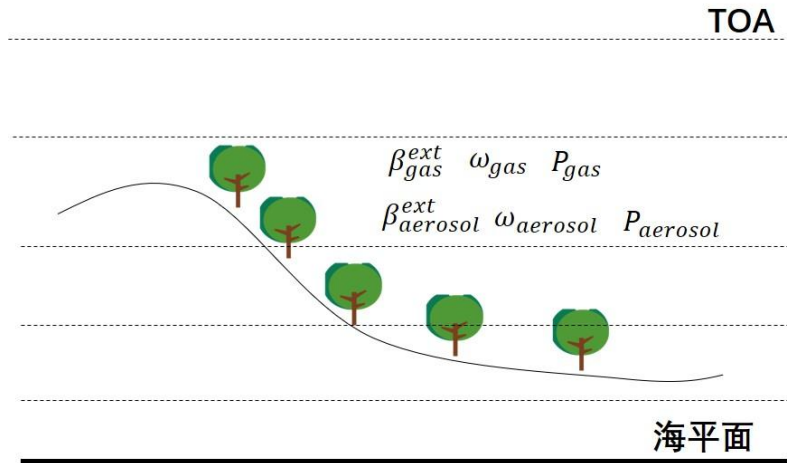


图 1-16 大气分层结构

### 1.2.9.2 气体介质中辐射传输计算

#### A. 辐射传输方程一般形式

如图 1-17 所示，假设在气体介质中位于  $t = 0$  处的入射辐射为  $L(0, \omega)$ ，传播到  $t = d$  时变为  $L(d, \omega)$ 。 $L(d, \omega)$  由三部分能量组成：直接透射  $L_{dir}(d, \omega)$ 、从  $t = 0$  到  $t = d$  过程中气体本身辐射部分  $L_e(d, \omega)$  以及周围其他介质沿  $\omega$  方向的散射贡献部分  $L_s(d, \omega)$ 。假设在传播路径上的  $t$  处，气体的消光系数和单次散射反照率分别为  $\beta(t)$  和  $\omega(t)$ ，以上三部分计算方式如下：

- **直接透射部分：**直接透射部分即入射辐射经过气体在传播路径上的消光作用后剩余部分的能量，其结果等于入射辐射乘以传播路径上的透过率，即

$$L_{dir}(d, \omega) = L(0, \omega) e^{\int_0^d -\beta(t) dt} \quad (1-19)$$

- **自身发射部分：**在入射辐射 $L(0, \omega)$ 从 $t = 0$ 到 $t = d$ 的传播过程中，气体本身可能发射一部分能量 $L_e(t, \omega)$ ，发射的能量又将从发射点 $t$ 传播到 $t = d$ 处，此过程也需要考虑气体的消光作用，即

$$L_e(d, \omega) = \int_0^d \beta(t)[1 - \omega(t)]L_e(t, \omega) e^{\int_t^d -\beta(t') dt'} dt \quad (1-20)$$

- **周围散射贡献部分：**该部分主要由周围气体经过散射后进入 $\omega$ 方向而引起的辐射增加的过程，其大小与散射系数以及散射相函数有关，即

$$L_s(d, \omega) = \int_0^d \beta(t)\omega(t) \int_{S^2} P(t, \omega, \omega') L(t, \omega') d\omega' e^{\int_t^d -\beta(t') dt'} dt \quad (1-21)$$

式中 $P(t, \omega, \omega')$ 是 $t$ 点处的散射相函数； $L(t, \omega')$ 为 $t$ 点附近周围气体沿 $\omega'$ 方向的入射辐射。

将以上三项合并可得：

$$L(d, \omega) = L(0, \omega)T(d) + \int_0^d T(d-t)[\sigma_a(t)L_e(t, \omega) + \sigma_s(t)L_s(t, \omega)]dt \quad (1-22)$$

其中 $\sigma_a(t) = \beta(t)[1 - \omega(t)]$ 表示气体吸收系数； $\sigma_s(t) = \beta(t)\omega(t)$ 表示气体散射系数； $T(t) = e^{\int_0^t -\beta(t') dt'}$ 表示从 $t = 0$ 到 $t$ 处的透过率，当气体内部为均匀介质时，消光系数 $\beta$ 为常数，因此可得 $T(t) = e^{-\beta t}$ 。

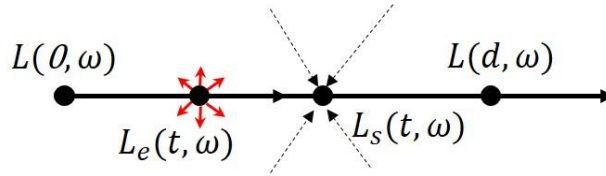


图 1-17 气体介质中辐射传输计算示意图

## B. 均匀介质中辐射传输方程求解

要求解辐射传输方程，就要计算式(1-22)中的积分。根据重要性采样（Importance sampling）理论（详见[错误!未找到引用源。](#)），要以较低的误差估算积分，就要尽可能地选择一个与积分项呈常数比例的概率密度函数。由于在均匀介质中 $T(t)$ 表达式已知，且存在于积分项中，因此可以取重要性概率密度函数（Importance density） $p(t) = cT(t)$ ，其中 $c$ 为常数，使得

$$L(d, \omega) = L(0, \omega)T(d) + \int_0^d \frac{T(d-t)}{p(t)} [\sigma_a(t)L_e(t, \omega) + \sigma_s(t)L_s(t, \omega)]p(t)dt \quad (1-23)$$

由于 $p(t)$ 满足 $\int_0^{+\infty} p(t)dt = 1$ ，因此

$$c = \frac{1}{\int_0^{+\infty} T(t)dt} = \frac{1}{\int_0^{+\infty} e^{-\beta t} dt} = \beta \quad (1-24)$$

因此，重要性概率密度函数 $p(t) = \beta e^{-\beta t}$ ，该式也表示光线在 $t$ 处产生大气散射事件的概率大小，即 $p_{suc} = \beta e^{-\beta t}$ ，如果光在 $t = 0$  到  $t = d$ 之间不发生大气散射，其概率为

$$p_{fail} = 1 - \int_0^d \beta e^{-\beta t} dt = e^{-\beta d} \quad (1-25)$$

要根据 $p(t)$ 随机采样得到 $t$ ，即光线（或者光子）在介质中随机前进一步的步长以得到下一次发生散射的散射点，可以通过逆变换采样（Inverse transform sampling）的方法对 $p(t)$ 进行随机采样(Morlet, 1983)。该采样方法首先计算 $p(t)$ 的累积分布函数(CDF: Cumulative Distribution Function)，即

$$CDF(t) = \int_0^t \beta e^{-\beta t'} dt' = 1 - e^{-\beta t} \quad (1-26)$$

由此可以得到随机采样的 $t$ 为

$$t = -\frac{\ln(1 - \xi)}{\beta} \quad (1-27)$$

式中 $\xi$ 为均匀分布的随便变量，且 $\xi \in [0,1]$ 。

### 1.2.9.3 分层大气中辐射传输计算

由于后向路径追踪比前向光子追踪算法在模拟影像方面具有较快的收敛速度，因此，在大气辐射传输计算中，本研究以后向光线追踪为主算法进行实现，从而可以模拟大气顶层和大气中的遥感影像，通过鱼眼相机等也可以模拟天空散射光的分布。

图 1-18 展示了后向光线跟踪模拟大气辐射传输的过程。光线从传感器出发进入大气后，会发生散射作用，散射点由随机采样得到的自由路径长度（Free path）所决定。由于场景中除了大气可能存在其他元素（比如地形），因此辐亮度的计算过程需要根据光线是否与场景其他元素发生碰撞的情况来决定：

- **光线在采样路径上不发生碰撞**，即发生大气散射（如图 1-18 中的 AB 点）。在这种情况下，需要根据大气的散射相函数 $P_{atm}^{ext}$ 以及光线在此处成功产生一次散射作用的概率 $p_{suc}$ 来确定辐亮度的大小。假设光线从 A 点出发在 B 点产生散射作用，那么太阳直接辐射经过 B 点散射后对 A 点的辐亮度贡献为

$$L_{B \rightarrow A} = T_{AB} \cdot \beta_B \omega_B \cdot P_{atm}^{ext} \cdot E_{dir} T_{B+\infty} / p_{suc}^{AB} \quad ($$

1-28)

式中 $T_{AB}$ 表示点 A 与点 B 之间的大气透过率； $E_{dir}$ 为太阳辐照度； $T_{B+\infty}$ 表示点 B 沿太阳反方向到大气层顶的大气透过率； $p_{suc}^{AB}$ 为光线从 A 点离开并在 B 点成功产生大气散射作用的概率，此概率与大气消光系数大小有关。

- **光线在采样路径上发生了碰撞**，即发生表面散射（如图 1-18 中的 C 点）。在这种情况下，应当根据交点 C 处的 BSDF 以及在此处未成功产生大气散射作用的概率  $p_{fail}$  来确定辐亮度大小。假设光线从 B 点出发在 C 点与场景元素发生表面散射，那么太阳直接辐射经过 C 点散射后对 B 点的辐亮度贡献为

$$L_{C \rightarrow B} = T_{BC} \cdot f(C, \omega_i, \omega_o) \cdot E_{dir} T_{C+\infty} / p_{fail}^{BC} \quad (1-29)$$

式中 $f(C, \omega_i, \omega_o)$ 为 C 点处的 BSDF 函数； $p_{fail}^{BC}$ 为光线从 B 点离开并在 C 点未产生大气散射作用的概率。

无论是光线与大气发生作用后还是与场景元素发生作用后，都会根据散射相函数或者 BSDF 产生一个新的光线，并重复上述过程，以计算多次散射的贡献。该过程与 1.2.5 节描述的过程类似，只是增加了大气散射部分。

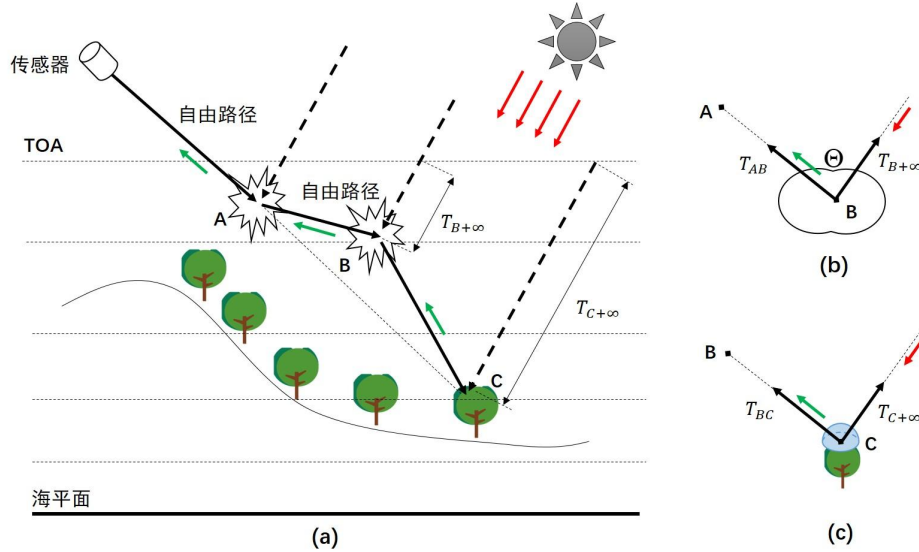


图 1-18 后向光线跟踪模拟大气辐射传输过程

由式(1-28)和(1-29)可知，要计算传感器接收的辐亮度，最重要的就是要计算发生或者不发生大气散射作用的概率 $p_{suc}$ 和 $p_{fail}$ ，以及随机采样的自由路径长度。然而对于大气来说，虽然在某一层内光学属性呈均匀分布，但层与层之间的光学属性却有较大差异，所以无法直接使用均匀介质中的采样方法来计算自由路径长度。为此，本研究采用了直接采样光学厚度而不是自由路径长度的方式对辐射传输过程进行求解，即 $\tau_x = -\ln(1 - \xi)$ 。由于大气不是完全均匀的气体介质，故不能直接通过 $\tau_x/\beta$ 得到自由路径的长度，因此对于每一个随

机采样的光学厚度 $\tau_x$ ，需要在传播路径上找到一个点，使得从起点到该点之间的累积光学厚度等于 $\tau_x$ 。如图 1-19 所示，假设光线从 $A_1$ 点传播到位于另一层的 $A_n$ 点（由 $\tau_x$ 确定），则 $AA_n$ 之间的透过率可表示为 $T_{AA_n} = e^{-\tau_x}$ 。由于光线在 $A_n$ 点处发生了大气散射，那么光线在其他层未发生大气散射作用，因此 $p_{suc}^{A_1A_n}$ 可表达为

$$p_{suc}^{A_1A_n} = p_{suc}^{A_{n-1}A_n} \cdot \prod_{i=2}^{n-1} p_{fail}^{A_{i-1}A_i} = \beta_{n-1} \cdot e^{-\sum_{i=1}^{n-1} \beta_i d_i} \quad (1-30)$$

$p_{fail}^{A_1A_n}$ 可表达为

$$p_{fail}^{A_1A_n} = \prod_{i=2}^n p_{fail}^{A_{i-1}A_i} = e^{-\sum_{i=1}^{n-1} \beta_i d_i} \quad (1-31)$$

式中 $\beta_i$ 表示光线起始层开始的第 $i$ 层的消光系数； $d_i$ 表示光线在第 $i$ 层穿透的路径长度。

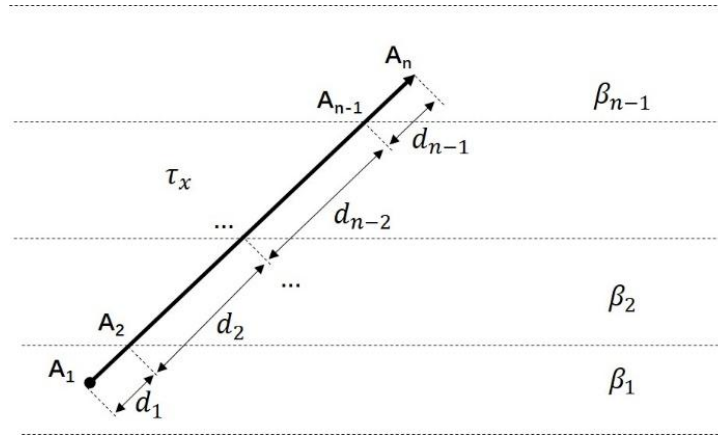


图 1-19 分层大气中自由路径采样方法

## 1.2.10 土壤非朗伯反射模型

### 1.2.10.1 RPV 模型（Rahman-Pinty-Verstraete）

RPV 是一个能够表面非朗伯反射的参数化模型。RPV 计算公式中包含了两大部分：一是表示总体反射强度的量 $\rho_0$ ，而是表示反射的方向分布的量。RPV 的计算公式如下：

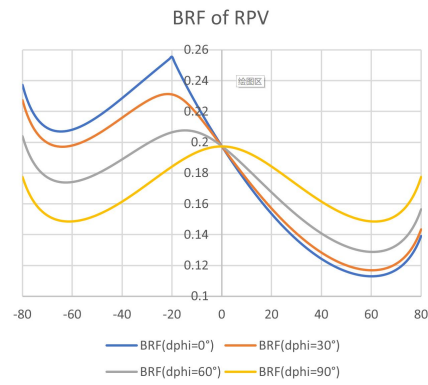
$$\rho(\mu_i, \mu_r, d\phi, \rho_0, \Theta, k, \rho_c) = \rho_0 \cdot M(k) \cdot F_{HG}(\Theta) \cdot H(\rho_c) =$$

$$\rho_0 = \frac{(\mu_i \mu_r)^{k-1} (1 - \Theta^2)}{(\mu_i + \mu_r)^{1-k} (1 + \Theta^2 + 2\Theta \cos g)^{1.5} (1 + \frac{1 - \rho_c}{1 + G})}$$

式中 $\rho(\mu_i, \mu_r, d\phi, \rho_0, \Theta, k, \rho_c)$ 为观测到的 BRF,  $\mu_i = \cos\theta_i, \mu_r = \cos\theta_r$ ,  $\theta_i$ 和 $\theta_r$ 分别表示观测天顶角和观测天顶角。  $\cos g = \cos\theta_i \cos\theta_r + \sin\theta_i \sin\theta_r \cos d\phi$ ,  $G = (\tan^2\theta_i + \tan^2\theta_r - 2\tan\theta_i \tan\theta_r \cos d\phi)$ ,  $d\phi$ 为观测和入射方位角夹角，两种重合时为 $0^\circ$ 。

案例:  $\rho_0 = 0.075, k = 0.55, \Theta = -0.25, \rho_c = 0.075$

$\theta_i = 20^\circ, \theta_r = -80^\circ \sim 80^\circ, d\phi = 0^\circ \sim 90^\circ$



## 第 2 章 LESS 模型实践练习

本章主要从实践的角度介绍 LESS 模型的使用方法，主要内容包含多光谱影像模拟、反射率模拟、热红外影像模拟以及 FPAR 等模拟。

### 2.1 操作案例

#### 2.1.1 多光谱影像模拟

- 打开 LESS 主程序（如图 2-1），然后新建一个模拟工程【File】->【New Simulation】，新建一个目录 Ex01，并选择该目录。若工程新建成功，将会在【Console】区域看到“Succeed: ‘save path’”信息。

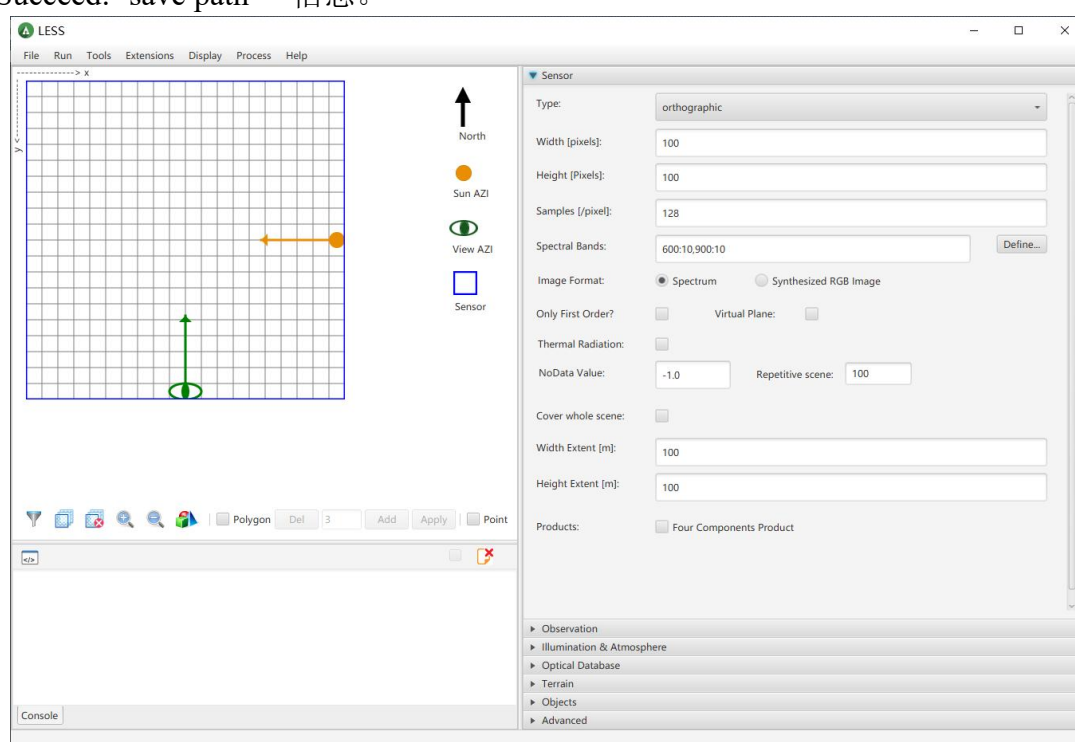


图 2-1 LESS 主程序窗口

- 在【sensor】窗口中定义传感器相关的参数，主要参数及说明如下（未提及参数保持默认即可）：

| 参数                | 值            | 说明                                                                                  |
|-------------------|--------------|-------------------------------------------------------------------------------------|
| Type              | Orthographic | 传感器类型，Orthographic 表示平行投影相机                                                         |
| Width [pixels]    | 500          | 最终模拟影像的宽度，以像素为单位                                                                    |
| Height [pixels]   | 500          | 最终模拟影像的高度，以像素为单位                                                                    |
| Samples [/pixel]  | 32           | 每个像素发射光线的条数，一般跟场景复杂程度有关，常用的值有 32，64，128 等，值越大，结果越精确，但是计算时间也越长。(空间分辨率为 1m 时，可设置为 64) |
| Spectral Bands    | 600:2,900:2  | 表示模拟波段为 600nm 和 900nm，波段宽度为 2nm；                                                    |
| Image Format      | Spectrum     | 最终输出的影像的格式，Spectrum 表示输出多波段格式，保存格式为 ENVI 标准格式；RGB image 输出 png 格式的图像，为程序自动合成的可见光照片； |
| Only First Order  | 不勾选          | 如果勾选，则表示只模拟单次散射，而忽略多次散射项；                                                           |
| NoData Value      | -1.0         | 如传感器视场范围超过场景大小，将超出的区域的像素值为该设定值；                                                     |
| Repetitive scene  | 100          | 水平扩展场景的次数，以模拟场景在水平方向无限延申的情形；                                                        |
| Width Extent [m]  | 100          | 平行投影传感器的视场大小（其他类型传感器无此参数）                                                           |
| Height Extent [m] | 100          | 平行投影传感器的视场大小                                                                        |

#### ■ 设置观测几何【Observation】

对于平行投影相机而言，主要需要设置的参数为观测天顶角【View Zenith】和观测方位角【View Azimuth】。方位角以正北方向为 0°，沿顺时针方向增加，正南为 180°。在

LESS 中，平行投影相机的视场范围实际上是一个立方体，场景被包含在立方体中，最终的成像平面为立方体的底面，成像平面与场景中心的距离由【Sensor Height】决定（如图 2-2 所示）。

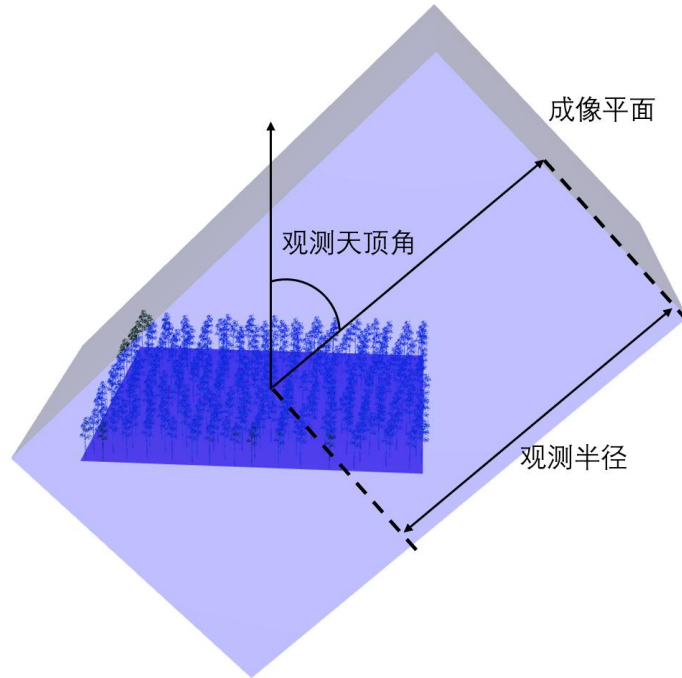


图 2-2 LESS 平行投影相机视场范围

#### ■ 设置光照参数【Illumination & Atmosphere】

| 参数             | 值            | 说明                                 |
|----------------|--------------|------------------------------------|
| Sun Zenith     | 45           | 太阳天顶角                              |
| Sun Azimuth    | 90           | 太阳方位角                              |
| Sky-Type       | SKY_TO_TOTAL | 天空散射光类型：SKY_TO_TOTAL 表示天空光占总入射的比例； |
| Sky-Percentage | 0,0          | 表示每个波段天空光入射的比例                     |

注：若勾选【Input solar spectrum manually】，则可以手工输入太阳和天空的入射光谱

#### ■ 设置光学属性【Optical Database】

在此练习中，保持默认值即可。事实上，Optical Database 定义了一系列的光谱用于之后设置地面及场景元素的光学属性。在 LESS 中，光谱包含了正面反射率、反面反射率以

及透过率。“birch\_branch”，“dark\_soil\_mollisol”，“birch\_leaf\_green”为 LESS 默认的光谱，请不要进行改动。

#### ■ 设置地形【Terrain】：

| 参数               | 值                  | 说明                                                        |
|------------------|--------------------|-----------------------------------------------------------|
| Type             | PLANE              | PLANE: 平面地形; RASTER: 用 ENVI 栅格影像作为地形; MESH: 用 OBJ 文件作为地形; |
| XSize            | 100                | 场景大小: 宽度, 单位为 m                                           |
| Ysize            | 100                | 场景大小: 高度, 单位为 m                                           |
| BRDF Type        | Lambertian         | 地面为朗伯反射                                                   |
| Optical Property | dark_soil_mollisol | 该列表列出了所有在【Optical Database】中定义的光谱曲线;                      |

#### ■ 设置场景元素【Objects】

点击按钮【Define Object】，将弹出下图所示窗口，在最左侧输入 tree，点击【add】，添加一个名称为“tree”的 Object。选择该 Object，点击中间窗口中【Import OBJ】选择一个 OBJ 文件。在 LESS 的安装目录下附带了一些 OBJ 文件，例“...\app\Database\3D\_Objects\Trees\RAMI”中的“BEPE\_FROM\_HET07\_JPS\_SUM.obj”文件。导入后，将显示该 OBJ 文件中的所有分组（即 OBJ 文件中的“g”定义的分组），分别选择每一个分组，为其设置相应的【Optical Property】。“tree\_leaves.obj”光谱属性设置为“birch\_leaf\_green”，“tree\_stem\_branch.obj”光谱属性设置为“birch\_branch”。

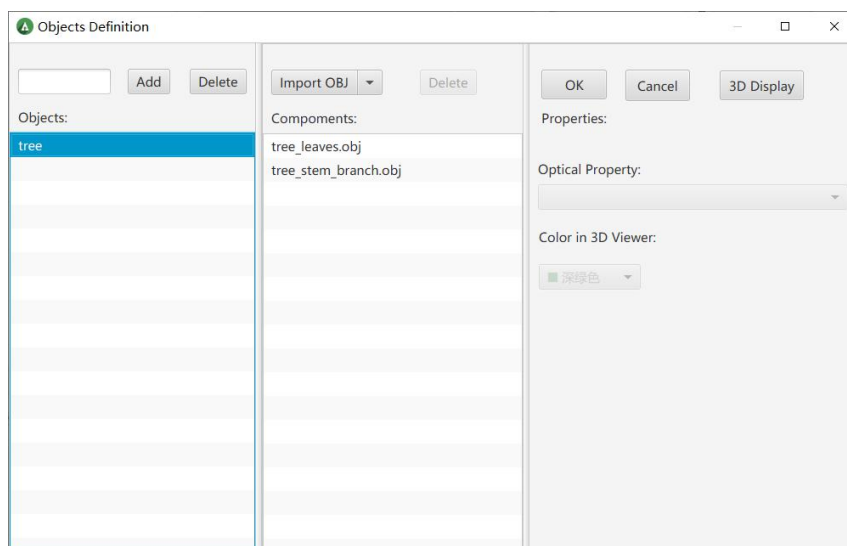


图 2-3 定义场景 Object

### ■ 定义场景元素的位置

在主窗口中选择需要分配位置的 Object，例如 tree，点击【Random】按钮，在弹出窗口中，定义任何两棵树之间的最小距离（避免两棵树完全重叠），点击【OK】后，tree 将随机分配到场景的不同位置。

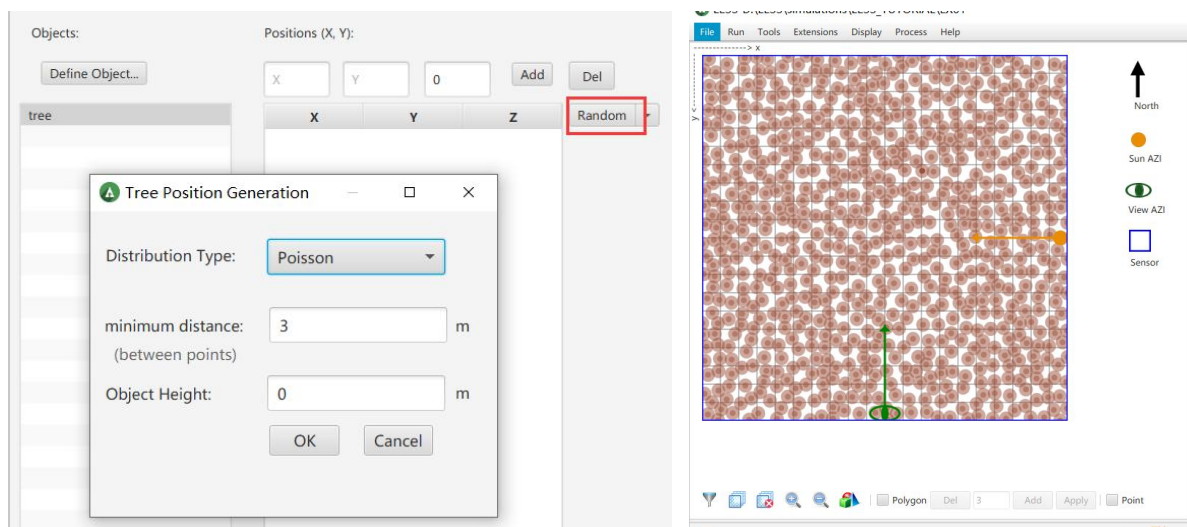


图 2-4 定义场景 Object 的位置

### ■ 三维显示

点击主界面上的按钮【】，将显示场景的三维结构，以检查所建场景是否符合预期目标。

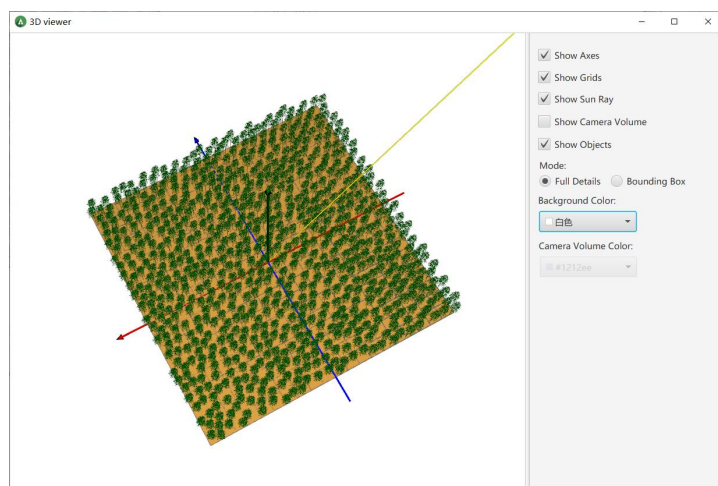


图 2-5 场景三维显示

### ■ 影像模拟及结果查看

点击菜单【Run】->【Run All】，开始模拟。模拟结束后，点击菜单【Tools】->【Open Results Folder】，打开模拟结果目录，文件“spectral\_VZ=0\_VA=180”为模拟得到的辐亮度影像，用 ENVI 可以查看该多光谱文件。

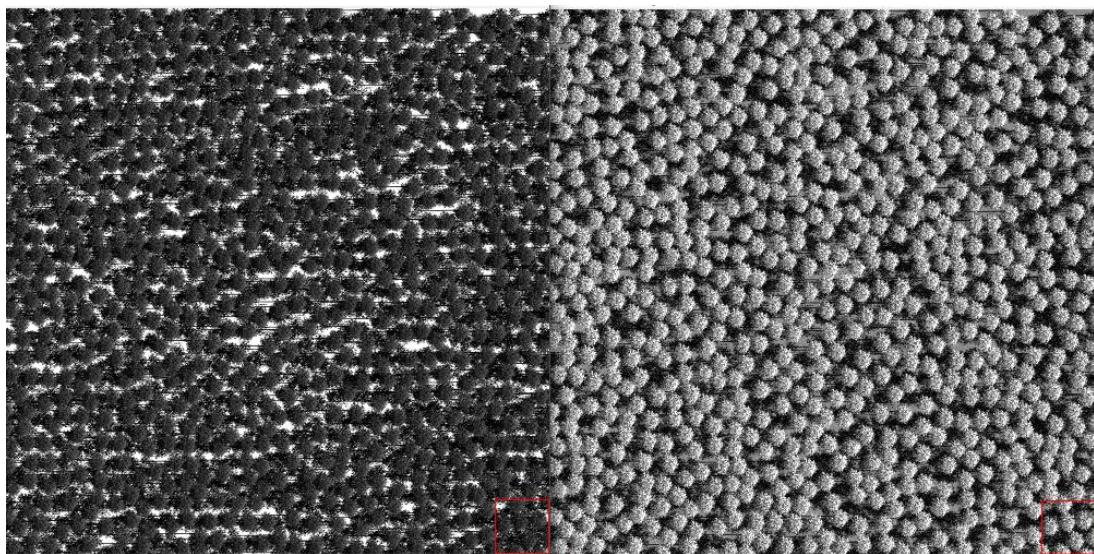


图 2-6 模拟影像：红光波段（左）和近红外波段（右）

### ■ 生成反射率文件

点击菜单【Process】->【BRF Processing】，可以将辐亮度影像转换成反射率影像，结果为“spectral\_VZ=0\_VA=180\_BRF”。

## 2.1.2 鱼眼相机模拟

### ■ 传感器参数设置

在【Type】中选择“CircularFisheye”，此时出现一组新的参数用于定义鱼眼相机的特征。【FOV】表示鱼眼相机的视场角，【Fisheye Projection】表示鱼眼相机的投影模式。LESS 中内置了四种投影模式：

- Equisoild：等面积投影
- Orthographic：正射投影
- Equidistant：等距离投影
- Stereographic：球极平面投影

常用的投影模式为等面积投影（即等立体角投影）

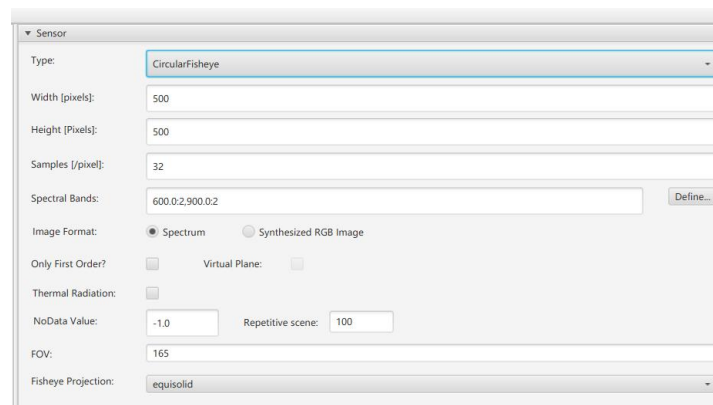


图 2-7 鱼眼相机参数设置

在选择鱼眼相机后，相机的观测方式亦发生了改变，此时需要重新设置【Observation】中的相关参数。【Camera Position】定义了相机的位置 A，【Target】定义了相机的观测点 B，这两则组成的向量 $\overrightarrow{AB}$ 实际上就定义了相机的观测方向。如果勾选【Relative Height】则表示上述的 AB 点的高度 Z 为相对于地面的高度（当有地形存在时）。在此练习中，A 和 B 可以设置为（50，50，0），（50，50，1），表示相机在场景中心并朝天顶方向观测。

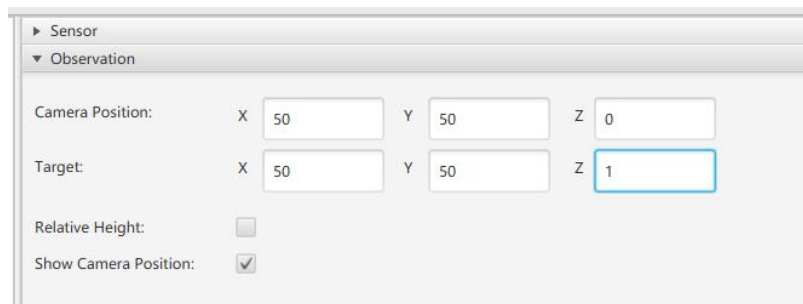


图 2-8 鱼眼相机观测参数设置

将【NoData Value】设置为 0 进行模拟，得到的 900nm 的鱼眼相机照片如下如图 2-9

所示（文件“spectral\_ox=50\_00\_oy=50\_00\_oz=0\_00\_tx=50\_00\_ty=50\_00\_tz=1\_00”）。

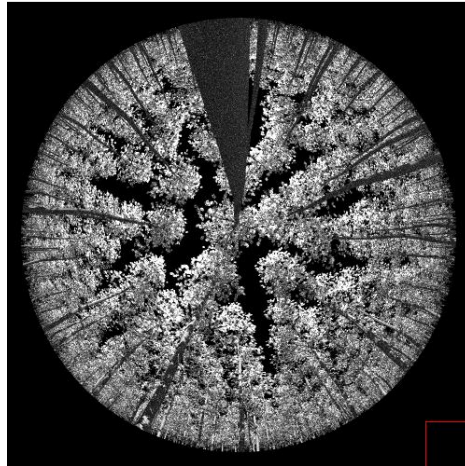


图 2-9 鱼眼相机照片示例

### 2.1.3 多角度反射率的模拟

在 Ex01 中，虽然模拟影像能够转换成反射率，但是这种模拟方式一次只能模拟一个方向，在需要同时模拟多个角度反射率时，效率较低，操作不便。因此，需要用到本练习中的多角度反射率模拟方法，其基本原理可参考第 1 章-1.2.4 节。

#### ■ 传感器参数设置

在【Type】中选择“PhotonTracing”（图 2-10），表示光子追踪模拟。在此模式下，由于不生成影像，因此无需设置影像宽度【Width】、高度【Height】以及光线数量【Samples】，但是需要设置【Illumination Resolution】，该参数表示入射光子的密度（间距），值越小，光子越密，结果越准确，但计算速度越慢。在此练习中，可以设置为 0.05。

在【Products】右侧，勾选“BRF”，将会显示【Number of Directions】、【Virtual Directions】以及【Virtual Detectors】：

- 【Number of Directions】表示上半球空间划分小面元（立体角）的个数（参考 1.2.4.2 节），每一个面元将会生成一个对应的反射率，在本次练习中可以设置为 10；
- 【Virtual Directions】表示用于计算方向反射因子 BRF 的虚拟方向，用天顶角和方位角表示，此处有两种设置方式：第一种为“天顶角 1:方位角 1;天顶角 2:方位角 2”，表示两个虚拟方向 11 和 22；第二种为“天顶角 1,天顶角 2;方位角 1,方位角 2”，表示两两组合四个方向（11, 12, 21, 22）。在此练习中，设置为“45:90;60:90”；
- 【Virtual Detectors】表示在某个方向放置一个具有一定尺寸的虚拟探测器，以模拟具有一定视场效应的方向反射率，该选项一般使用率较低，可以留空。

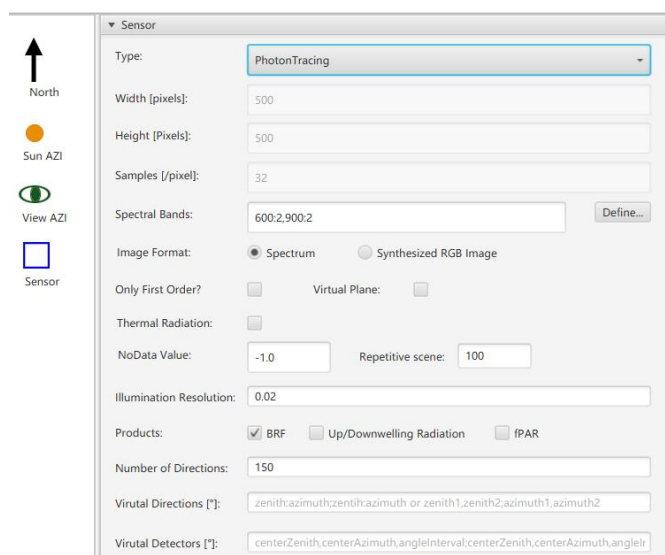


图 2-10 BRF 模拟传感器参数设置

### ■ BRF 模拟

点击菜单【Run】->【Run All】，开始模拟。模拟结束后，点击菜单【Tools】->【Open Results Folder】，打开模拟结果目录，文件“photontracing\_0\_05\_BRF.txt”保存了模拟的 BRF 结果（图 2-11），第一列为天顶角，第二列为方位角，从第三列开始，为每个波段对应的反射因子 BRF。前 10 行结果为【Number of Directions】设置的面元数对应的 BRF，最后两行为【Number of Directions】设置的虚拟方向对应的 BRF。一般而言，虚拟方向估算得到的 BRF 较为准确。

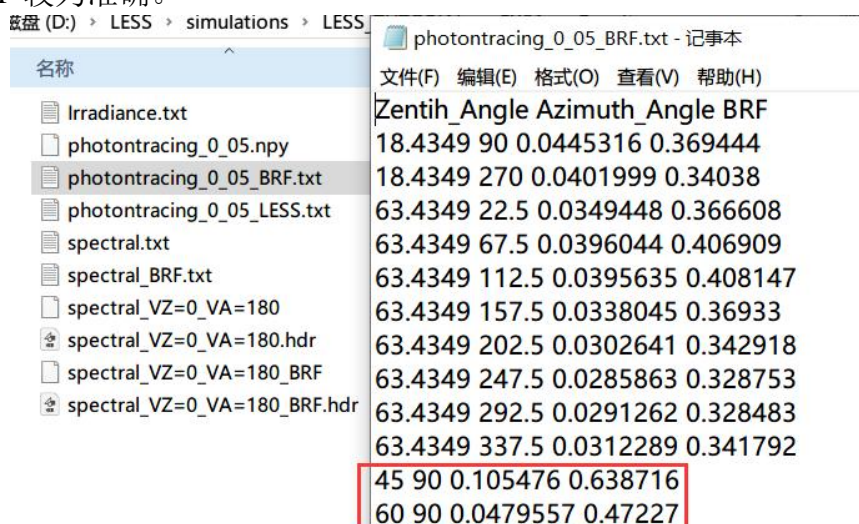


图 2-11 BRF 模拟结果

### ■ 反照率模拟

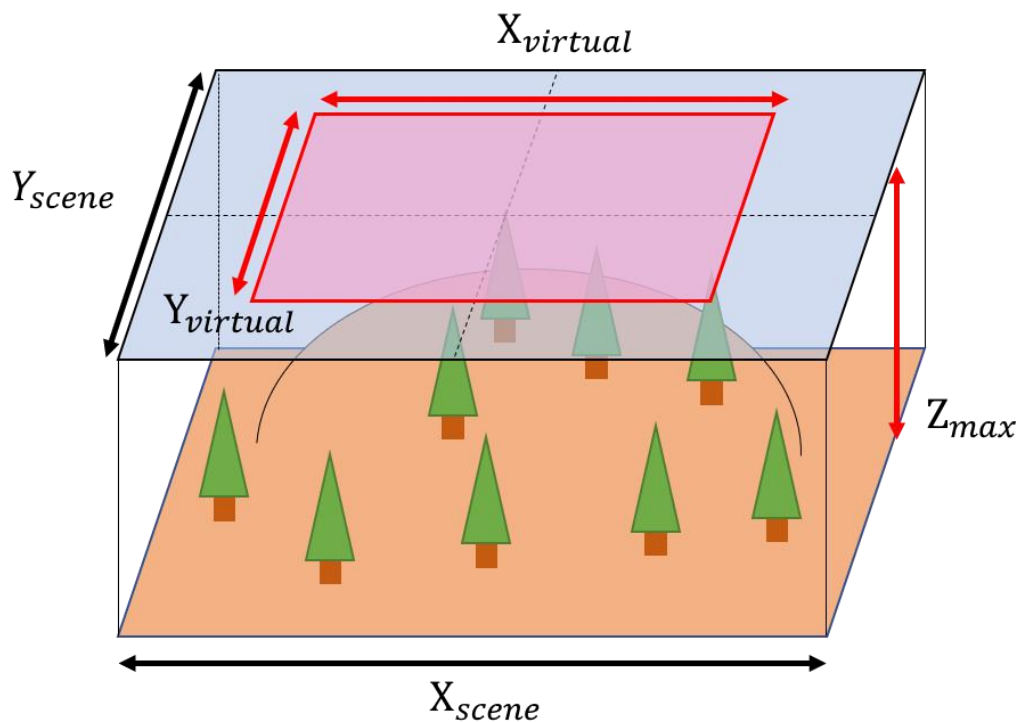
在结果文件夹中，“photontracing\_0\_05\_LESS.txt”保存了场景的反照率值，该值为对上半球空间所有面元中累计的能力进行积分得到。

## 虚拟平面

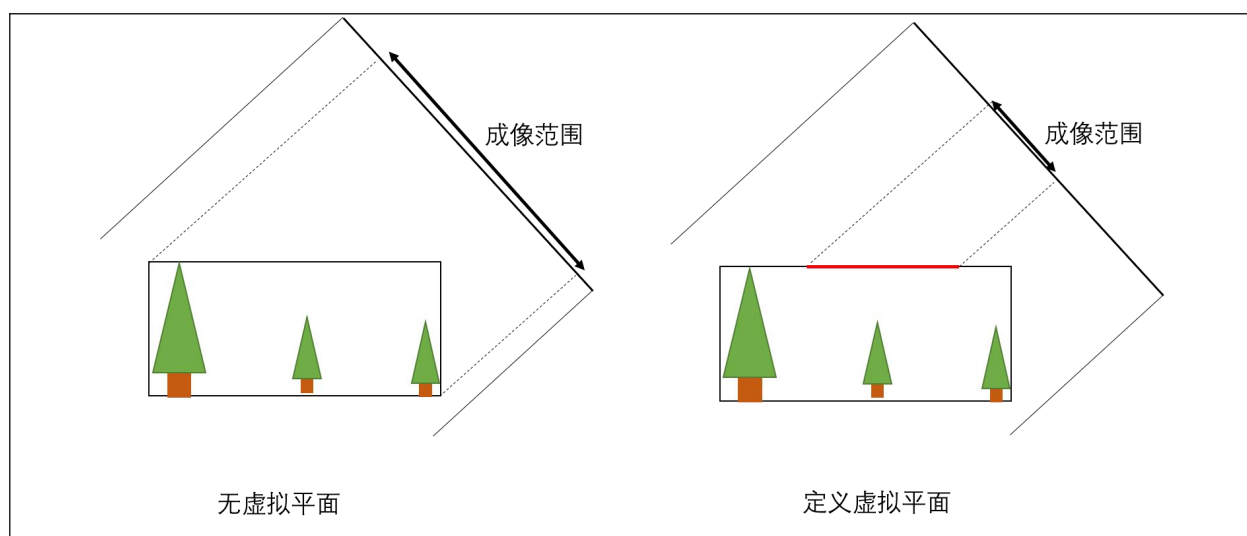
在【Sensor】中，有一个【Virtual Plane】选项，勾选之后会出现一组新的参数，这组参数能够定义一个虚拟平面。虚拟平面是指位于场景顶部的一个具有一定范围的四边形，在模拟场景 BRF 时以该四边形范围为基准，计算总的入射能量和出射能量（即只搜集穿过该四边形范围内的光子）。虚拟平面有平面中心位置【X】【Y】、尺寸【XSize】【YSize】决定，默认高度为场景的最高处，因此此处【Z】的值“MAX”不能修改。

|             |         |                                   |        |                                   |
|-------------|---------|-----------------------------------|--------|-----------------------------------|
| Center[XY]: | X:      | <input type="text" value="50.0"/> | Y:     | <input type="text" value="50.0"/> |
| Size:       | XSize:  | <input type="text" value="100"/>  | YSize: | <input type="text" value="100"/>  |
|             | Z[0~Z]: | <input type="text" value="MAX"/>  |        |                                   |

在 Photontracing 模式下，如果不定义虚拟平面，将会使用整个场景的包围盒顶部平面作为默认的虚拟平面。



在平行投影相机（orthographic）模式下，如果不定义虚拟平面则默认无虚拟平面；在虚拟平面时，成像范围为整个场景，在有虚拟平面时，成像范围为虚拟平面中所包含的场景。这种虚拟平面对于统计冠层的反射率以及方向间隙率等有十分重要的作用，可以避免场景侧面所带来的影响。



### 2.1.4 分层分组分 FPAR 模拟

FPAR 表示植被冠层绿色部分吸收的光合有效辐射(PAR)占总 PAR 的比率，一般吸收范围为 400~700nm。因此在模拟 fPAR 时，需要将波段范围设置为 400 到 700nm，波段数量尽量多，例如 30（图 2-12）。此外，在【Products】右侧勾选【fPAR】，勾选后会出现【Layer definition】，表示对场景进行分层，以得到每一层的 fPAR。设置格式为“起始高度:步长:结束高度”。在本练习中可以设置为“0:2:20”。

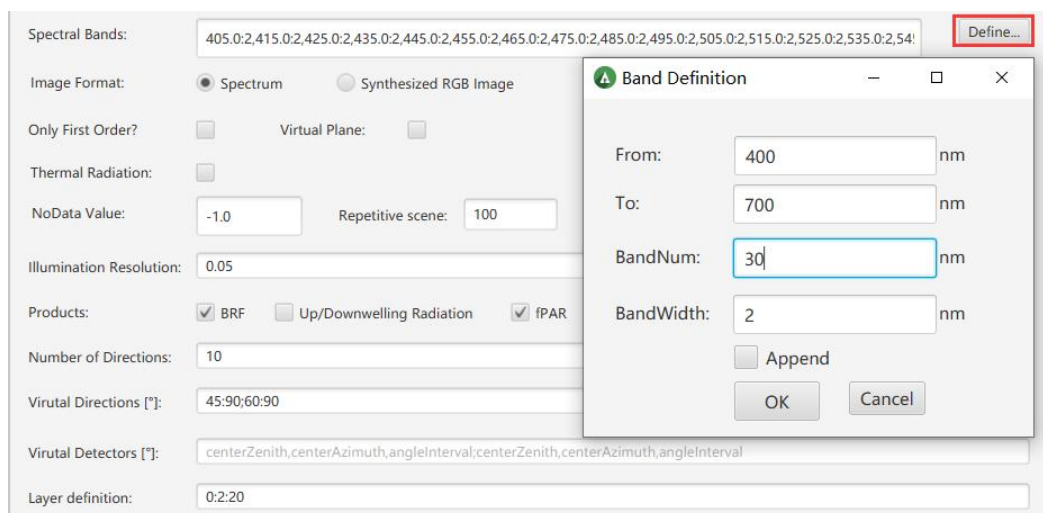


图 2-12 FPAR 模拟参数设置

点击菜单【Run】->【Run All】，开始模拟。模拟结束后，点击菜单【Tools】->【Open Results Folder】，打开模拟结果目录，文件“photontracing\_0\_05\_Layer\_fPAR.txt”保存了模拟的 fPAR 值。第一列及第二列表示每一层的起始和结束高度，第三列为总的 fPAR 值，第四列起始为每一个组分对应的 fPAR 值。

| 名称                                | 修改日期           | 类型   | 大小   |
|-----------------------------------|----------------|------|------|
| Irradiance.txt                    | 2019/8/20 3:12 | 文本文档 | 1 KB |
| photontracing_0_05.npy            |                |      |      |
| photontracing_0_05_BRF.txt        |                |      |      |
| photontracing_0_05_Layer_fPAR.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      |                |      |      |

|                                                                     |       |       |       |       |
|---------------------------------------------------------------------|-------|-------|-------|-------|
| 文件(F)                                                               | 编辑(E) | 格式(O) | 查看(V) | 帮助(H) |
| layer_bottom layer_upper TfPAR tree_leaves terrain tree_stem_branch |       |       |       |       |
| 0.0000 2.0000 0.2981 0.0000 0.2885 0.0096                           |       |       |       |       |
| 2.0000 4.0000 0.0087 0.0000 0.0000 0.0087                           |       |       |       |       |
| 4.0000 6.0000 0.0092 0.0000 0.0000 0.0092                           |       |       |       |       |
| 6.0000 8.0000 0.0634 0.0441 0.0000 0.0193                           |       |       |       |       |
| 8.0000 10.0000 0.1401 0.1180 0.0000 0.0221                          |       |       |       |       |
| 10.0000 12.0000 0.2736 0.2527 0.0000 0.0209                         |       |       |       |       |
| 12.0000 14.0000 0.1691 0.1614 0.0000 0.0076                         |       |       |       |       |
| 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                         |       |       |       |       |

图 2-13 FPAR 模拟结果

练习：用“photontracing\_0\_05\_LESS.txt”文件中的反照率值验证 fPAR 模拟结果是否正确？

在默认模式下，LESS 输出的是分层分组分，即同一个单木，如果被放置到场景中的不同位置，最后的值是累加在一起。如果要区分不同位置处的单木，则可以在 LESS 中通过在【Layer Definition】来设置，例如 0:2:20/1，其中 0:2:20 的定义与前述一直，/1 表示以**模式 1**进行模拟，该模式可以区分相同单木在不同位置处的 FPAR 模拟值，并且每个位置处的不同组分亦能够输出。除了区分不同位置以外，LESS 还提供了**模式 2**（例如 0:2:20/2），相比模式 1，模拟 2 则更加精细，是以三角面为基本单元进行输出，统计场景中每个单木在每个位置以及单木中每个三角面所接受的太阳辐射。需要注意的是，如果面元比较小，则要增加光子数，以减少模拟的不确定性，可以通过设置更小的【Illumination Resolution】来实现。在模式 2 中，如果只对场景中特定的部分物体比较感兴趣，比如样地中的某一株单木，则可以在 Layer Definition 中添加过滤，例如 0:2:20/2:tree\_0，则表示只对场景中名称为 tree，且位置为第一个的单木计算每个三角形的吸收能量，0:2:20/2:tree\_1 表示对场景中名称为 tree，且位置为第二个的单木计算每个三角形的吸收能量，其他单木则采用模式 1 来计算，即只统计每个位置处单木的每个组分总的能量。0 表示单木位置编号，从 0 起始，该顺序为【Objects】中每个物体的位置列表中的顺序。如果同时过滤多个物体，可以采用 0:2:20/2:tree\_0, tree\_1。模式 1 和模式 2 可以通过【3D view】进行可视化（下图）。需要注意的是，如果是可视化模式 2，最好是选择 PAR 文件进行可视化（\*\_fPAR\_par.txt），因为单个三角面元的 FPAR 值很小（输出的 FPAR 总入射为冠层顶部入射），可能会有浮点误差，颜色不太准确。如果在显示颜色时，需要设定颜色的范围，比如最小和最大范围，可以在 Parameters 目录下创建一个名为 3d\_par\_range.conf 的文件，内容为“最小值 最大值”，例如“0 10”，表示显示颜色范围为 0 到 10，两个数字用空格隔开。

FPAR 和 PAR 的输出文件中，默认输出的为波段积分值，也就是将所有模拟的波段进

行分段积分得到宽波段的总 PAR，如果还要输出每个模拟波段处的值，可以在 Parameters 目录下新建一个 par\_each\_band.conf 文件，该文件可以不填内容。

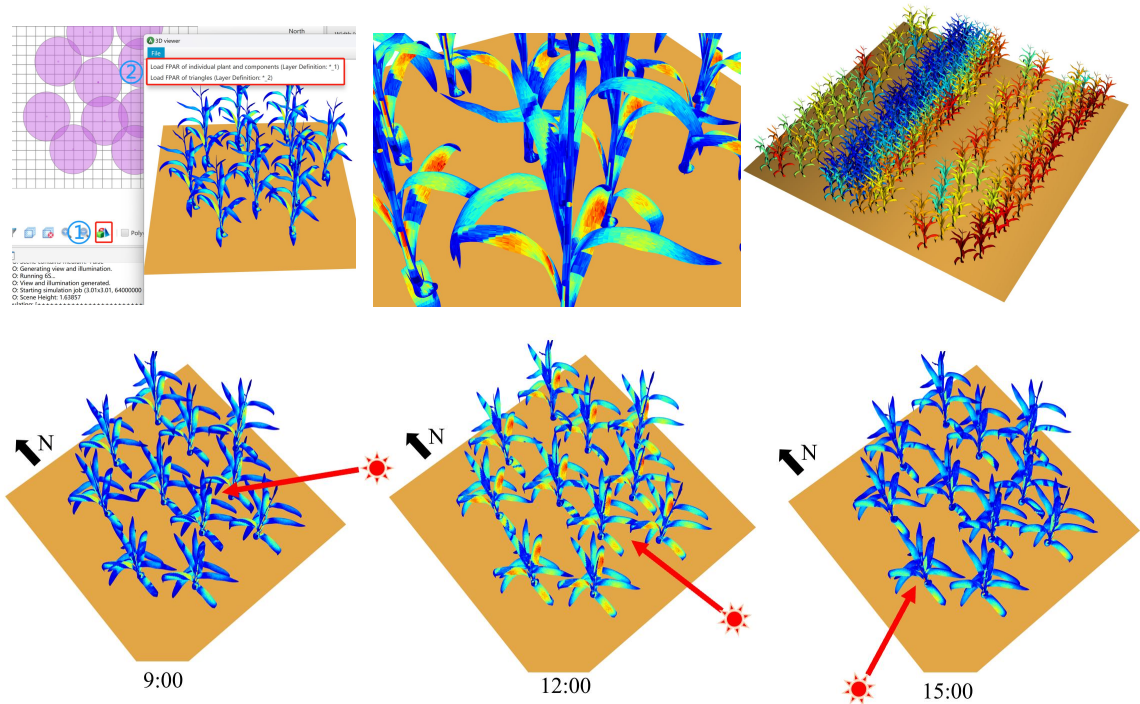


图 2-14 FPAR 模拟结果可视化

### 2.1.5 方向间隙率模拟

方向间隙率一般用于描述植被冠层在某一方向上的间隙的概率。在 LESS 中，可以通过模拟影像再分类得到，但是更为准确的方式是利用四分量影像，统计树冠间隙所占的比例。

#### ■ 参数设置

【Sensor】【Type】中选择“Orthographic”；【Spectral bands】设置波段数量大于或等于 5 个，例如“630.0:2,690.0:2,750.0:2,810.0:2,870.0:2”；勾选【Virtual Plane】；勾选【Products】右侧的“Four Components Product”；

【Observation】中设置【View Zenith】和【View Azimuth】为所需方向，例如 0 和 180；

#### ■ 四分量影像模拟

点击菜单【Run】->【Run All】，开始模拟。模拟结束后，点击菜单【Tools】->【Open

**Results Folder】**，打开模拟结果目录，文件“spectral\_VZ=0\_VA=180\_4Components”即为四分量文件（图 2-15）。该文件一共包含有 5 个波段：

- 波段 1 为每个像素所属的四分量类别：1-光照土壤；2-光照叶片；3-阴影土壤；4-阴影叶片；这里的类别是根据像素中四个组分所占比例的最大的组分决定的。
- 波段 2 为每个像素中光照土壤所占的比例，其值为[0,1]之间的数；
- 波段 3 为每个像素中光照叶片所占比例；
- 波段 4 为每个像素中阴影土壤所占比例；
- 波段 5 为每个像素中阴影叶片所占比例；

#### ■ 方向间隙率计算

因此，计算间隙率的公式为  $\text{mean}(b2+b4)$ ，即波段 2 与波段 4 之和，再对所有像素取平均值。

为了方便计算，LESS 提供了一个 Python 脚本“DirectionalGapProb.py”，位于目录“[安装路径]\LESS\app\Python\_script”中。可以使用 LESS 自带的 Python 解释器运行该脚本，例如：`[安装路径]\LESS\app\bin\python [安装路径]\LESS\app\Python_script\DirectionalGapProb.py -i [LESS 输出结果路径]\spectral_VZ=0_VA=180_4Components -o [自定义结果保存路径]\gaps.txt`，最终结果储存在“gaps.txt”文件中。如果有多个输出文件，可以采用通配符的形式调用该脚本。例如所有含有“4Components”的文件可以同时进行处理：`[安装路径]\LESS\app\bin\python [安装路径]\LESS\app\Python_script\DirectionalGapProb.py -i [LESS 输出结果路径]\spectral_VZ=*_VA=*_4Components -o [自定义结果保存路径]\gaps.txt`

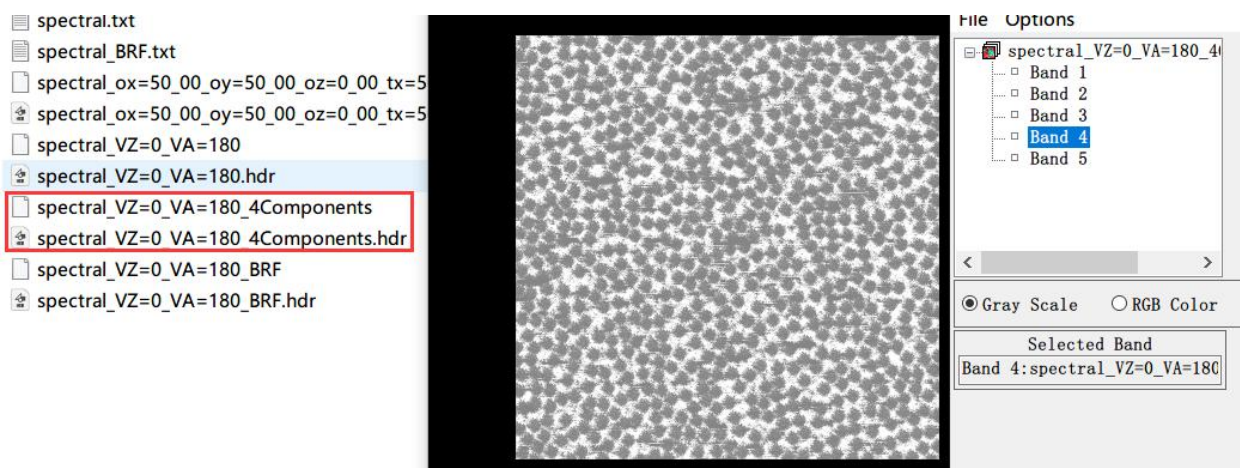


图 2-15 四分量模拟结果

### 2.1.6 热红外影像的模拟

#### ■ 参数设置

在【Sensor】中选择“Orthographic”，勾选“Thermal Radiation”，【Spectral bands】设置为 14um，例如“14000:1”。

在【Optical Database】中可以定义一系列组分温度，例如“T300”定义为“300:5”，表示场景元素被太阳光照射时的温度为 302.5K，处于阴影部分的温度为 197.5K。以同样的方式再定义一组组分温度“T290”，其值为“290:8”。

此时在【Terrain】下将会出现【Temperature】选项，选择“T290”；再打开【Object】下的【Define Objects】，将“tree\_leaves.obj”的温度设置为“T300”，将“tree\_stem\_branch.obj”的温度设置为“T290”。

#### ■ 影像模拟

点击菜单【Run】->【Run All】，开始模拟。模拟结束后，点击菜单【Tools】->【Open Results Folder】，打开模拟结果目录，文件“thermal\_VZ=0\_VA=180”即为模拟得到的热红外影像，再 ENVI 中打开该影像，每个像素值代表了方向亮温（Brightness Temperature）。

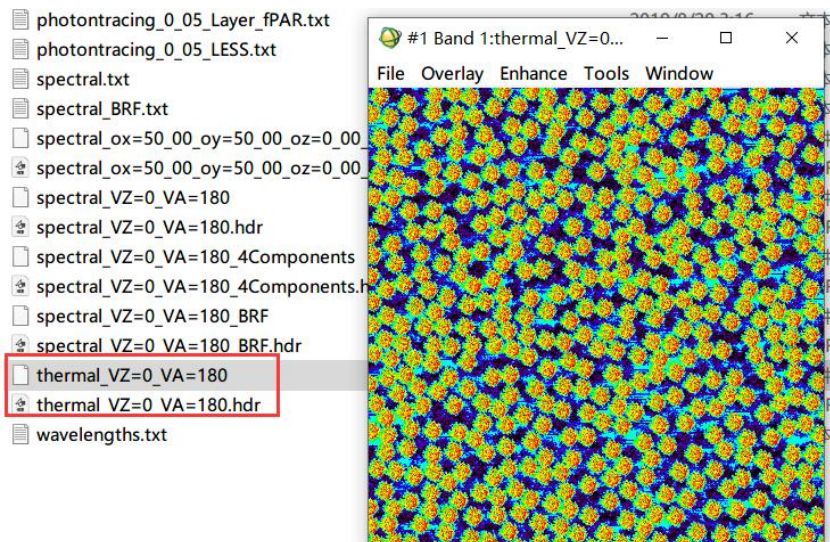


图 2-16 热红外模拟结果

### 2.1.7 激光雷达数据模拟

在 LESS 中目前可以从 GUI 界面中模拟机载激光雷达 ALS 和地基激光雷达的点云数

据（通过配置文件亦可模拟其他类型的点云数据，例如 MLS/星载）。激光雷达参数设置位置为【Sensor】【Type】【LiDAR】，首先新建一个工程，之后打开【LiDAR Simulator】，界面如下：

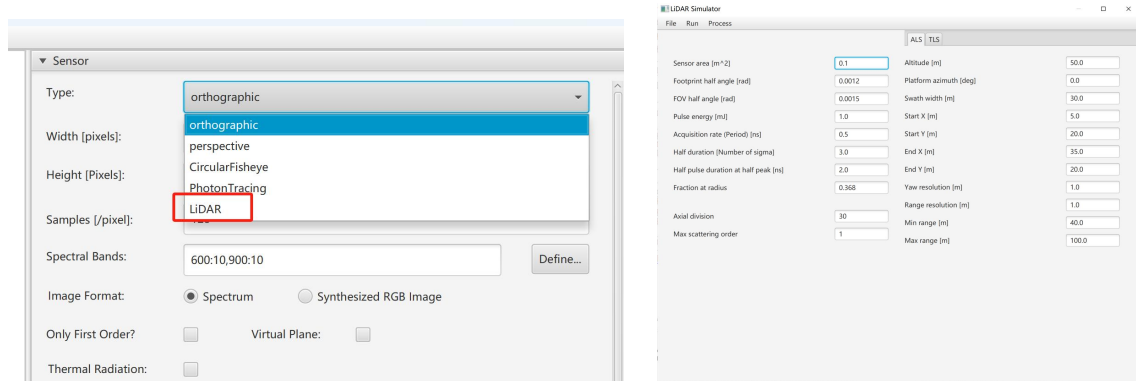


图 2-17 激光雷达模拟界面

参数分为两个部分，一是左侧的仪器参数，二是右侧的平台参数。各参数含义如下：

| Sensor 参数            |                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------|
| Sensor Area          | 激光雷达接收能量的传感器尺寸（面积），该值越大，接收到的能量越多。                                                                                 |
| FOV half angle       | FOV 表示激光雷达接收能量的视场范围，这里设置为视场圆锥张角的一半。该值越大，可接收范围越大，通过设置比发射脉冲张角较大即可。                                                  |
| Footprint half angle | 激光雷达发射脉冲的张角。注意：此处为视场圆锥张角的一半。                                                                                      |
| Axial division       | 将激光雷达脉冲横截面划分成 $n*n$ 等分，然后发射子光线。此处的 $n$ 即为 Axial division。这个仅对【Multi rays point cloud】模拟有效。值越大，每个脉冲模拟越精细，但是模拟时间越长。 |

| Mono Pulse 参数（单个脉冲）       |                                            |
|---------------------------|--------------------------------------------|
| X Y Z                     | 脉冲起始所在的位置。                                 |
| zenith/azimuth            | 脉冲天顶角和方位角。                                 |
| Delta zenith/azimuth      | 以中心天顶角和方位角为中心的扫描范围                         |
| Resolution zenith/azimuth | 天定方向/方位方向扫描的角度分辨率，单位为°                     |
| Min range, Max range      | 最终保存数据的范围，即距离传感器范围内[Min range, Max range]范 |

|                                   |                                                     |
|-----------------------------------|-----------------------------------------------------|
|                                   | 围内的数据。【注意：如果该参数不合理，可能无法得到模拟数据】                      |
| Pulse position and direction file | 输入一个 txt 文件，每行代表一个脉冲的起始位置和方向，x y z dx dy dz，可用于批处理。 |

| ALS 参数                            |                                                                                 |
|-----------------------------------|---------------------------------------------------------------------------------|
| Altitude                          | 激光雷达所在平台飞行高度，相对于场景零点而非地形。                                                       |
| StartX, StartY, EndX, EndY        | 扫描的起始点和结束点，即在地面上的投影，平台将沿直线从点（StartX, StartY, Altitude）飞行到（EndX, EndY, Altitude） |
| Swath width                       | 扫描的宽度，即垂直于飞行方向的水平宽度。                                                            |
| Yaw resolution                    | 沿着飞行方向的脉冲间隔（地面距离）                                                               |
| Range resolution                  | 垂直于飞行方向的脉冲间隔（地面距离）                                                              |
| Min range, Max range              | 最终保存数据的范围，即距离传感器范围内[Min range, Max range]范围内的数据。【注意：如果该参数不合理，可能无法得到模拟数据】        |
| Pulse position and direction file | 输入一个 txt 文件，每行代表一个脉冲的起始位置和方向，x y z dx dy dz，可用于模拟复杂扫描模式。                        |

| TLS 参数                     |                                                                                 |
|----------------------------|---------------------------------------------------------------------------------|
| TLS Position X Y Z         | 地基激光雷达所在的位置。                                                                    |
| StartX, StartY, EndX, EndY | 扫描的起始点和结束点，即在地面上的投影，平台将沿直线从点（StartX, StartY, Altitude）飞行到（EndX, EndY, Altitude） |
| Center zenith/azimuth      | 扫描的中心天顶角和方位角。                                                                   |
| Delta zenith/azimuth       | 以中心天顶角和方位角为中心的扫描范围                                                              |
| Resolution zenith/azimuth  | 天定方向/方位方向扫描的角度分辨率，单位为°                                                          |
| Min range, Max range       | 最终保存数据的范围，即距离传感器范围内[Min range, Max range]范围内的数据。【注意：如果该参数不合理，可能无法得到模拟数据】        |
| Echo detection mode        | 从内部全波形生成离散点云的模式，1: simple 简单局部最大值；2: Gaussian 高斯分解                              |

|                                   |                                             |
|-----------------------------------|---------------------------------------------|
| Pulse position and direction file | 输入一个 txt 文件，每行代表一个脉冲的起始位置和方向，x y z dx dy dz |
| TLS location file                 | 输入一个 txt 文件，每行可以代表一个 TLS 的位置 x y z          |

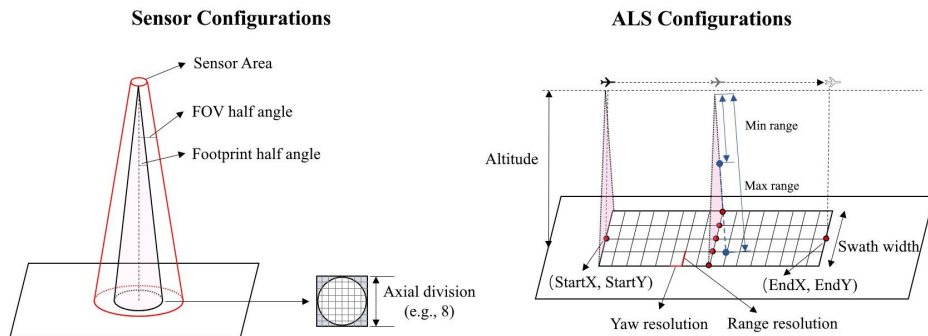


图 2-18 部分参数示意

### 离散点云模拟

设置好参数之后，点击【Run】菜单中的【Generate All】生产三维场景和激光雷达配置参数。然后点击【Single ray point cloud】或者【Multi rays point cloud】进行模拟。前者对每个脉冲发射一条光线，因此计算机速度较快。后者对每个脉冲发射多条光线（数量由 Axial division 决定），该模式在内部首先模拟得到全波形，再通过高斯分解得到离散点，因此计算速度比【Single ray point cloud】要慢。模拟完成后的结果分别在 Results 文件夹下的 singleRay 和 pointcloud 文件夹中。

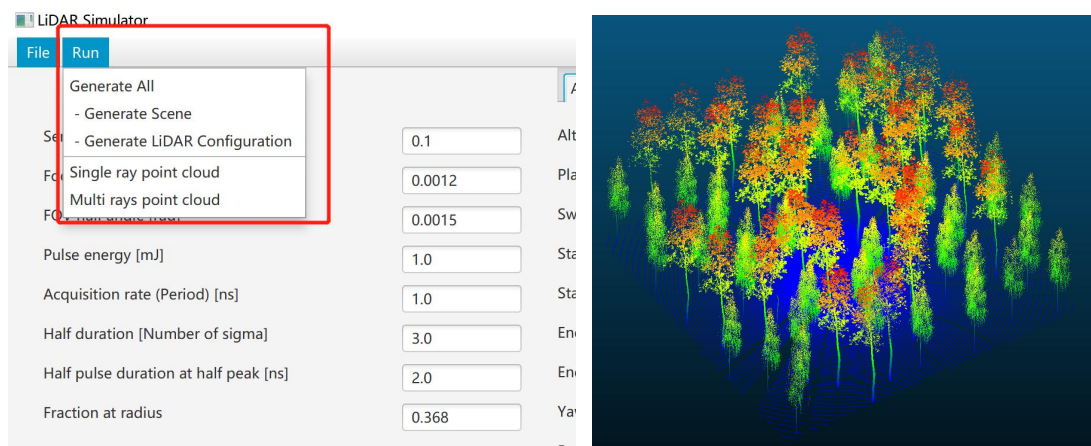


图 2-19 LiDAR 数据模拟示例

### 全波形多/高光谱激光雷达数据模拟

设置好参数之后，点击【Run】菜单中的【Generate All】生产三维场景和激光雷达配置参数。然后点击【Single ray point cloud】或者【Multi rays waveform】进行模拟。模拟完

的结果会在 Results 文件夹下的 records 文件夹中。但是该结果为中间结果，并不是最终方便使用的全波形数据。为了得到全波形数据，点击【Process】Convolve to generate waveform】按钮，进行卷积运行，得到的结果在文件夹 convolved\_waveform 中。该结果是成对出现的，例如 0.txt 和 pos\_0.txt [或者 1.txt 和 pos\_1.txt]。其中 0.txt 文件(如图 2-20)记录了所有的波形数据，数据总行数为脉冲数量\*每个脉冲离散数量（即 NumberOfBinsForEachPulse），因此第一个脉冲的数据为 0 到 NumberOfBinsForEachPulse-1 行，第二个脉冲数据为 NumberOfBinsForEachPulse 到 2\*NumberOfBinsForEachPulse-1 行，以此类推。数据的第一列为每个脉冲离散区间（bin）起始位置距离传感器的距离，第二列开始为每个波段的回波强度。另一个文件 pos\_1.txt 记录了脉冲的 POS 信息，前三列为脉冲起始位置，即传感器位置，的 XYZ 坐标，后三列为每个脉冲的反射方向。坐标系与 LESS 界面的坐标系一致。

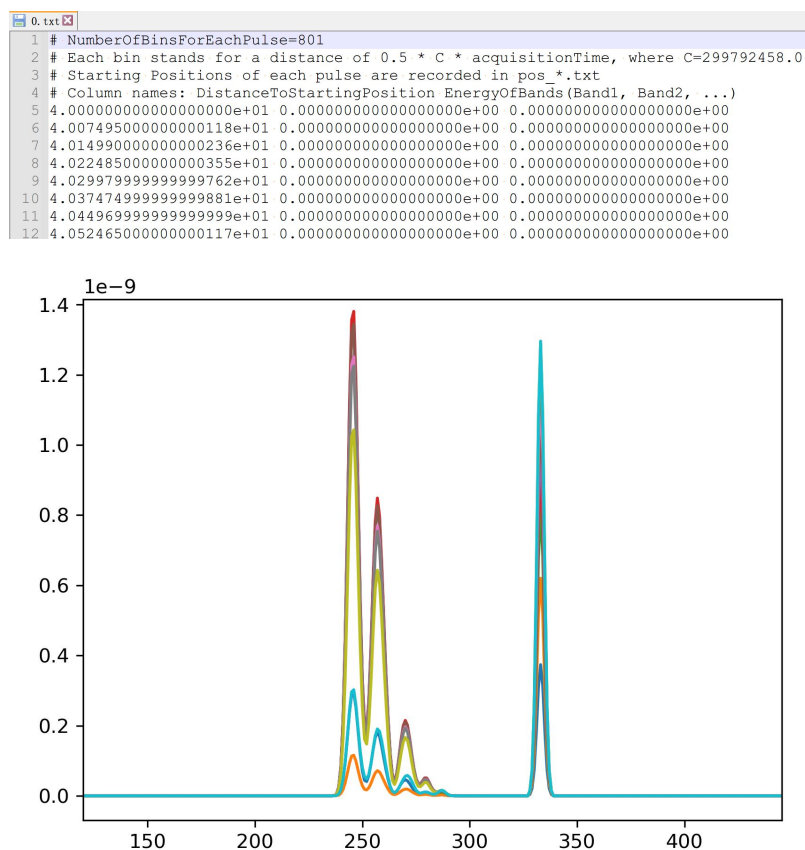


图 2-20 全波形文件及多光谱波形数据模拟

## 光子计数激光雷达

光子计数激光雷达能够记录单个光子的能量，其辐射传输过程与全波形类似，区别在于传感器探测时需要考虑对光子的探测概率，死区期间（探测一次之后需要一定时间恢复下一次探测能力），暗计数率（系统噪声频率）。在 LESS 中，如果要想实现光子技术

雷达模拟，则只需要在全波形模拟的基础上点击【Process】【Waveform to photoncounting】，则能够生成光子计数数据，生成结果在 photon\_counting 目录下。

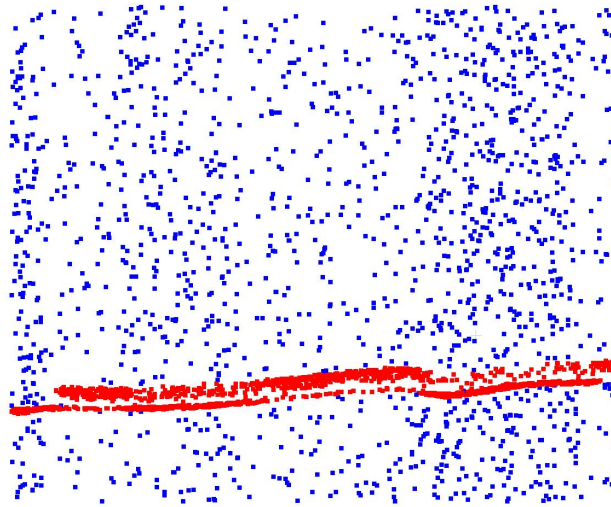


图 2-21 光子计数 LiDAR 模拟结果示意

### 2.1.8 日光诱导叶绿素荧光模拟

目前，LESS 在 Fluspect 基础上模拟了区分/不区分光系统 I 和 II 的多角度、多成像模式的冠层叶绿素荧光模拟。用户只需将【Sensor】【Products】【Chl.F】置为非“**No**”，并给发射荧光的 object 赋上荧光光谱，即由 Fluspect 模型生成的光谱，其余操作与模拟 BRF 无异，即可模拟叶绿素荧光。

【Sensor】【Products】【Chl.F】置为非“**No**”后，**需要注意**，所有与波长有关的输入参数（例如：入照辐射、天空散射光比例、自定义的光谱、输入的 Land Albedo Map 等），必需包含一个固定光谱，如**错误!未找到引用源**。小节所述，实际的模拟波段写在“.less\EFwlinsim.txt”中。

LESS 中内置了两个版本的 Fluspect，分别为 Fluspect\_B\_CX\_P6（区分 PSI 和 PSII 荧光）和 Fluspect\_B\_CX\_PRO（不区分 PSI 和 PSII 荧光），用户将【Sensor】【Products】【Chl.F】选为“Split PSI PSII”时使用前者，选为“Combine”时使用后者。两个模型对于干物质含量和荧光量子效率的处理有些许不同，前者不区分干物质类别，但区分 PSI 和 PSII 的荧光量子效率，后者可以区分碳基干物质和氮基干物质，但不区分 PSI 和 PSII 的荧光量子效率。需要注意的是，只有干物质含量设为 0 时，后者才能区分干物质类别，否则不区





图 2-22 荧光多角度文件及制图，和平行投影、鱼眼相机、透视投影的荧光影像

### 2.1.9 冠层光合速率模拟

在 PAR 模拟的基础上，能够得到面片或浑浊介质吸收的光能，再结合叶片光合作用模型，可以估算冠层的光合作用速率。该过程采用的叶片光合作用模型为 Farquar 模型，形式与 SCOPE 模型中的 biochemical 模块一致。

新建工程并创建场景后，选择 PhotonTracing 模式。在【Sensor】 【Products】 【Photosynthesis】中选择【FvCB】，此时仅有【Layer definition】可设置，其他产品不可模拟。【Layer definition】设置沿用 FPAR 模拟的设置方式。由于要模拟 PAR 波段的吸收，模拟的波段范围需要包含 400 到 700 nm。【Illumination Resolution】也需要适当加密。

【Photosynthesis】选择【FvCB】后，【Illumination & Atmosphere】中会出现【Ambient】参数组，目前起作用的参数仅有氧气浓度和大气压强。在【Optiacl Database】中，会出现【TemperatureDefinition】和【BiochemicalDefinition】，点击这两个按钮将弹出温度和光合参数设置界面，各参数区分阴阳叶输入，输入格式与热红外模拟一致。注意：【TemperatureDefinition】中的“T300”和【BiochemicalDefinition】中“-”不要删除。光合参数中， $e_l$  表示叶边界的水汽压， $C_s$  表示叶边界的  $CO_2$  浓度，其余参数与 SCOPE 对齐。由于叶片光合模型是与 SCOPE 对齐的，要求计算叶片叶绿素的光能吸收量。在 LESS 中，当光合组分的光谱被设置为 Fluspect 和 PROSPECT 模型生成的光谱，或光谱库中的 birch\_leaf\_green 和 larch\_leaf\_green 时，程序会在【Parameters】文件夹中自动生成 \*\_kChlrel.txt 文件，表示叶片叶绿素各波段的光能吸收比例。当光合组分的光谱不是上述的光谱时，则默认 kChlrel 为 1。

在【Objects Definition】中为光合组分设置温度和光合参数。在【Components】中选中光合组分，然后选择要赋予的【Optical Property】、【Temperature】和【Biochemical】。非光合组分也需要赋予【Optical Property】，但要在【Biochemical】中选择“-”来标识该组分不进行光合作用。【Biochemical】选中“-”后，非光合组分的温度为填充的“T300”，

且不允许修改。

光合模拟结果放在 **Results** 文件夹中，并带有“**photosynthesis**”标记。当前光合组分仅支持使用面片表示或浑浊介质表示，不支持二者混合表示。结果文件均以“,”分隔。

当光合组分使用面片表示时，\*\_A.txt 为光子击中（采样到）的面片的单位面积光合速率，\*\_APAR\_Ch1.txt 为光子击中（采样到）的面片的单位面积光子通量密度，\*\_S.txt 为光子击中（采样到）的面片的面积。\*\_A\_layer.txt 为估计的每层总光合速率，\*\_A\_layer\_sampled.txt 为采样到的每层总光合速率，\*\_APAR\_Ch1\_layer.txt 为估计的每层总光子通量，\*\_A\_layer\_sampled.txt 为采样到的每层总光合速率。如果在【**Layer definition**】中设置模式 1，则输出的 \*\_A\_instance.txt 为估计的单个组分的总光合速率，\*\_A\_instance\_sampled.txt 为采样到的单个组分的总光合速率，\*\_APAR\_Ch1\_instance.txt 为估计的单个组分的总光子通量，\*\_APAR\_Ch1\_instance\_sampled.txt 为采样到的单个组分的总光子通量。

当光合组分使用浑浊介质表示时，\*\_A\_layer.txt 为每层总光合速率，\*\_APAR\_Ch1\_layer.txt 为每层总光子通量，\*\_A\_instance.txt 为单个组分的总光合速率，\*\_APAR\_Ch1\_instance.txt 为单个组分的总光子通量。

### 2.1.10 分层光照/阴影叶片面积比例<sup>1</sup>

分层光照/阴影叶片面积比例是指冠层的每个水平分层中，所以叶片面积中，光照/阴影的面积比例。此处的面积是叶片的绝对面积，与四分量模拟中四组分（观测方向）比例有所不同。

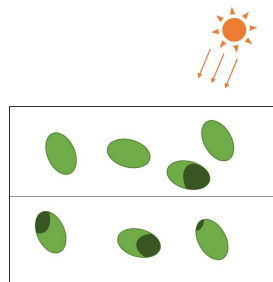


图 2-23 分层光照/阴影叶片面积比例

新建工程并创建场景后，选择 **PhotonTracing** 模式。为了启用光照/阴影面积比例计算功能，在【**Parameters**】目录下，创建一个名为 **sunlit\_leaf.conf** 的文件，文件内容为分层数量，例如“0:1:20”，表示从高度层 0 米到高度层 20 米，以 1 米间隔为分层。需要设置的参数

<sup>1</sup> 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。

为【Illumination Resolution】，该值越小，计算精度越高。其他产品，例如 BRF，fPAR 等可以不勾选。执行模拟之后，在【Results】目录下会生成 **photontracing\_\*\_\*\_Layer\_fSunlitLeaf.txt** 文件，该文件记录了分层光照/阴影叶片面积比例。在结果文件中，All Scene Components 表示场景中所有元素的光照/阴影比例。之后的结果分别对不同场景组分进行统计，例如场景中的叶片和枝干。结果中的 Total 表示冠层总的光照/阴影面积比例，需要注意的是，冠层总光照/阴影比例并不是每层比例的直接平均。

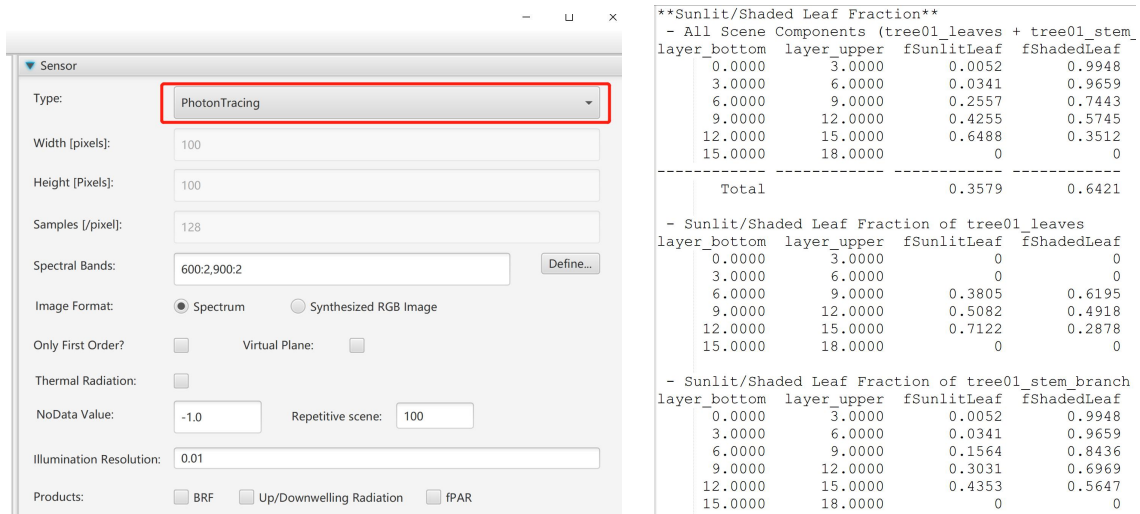


图 2-24 分层光照/阴影叶片面积比例计算界面和结果

## 案例

图 2-25 展示了一个简单场景的结果示例，在该场景中，有两个大叶片，距离地面的距离分别为 1m 和 2m，它们垂直投影完全重合。

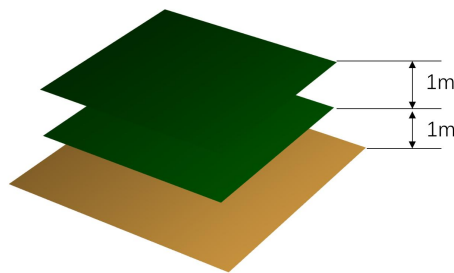


图 2-25 分层光照/阴影叶片面积比例结果示例

如果太阳在天顶垂直入射，则计算结果如下：

```

**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      |

下层叶片完全遮挡，上层完全光照，总冠层光照比例为 0.5。如果太阳天顶入射角度为  $45^\circ$ ，方位角  $90^\circ$ （正东方向），则计算结果如下：

```

**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      |

### 2.1.11 TRAC 模拟\*

TRAC 是常用的 LAI 测量仪器，其主要原理是记录太阳直射光的透过率。TRAC 测量过程通常是在样地中选择一些样线，在沿着样线行走过程中，TRAC 会以均匀时间间隔测量所在位置处的冠层透过率。在 LESS 中，可以通过等距间隔地形式，模拟一条样线上各位置处的透过率。透过率通过沿太阳方向发射光线确定，并且考虑太阳视半径( $0.25^\circ$ )。

使用方法为，在在【Parameters】目录下，创建一个名为 **trac.conf** 的文件，文件内容为样线起始点（第一行）、样线终止点（第二行）、距离间隔（第三行）以及每个位置处计算透过率所用的光线数量的量级（第四行）。需要注意的是，此处的光线数量只表示一个量级，如果设置 30，则发射光线数量约为 900 条。

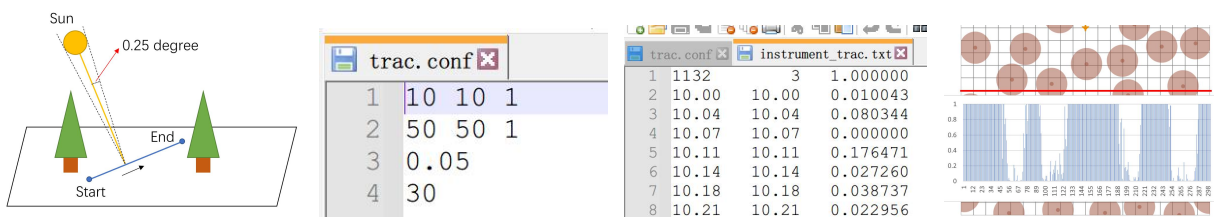


图 2-26 TRAC 模拟示例

模拟完成后，结果保存在 Results 目录下的 **instrument\_trac.txt** 中，其中第一行第一列的数表示总的打点数量，3 表示结果的列数，1.000000 表示一起的高度（在 **trac.conf** 中设置的 **z** 值）。从第二行开始，第一到第三列依次为打点处的 **x**，**y** 和透过率。

### 2.1.12 鱼眼影像透过率模拟\*

鱼眼影像透过率是指模拟一幅鱼眼图像，图像中每个像素是光线的透过率，天空部分为 1，完全被遮挡的像素为 0，混合像元为透过率的值，范围为 (0, 1)。需要注意的事，该模拟也支持浑浊介质模式，在此模式下，光线会计算浑浊介质的透过率，即  $e^{-G \cdot LAD \cdot t}$  (LAD 为叶面积体密度)。该模拟的设置与 2.1.2 节中鱼眼相机模拟完全一致，只是需要在 **Parameters** 文件夹下创建一个 **dtrans.conf** 文件，以启用该模式，否则模拟结果与 2.1.2 完全一致（即模拟得到每个像素的辐亮度信息）。

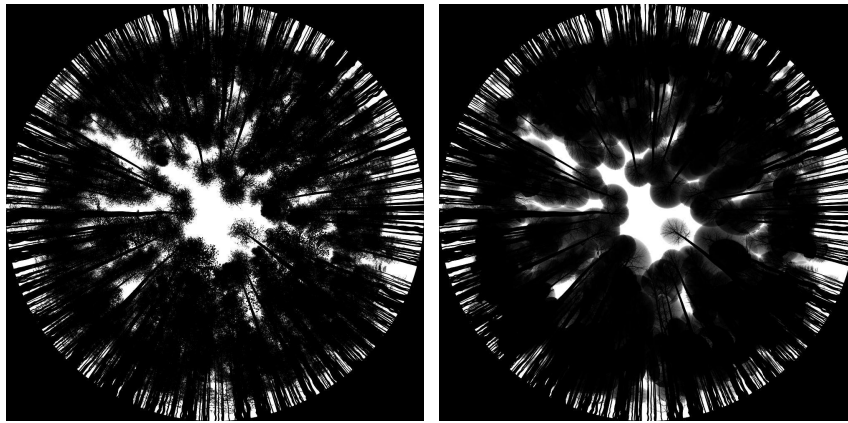


图 2-27 鱼眼相机透过率图像模拟示例

### 2.1.13 灯光模拟\*

在 LESS 中，某些物体可以被设置为具有一定发射功率的光源，因此，可以模拟夜间灯光的场景，要使用 LESS 的灯光模拟，可以像正常模拟方式一样构建场景，然后在 **【Parameters】** 目录下新建一个名为 **lights.conf** 的文件，文件内容为某个组分的发射光谱，发射光谱可以通过黑体发射(blackbody)和直接输入光谱来定义(spectrum)，例如图 2-28 所示，lights.conf 包含了两行内容，第一行表示将场景中的组分 light\_Disc.obj 定义为黑体，温度为 5000K，LESS 会根据温度和黑体发射自动计算所需要模拟的波段的发射光谱，第二行通过 spectrum 来定义光谱，光谱波段需要与 sensor 中定义的波段数量一致，其单位为  $w/m^2/sr/nm$ 。组分的名称可以从 objects.txt 文件或者 GUI 中查询。默认情况下，光学只从物体表面的法向量所在的一面（正面）发射，因此使用过程中需要注意 obj 文件法向量的定义，默认为右手定则。

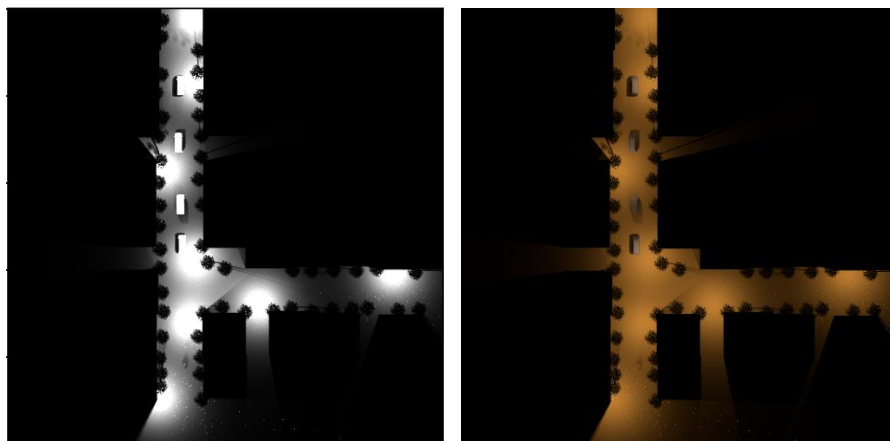
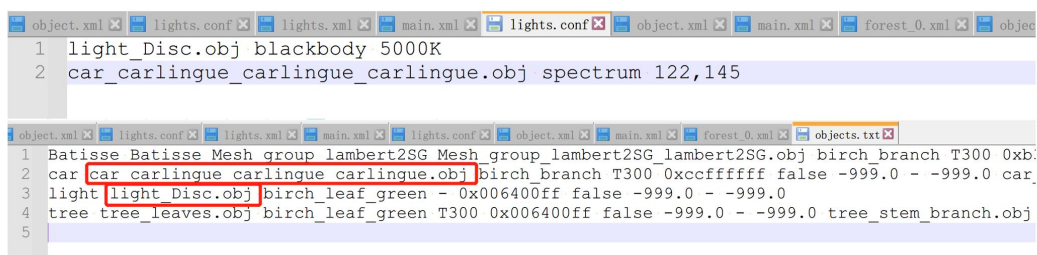


图 2-28 灯光模拟参数设置

### 2.1.14 通量模拟 (Flux) \*

### 2.1.15 设置模拟图像的投影坐标

对于正射影像，在 LESS 中可以设置其坐标系。LESS 默认采用 UTM 坐标系，在 Parameters 目录下可以新建一个 utm.conf 文件，文件有 5 行内容，如下：第一行表示 UTM 投影带，第二行为图像左下角的 X 坐标，第三行为图像左下角的 Y 坐标，第四行为 X 方向分辨率，第 5 行为 Y 方向分辨率。如果 utm.conf 文件存在，则在模拟生成的图像中将会有投影坐标信息。

48N

369308.30627884204

151343.60646855

5.001419879947476

5.00810103543689

### 2.1.16 再碰撞概率模拟

再碰撞概率是指光子与冠层发生散射之后，再次碰到冠层的概率，是光谱不变理论（spectral invariant）的核心概念，LESS 提供了模拟再碰撞概率、截获率、半球及方向逃逸概率等与光谱不变理论密切相关的参数。在 LESS 中，模拟再碰撞概率的方式与 FPAR 模拟同时进行，因此在【Sensor】中选择【PhotonTracing】之后，需要勾选【fPAR & Recollision Prob.】（如果只模拟再碰撞概率，则无需关心 Layer Definition 的值）。模拟结束之后会在结果文件夹下生成\*\_Prob.txt 文件，结果如下图，其中 Total probability 表示冠层总的再碰撞概率，是所有散射次数的累加；Canopy interception probability 表示入射光线总的截获率，即一次散射被冠层拦截的概率；Upward/Downward escape probability 表示冠层向上半球空间和下半球空间逃逸的概率，逃逸概率定义为与冠层发生散射后的光子逃逸出冠层的概率，该值为多次散射的累加；再碰撞概率+上半球+下半球逃逸概率=1。除了总的概率以外，输出结果还提供了每次散射的光子数，根据该光子数可以计算分散射次数的概率或者将其累加得到总的概率，Scat\_order 表示散射次数，Tot\_photons 表示与从冠层散射出发的总光子数，Its 表示再次与冠层碰撞的光子数，Up\_escape/Down\_escape 表示上/下半球逃逸的光子数。

上半球逃逸概率，又进一步细分为了方向逃逸概率，即提供了每个散射次数下在每个方向的逃逸光子数，根据光子数可以计算概率（如下图右）。Escape\_photons 表示该立体角内的总逃逸光子数量，所有方向之和等于半球总逃逸。Number of directions（N）表示将上半球空间均匀划分成了相等的立体角方向，每个立体角的大小为  $\frac{2\pi}{N}$ ，Zenith\_Angle/Azimuth\_Angle 表示每个立体角的中心方向，Azimuth\_Angle 为从北顺时针 [0°, 360°] 范围。根据提供的中心方向，可以反算每个立体角的天顶角的方位角范围，例如下图讲上半球空间划分为 10 个方向，天顶角和方位角均从 0 开始，则 18.43 表示 [0, 36.86] 这一天顶角范围，据此可以计算下一个天顶角范围为 [36.86, 63.43+(63.43-36.86)] = [36.86, 90.00]，以此类推。上半球方向逃逸概率的方向数量可以通过以下方式设置（默认为 1），在【Parameters】目录下，创建一个名为 p\_prob.conf 的文件，文件内容为方向数量，例如“100”，则输出文件中会出现相应的方向数量。

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |  |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|--|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> 10 -&gt; Spectral invariant (probability)** 11 -&gt; Canopy interception probability 12 0.4829 13 -&gt; Total probability 14 0.5396 15 -&gt; Upward/Downward escape probability 16 0.2289/ 0.2315 17 18 -&gt; Probability for each scattering order 19 20 Scat_order Tot_photons Its_photons Up_escape_photons Down_escape_photons 21 1 384302 199276 104944 81132 22 2 407110 215764 89168 102178 23 3 254787 140268 55787 58732 24 4 130078 105158 40855 44965 25 5 133949 74240 28656 31053 26 6 96527 53668 20456 22403 27 7 68900 38665 14592 15643 28 8 49656 27747 10505 11404 29 9 35313 19794 7482 8037 30 10 25288 14060 5501 5727 31 11 18010 10047 3832 4131 32 12 12906 7085 2822 2999 33 13 9132 5115 1946 2071 34 14 6564 3631 1354 1579 35 15 4653 2623 978 1052 36 16 3419 1926 726 767 37 17 2412 1343 524 545 38 18 1723 945 381 397 39 19 1191 672 239 260 40 20 2 0 0 2 41 21 0 0 0 0 42 22 1 0 1 0 43 23 0 0 0 0 44 24 0 0 0 0 45 25 0 0 0 0 46 26 0 0 0 0 47 27 0 0 0 0 48 28 0 0 0 0 49 29 0 0 0 0 50 30 0 0 0 0 </pre> |  |  |  |  | <pre> - 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 </pre> |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|--|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

图 2-29 再碰撞概率模拟

### 2.1.17 浑浊介质表达叶片

在 LESS 中，除了利用三角面片详细描述每片树叶之外，还可以利用浑浊介质对一簇树叶进行统计描述，该方法的优势在于无需每片叶子的详细信息，而只需要设定其叶面积体密度、叶倾角分布以及热点因子。使用这种表达方式的前提就是需要创建树冠的外包络，可以是简单几何体，也可以是体元等规则形状。在 LESS 中，常用的表达方式是 **alphashape**，他可以在三维点云或者面元模型的基础上创建一个紧密的外包络，从而模拟树冠的外形，内部则可以设定浑浊介质的参数。树冠的外包络可以利用 LESS 的【Tools】【Simple Crown Generator】进行创建，也可以从已有的三维 OBJ 模型（面片）创建。LESS 提供了一个从已有 OBJ 模型创建外包络的工具，为了方便批处理，该工具以 Python 函数的形式提供，可以直接从 LESS 主界面（下图右图）图标或者【Tools】【Python Console】进入 Python 终端，在此处可以直接运行代码。构建外包络的函数为：

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

其中 **obj\_path** 表示输入的 obj 路径；**alpha\_value** 表示构建 alphashape 时的半径参数，值越大，则构建的外包络越接近凸包，越小则为凹包，但是如果值过小则可能会失败，默认为 0.5；**out\_dir** 为输出文件的目录，如果不设置，则输出文件会默认保存到输入的 obj 文件相同目录；**keep\_groups** 表示只计算部分分组，即原始输入的 obj 文件可能存在多个分组，例如既包含了枝干，也包含叶片，通常我们构建外包络只需要针对叶片进行构建，因此这里可以只包络叶片，如果不设置该参数，则默认包含所有组分。一个典型调用示例如下图。在输出外包络的同时，还会输出一个相同名称的 txt 文件，文件记录了所有组分的表面积，以及计算得到的叶面积体密度，可以作为 LESS 的输入。

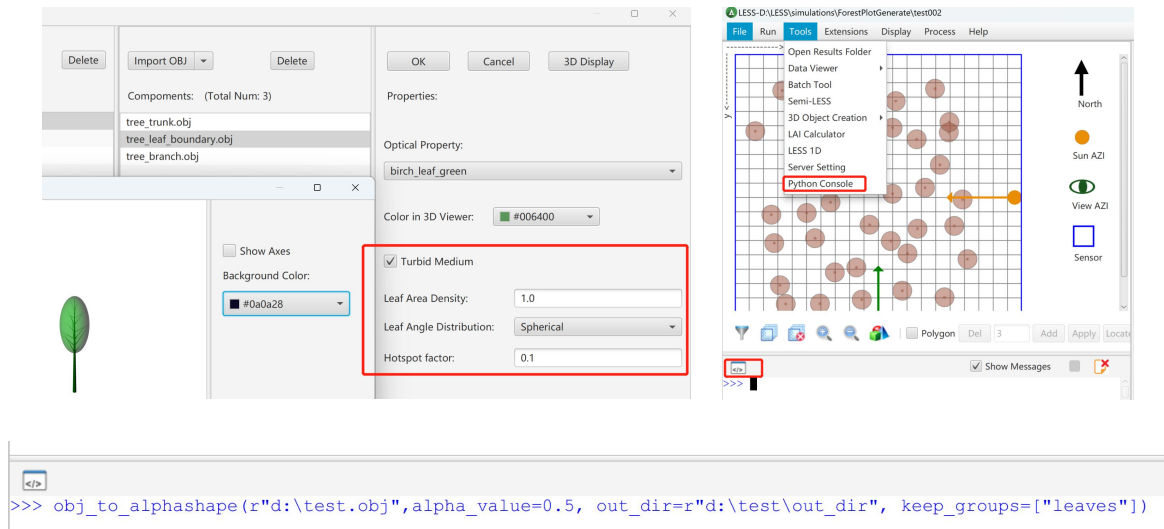


图 2-30 浑浊介质树冠创建及设置

## 2.2 实用工具

### 2.2.1 LAI 计算工具

点击菜单【Tools】->【LAI Calculator】，打开计算 LAI 界面。【Grid Size】为划分场景的网格个数，“Rows”为场景 X 方向的划分网格个数，“Cols”为场景 Y 方向的网格个数，“Layers”为场景 Z 方向的网格个数。例如，【Grid Size】值为 1x1x1，表示将整个场景计算出一个 LAI 值。若【Grid Size】值为 2x2x2，则表示将整个场景在 X 方向划分为 2 块区域，在 Y 方向划分为 2 块区域，在 Z 方向划分为 2 块区域，最终结果为 8 个 LAI 值。需要注意的是，【LAI Calculator】能够计算的并不仅仅是 LAI，需在下方勾选要计算的组分名称，若仅勾选叶片组分，则结果为 LAI。若勾选全部组分，结果为 PAI。如图，将场景划分为 8 块，分别计算 LAI 值，参数设置好后，点击【Run】。打开结果文件“LAI.txt”，第一行“Scene Size”为场景大小。第二行“LAI Dimension”为设置的划分场景的网格个数。从第三行开始为 LAI 值，LAI 值的排列对应分块的空间位置，前四个值为 height 为 1 时对应的值。

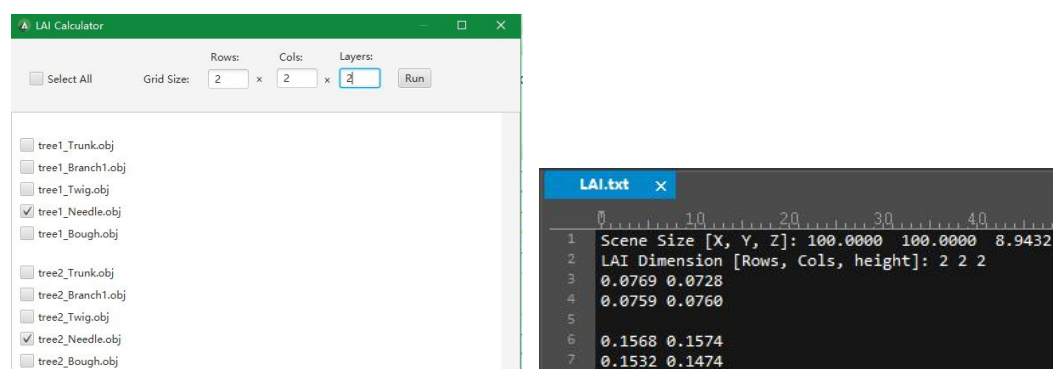


图 2-31 LAI 计算工具

### 2.2.2 批处理工具

当需要模拟的为同一个场景，而仅改变少量参数时，例如研究同一场景在太阳天顶角不断增加时天地反射率的变化，使用批处理工具将大量减少模拟所需的工作量。点击菜单【Tools】->【Batch Tool】，打开批处理界面。在【Group】中输入任意名称，选中刚才新建的【Group】，在左侧【Parameter】中找到需要改变的参数选中，例如“sun\_zenith”，点击【>>】，将其加入到右侧的【Paramter】中。加入改变的参数后，输入一系列该参数想要设置的值，例如在加入“sun\_zenith”后，输入“LIST:0/10/20/30/40”（此处需注意，“LIST”

为必填，以及在输入法为英文状态下输入逗号）。参数设置完成后，点击【Run】，在弹出窗口中保存文件，之后程序自动运行。打开结果文件夹，太阳天顶角为  $0^{\circ}$ ， $10^{\circ}$ ， $20^{\circ}$ ， $30^{\circ}$ ， $40^{\circ}$  的 BRF 文件分别保存在 test0 至 test4 中。

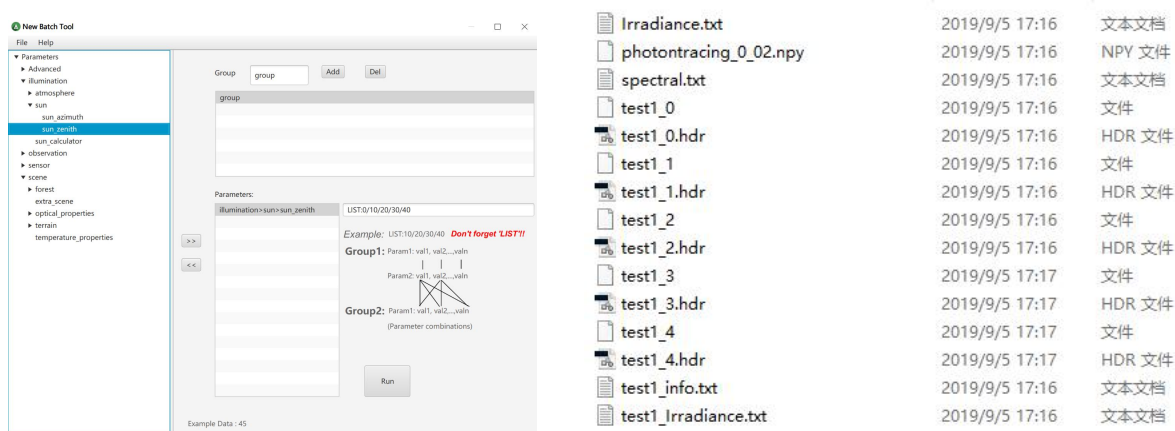


图 2-32 可视化批处理工具

## 2.2.3 影像和 LiDAR 波形查看器

LESS 默认模拟生产的影像格式为 ENVI 格式（即一对文件，其中一个为 hdr 头文件），默认保存在 Results 目录中。为了方便快速地查看模拟图像，LESS 提供了一个多波段影像查看器，位置为【Tools】【Data Viewer】【Image Viewer】，打开之后会默认打开 Results 目录下的所有的 ENVI 文件，可以选中其中一个，下方会列出该文件的所有波段，在波段基础上，可以选择灰度或者 RGB 合成来展示图像（如下图左）。

为了方便地查看 LESS 模拟的波形数据，LESS 同时提供了一个波形查看器，位置为【Tools】【Data Viewer】【LiDAR Waveform Viewer】。波形查看器默认选中 Results 目录下 convolved\_waveform 目录中的 0.txt，也可以自行切换波形文件，选中之后会列出该文件中的所有波形，选择其中一个，可以点击【Show】来展示波形。同时，如果选中下方的【Showing by selection】则在选中一个波形之后则会立即展示波形，而无须再点击【Show】，该功能可以快速浏览波形（如下图右侧）。

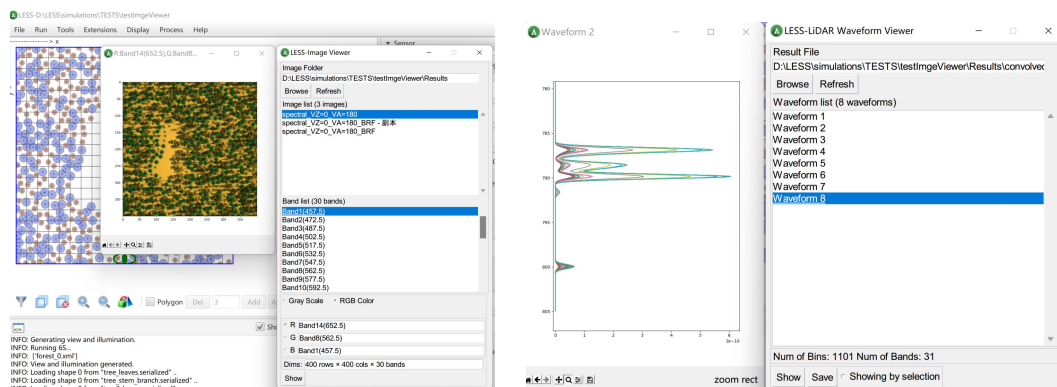


图 2-33 影像查看器和波形查看器

## 2.2.4 简单树冠生成工具

LESS 的图形界面提供了一个创建 OBJ 格式简单树冠的工具，【Tools】【3D Object Creation】【Simple Crown Creator】(如下图所示)。除了四种简单树冠（立方体、椭球、圆锥、圆柱）以外，LESS 提供了一种利用非对称高斯分布函数来描述的树冠，该树冠利用三个参数来控制其形状，Lower sigma 用于控制树冠下部形状，其值越小，树冠下部越平，Upper sigma 控制树冠顶部形状，其值越大，树冠越尖，Shape factor 控制树冠整体形状，其值越大，树冠越接近圆柱。如果勾选上 Generate boundary only，则只生成树冠的外轮廓，这种树冠模型主要用于浑浊介质表达叶片，在导入 LESS 之后，可以设定其叶面积体密度以及叶倾角分布。如果勾选上枝干【Include branches】，除了设定枝干参数以外，还能利用枝干来控制叶片的聚集，Leaf clumping factor 决定了叶片聚集的程度，当它为 1 时，叶片为随机均匀分布，当约接近于 0 时，叶片越聚集于枝条。Leaf distance factor 则表示了叶片距离树枝根部的距离，值越接近 1，则叶片越聚集于枝条的尖端位置。

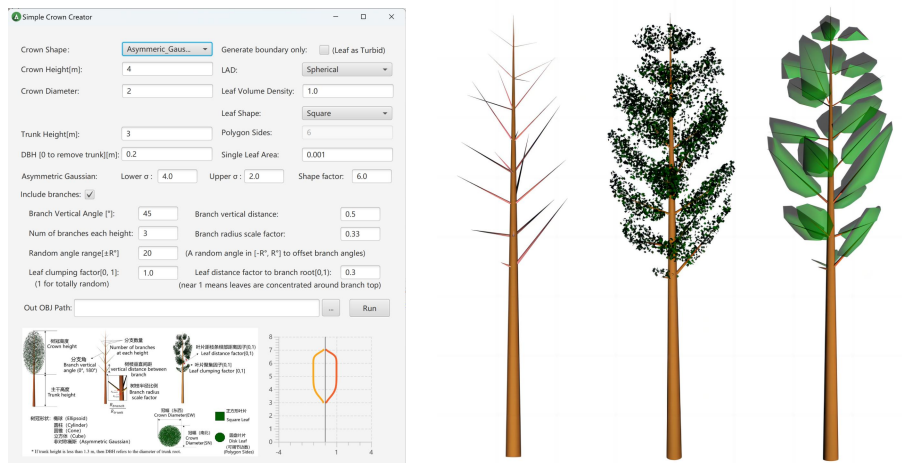


图 2-34 简单树冠生成工具

## 2.2.5 从机载点云重建三维场景

LESS 的图形界面提供了从机载点云重建三维场景的工具【Tools】【3D Object Creation】【3D Forest from LiDAR(ALS)】(如下图所示)。LESS 提供了体元、alphashape、椭球、圆柱和圆锥五种浑浊介质表达树冠。体元是将空间划分位均匀网格，以每个网格为基本单位计算其辐射散射；alphashape 是一种轻量级的树冠表达方式，在兼顾结构还原的准确性的同时计算效率也很高。椭球、圆柱、圆锥的方法计算效率更高，但模型过于简化，精度有限。

针对 alphashape，椭球，圆柱以及圆锥，该工具提供了四种创建树冠（单木分割）的方

法：分水岭算法，六边形算法和基于 CHM 或者点的聚类算法。分水岭算法适用于分散的树冠，对于树冠较密的连续场景分割效果不佳，结果造成有的树冠很大，有的很小，LESS 默认的分水岭窗口大小为 4.5m，也可以自行设置；六边形算法适用于密集均匀的树冠，其基本原理是在水平面内按照 6 边形对空间进行划分，以每个 6 边形为基本树冠单元；Kmeans 聚类算法适用于所有类型的树冠，该工具提供了两种聚类方式，一种是基于二维 CHM 影像聚类，效率较高，另一种为直接对三维点云进行聚类，效果较好，但效率可能较低。另外，可以设定一个高度阈值定义林下冠层的范围，如果勾选上 **Include Understory**，则还会对林下部分构建模型。该工具会对于重建好的每一个外包络进行植物面积密度（PAD）的计算，计算方法有四种：一种是脉冲追踪的方法，这种方法需要点云的 GPS 时间以恢复脉冲，从而进一步追踪每个脉冲的过程计算透过率，最终实现 PAD 的计算；一种是基于点云数量的方法，仅需要点云的三维坐标，利用地面点和非地面点统计透过率；此外，也可以设定一个 PAD 的定值，或者设定整个场景的 PAI，然后 PAD 可以通过树冠体积自动计算得到。

叶倾角分布有球型、统一型、平面型、竖直型、倾斜型、极端型六种分布，LESS 默认为球型分布。默认的 LESS 工程文件名为 **sim\_proj01**，默认的叶片光谱属性名称为 **leaf\_op\_name01**。参数设置好后，点击【**Create**】。完成之后，会在【**Output Directory**】目录下生成一个名为 **sim\_proj01** 的 LESS 工程，可以用 LESS 的 GUI 或者 SDK 打开，默认情况下，所有的树冠的光学属性均预设设为 **leaf\_op\_name01**，因此，打开工程后需要在【**Optical Database**】中去定义 **leaf\_op\_name01** 的具体的反射率/透射率值，然后可以进行模拟。

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]:

Segmentation Method: ☒ Watershed ☐ Hexagon ☐ Kmeans (CHM) ☐ Kmeans (Points)

Watershed window size [m][4.5 by default]:

Include Understory: ☐ Understory height threshold:

PAD Inversion method: ☒ Pulse tracing (need gps time) ☐ Point number (needs only XYZ)

☐ User defined constant PAD [m2/m3]:

☐ Total scene LAI (e.g., field-measurement):

Leaf Representation: ☒ Leaf as facet ☐ Single leaf area [m2]:

Output Directory:

LESS Project Name:

Leaf Angle Distribution:  (for LESS simulations)

Leaf Optical Property:  (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

图 2-35 简单树冠生成工具

### 2.2.6 Python 代码自动生成工具

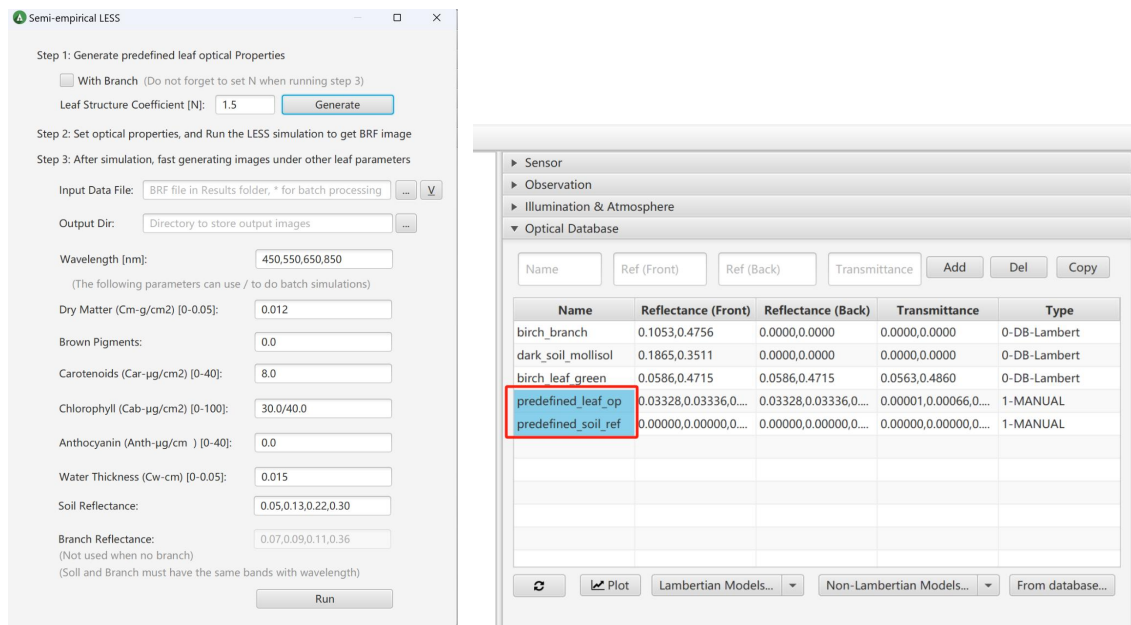
目前 LESS 具有两种运行方式,一种是通过操作 GUI,另一种是通过 Python SDK 脚本执行模拟。前者的好处是操作直观简单,后者的好处是能够很方便地控制模拟过程,例如添加元素,修改光谱,进而进行批量模拟。但是通过 SDK 进行模拟首先需要熟悉 SDK 的函数接口,具有一定的学习成本。为了降低使用代码的难度,LESS 提供了根据 GUI 参数自动生成相应的 Python 脚本的功能。使用方法为:首先打开 GUI 界面,然后设置相应的模拟参数,点击【Tools】【Generate Python Code】,选择保存路径,则会生成一个 py 脚本文件,该文件内容为利用 Python SDK 自动生成 GUI 所创建的模拟工程相对应的脚本,对该脚本进行一定的修改,则能够方便地修改参数或者进行批量模拟。

### 2.2.7 Semi-LESS 快速模拟数据集

利用辐射传输模型进行参数反演是定量遥感的主要研究内容之一。无论是查找表还是神经网络算法,通常均需要模拟不同 LAI,不同叶片色素以及不同土壤反射率下的冠层反射率。对于三维辐射传输模型来说,不同的 LAI 是通过构建不同的场景计算得到,例如不同树冠密度,而不同色素则可以通过设置不同的 PROSPECT 参数来模拟。一般而言,这样的参数组合数量是庞大的,这就要求辐射传输模拟能够以较高的效率来模拟数据集。然而,对于三维模型而言,大量的模拟仍然耗时。Semi-LESS 针对此问题提供了一种解决方案(Qi 等, 2023)。Semi-LESS 通过将解析模型与三维模型进行结合,从而能够得到与三维模型一样精确的模拟值,同时能够有效地利用解析模型的高效率性能。具体而言,对于一个构建好的三维场景(结构固定),当光照/观测几何以及叶片参数 N 固定时,可以先执行一次 LESS 模拟,将模拟的结果保存,基于这个模拟结果可以进一步解析地计算任意波段、叶片参数、土壤反射率下的冠层反射率。通过这样的方式能够很大程度减少批量模拟时对叶片和土壤反射率的组合数量,从而方便构建数据集。

在 LESS 中, Semi-LESS 模块位于【Tools】【Semi-LESS】(图 2-36),界面分为三步,首先第一步是生成一个预定义的叶片和土壤光谱,这两条光谱需要赋给场景的中叶片和土壤。当点击 Generate predefined leaf optical properties 时, LESS 界面的 Optical Database 中会新增两条光谱,名称为 predeterminded\_leaf\_op 和 predeterminded\_soil\_ref,请将这两条光谱分别赋给场景的土壤和所有的叶片组分。如果场景存在枝干,那么可以勾选界面中的【With Branch】。在执行所有模拟前,叶片参数 Leaf Structure Coefficient 也需要提前确定,如果

该值发生变化，那么 LESS 需要重新模拟。第一步完成后，第二步设置好光谱，然后可以执行正常的模拟，得到 BRF 图像（ENVI 格式）或者 Photon tracing 模式下的 BRF 文件（txt 格式）。第三步，将上一步得到的文件输入 Input DataFile，此时可以点击右侧的【V】按钮，改按钮表示对模拟结果的验证，点击之后会生成两张散点图，表示 Semi-LESS 的准确性。如果没有问题，那么就可以正式开始模拟其他叶片参数下的冠层反射率。输入需要模拟的波长，波长数量没有限制，其他参数按照正常的 PROSPECT 参数范围设置。需要注意的是，其他参数可以通过/进行批处理，带有/的参数会与其他参数进行两两组合，因此可以方便地进行大量数据的模拟。点击 Run 之后，会在 Output Dir 中输出所有的模拟结果。模拟结果通常会以\*\_0, \*\_1 的形式输出，每个文件对应的参数可以在同一目录下的 semi\_less\_parameters.txt 文件找到。图 2-36 对比了 Semi-LESS 模拟的结果与直接用 LESS 模拟的结果，可以看到近红外和红光波段的精度均很高。此外，如果想要用 Python 进行批量模拟，比如构建系列场景，可以通过命令行的方式。当点击 Run 之后，会在 LESS 的输出窗口中输出一条命令” E:\03-Coding\lessrt\Utility\Python310\python E:\03-Coding\lessrt\Utility\Python\_script\semiLESS\GenerateSpectraOrImage.py -i G:\LESS\simulations\TESTS\testSemi\Results\spectral\_VZ=0\_VA=180\_BRF -o G:\LESS\simulations\TESTS\testSemi\Results -wl 450,550,650,850 -N 1.5 -Car 8.0 -BP 0.0 -Cm 0.012 -Cab 30.0/40.0 -Anth 0.0 -Cw 0.015 -soil\_ref 0.05,0.13,0.22,0.30 -branch\_ref 0.07,0.09,0.11,0.36”，可以对改名了进行修改以实现批处理。



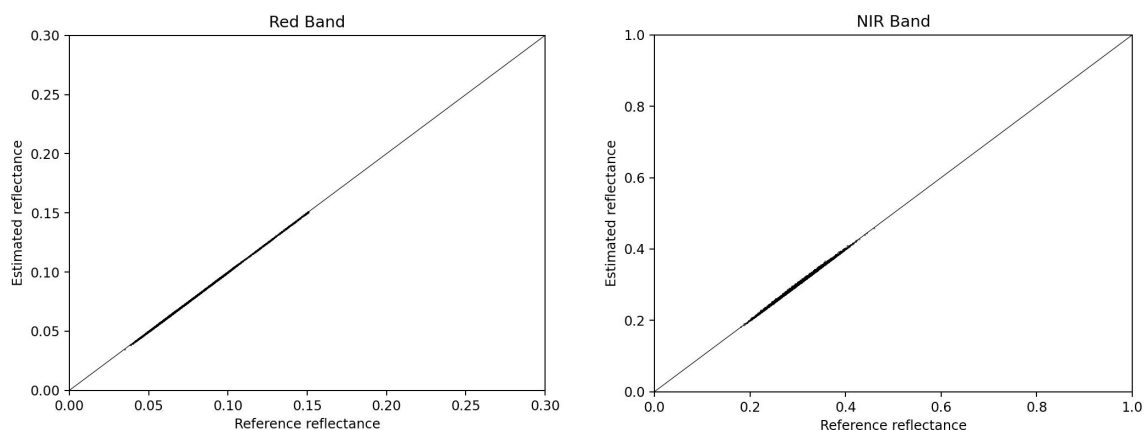


图 2-36 Semi-LESS 模块及精度验证

## 2.3 LESS 文件格式

### 2.3.1 instance.txt

在模拟工程的 Parameters 目录下，instance.txt 文件保存了每个 object 的位置信息，该文件有三种格式，分别表示不同的含义：

- 5 列模式：object\_name X Y Z Rotate, 该 5 列分别为 object 的名称，所在的 XYZ 位置，以及一个旋转角度，旋转角度默认为在水平面内逆时针旋转，单位为度。
- 8 列模式：object\_name X Y Z Rotate Axis\_X Axis\_Y Axis\_Z, 新增的三列为旋转时的旋转中心轴，采用右手旋转法则，旋转中心为 obj 文件的原点，先旋转再平移到 XYZ 的位置。
- 11 列模式：object\_name X Y Z Rotate Axis\_X Axis\_Y Axis\_Z Scale\_X ScaleY ScaleZ, 新增的三列为对 object 尺度定义，即最终该 object 的大小（非缩放因子），顺序为先缩放，再旋转，再平移。

为了 LESS 的 GUI 界面只提供了对 5 列模式的直接修改，其余模式可以直接修改 instance.txt 文件，然后再重新打开工程，或者从 GUI 界面导入 instance.txt 文件。该文件中的所有坐标定义同 GUI 界面相同，即 X 屏幕向右，Y 屏幕向下，Z 为竖直方向。此外，在 LESS 的 GUI 中对于每个选中的位置信息，点击【Extra Info...】按钮，在弹出的对话框中可以编辑列表中无法编辑的信息，即 5 列以外的额外信息。

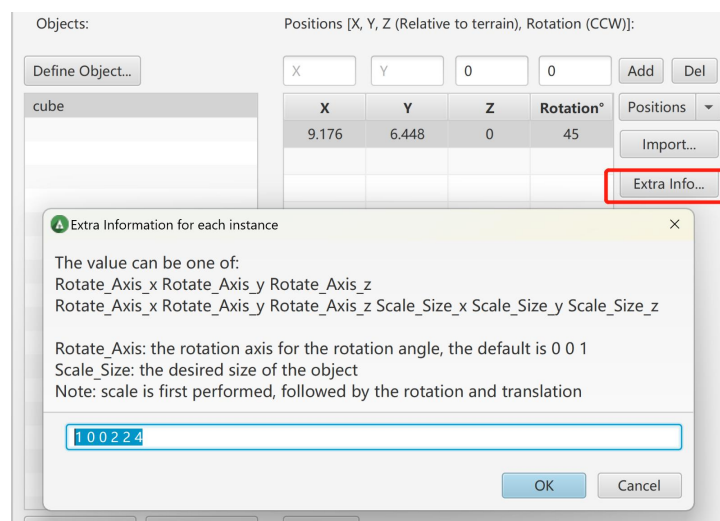


图 2-37 简单树冠生成工具

## 2.4 SDK for Python

### 2.4.1 利用 SDK for Python 实现 LESS 模拟

除了通过 GUI 程序实现模拟以外，LESS 还提供了一套 Python 版本的 SDK，通过该 SDK 可以实现利用 Python 脚本新建模拟工程、创建三维场景、修改模拟参数并最终进行数据模拟等功能。该 SDK 位于 LESS 安装目录下 pyLessSDK 文件夹中（例如：D:\LESS\app\Python\_script\pyLessSDK）。该 SDK 中主要的类及其关系如图 2-38 所示。

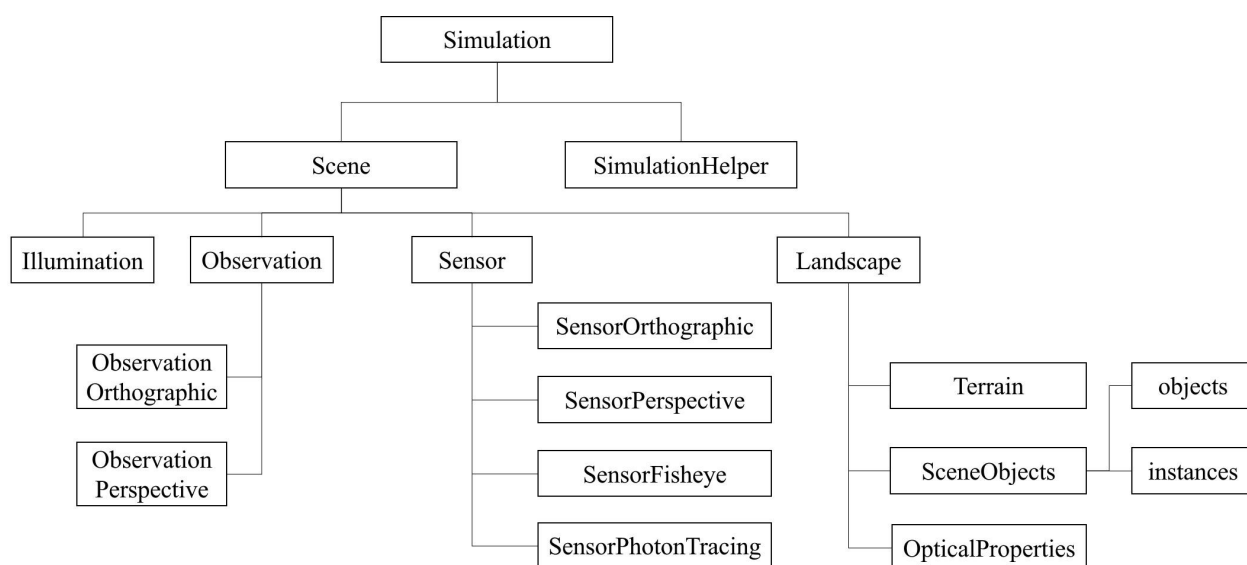


图 2-38 LESS SDK 中主要类及其关系

要使用 LESS SDK，首先需要合适的 Python 解释器，LESS 安装包已经集成了一个完整的 Python3.6 解释器（也可使用其他 Python 解释器），位于“LESS 安装目录\app\bin\python\python.exe”，推荐使用 PyCharm 集成开发环境，方便代码提示和程序调试。

新建工程后（例如 PyLessTest），在【File】【Settings】【Project Interpreter】中选择合适的 Python 版本，如图 2-39 所示。之后，可以新建 py 文件开始编写程序。为了能够使得该程序找到 LESS SDK，有两种方式，第一种通过 sys 模块添加路径：

```
import sys

sys.path.append(r"D:\LESS\app\Python_script\pyLessSDK")

from SimulationHelper import SimulationHelper
```

通过这种方式虽然可以执行程序，但是不利于程序开发，没有代码提示。在 PyCharm 中还有另外一种方式，如图 2-40 所示，【File】【Settings】【Project Interpreter】处点击下拉箭头，选择【Show All】，然后选择所使用的 Python，在其路径处添加 LESS SDK 所在的位置，之后即可直接导入 SDK 的模块。

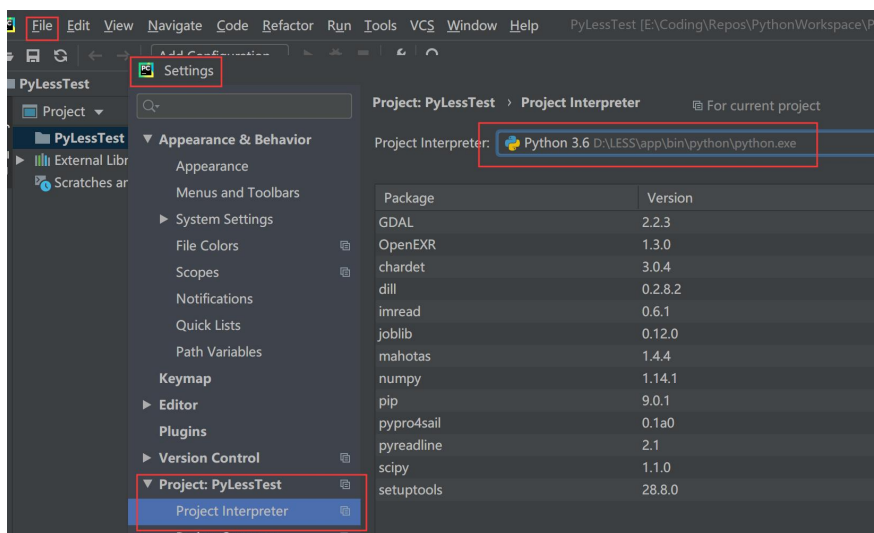
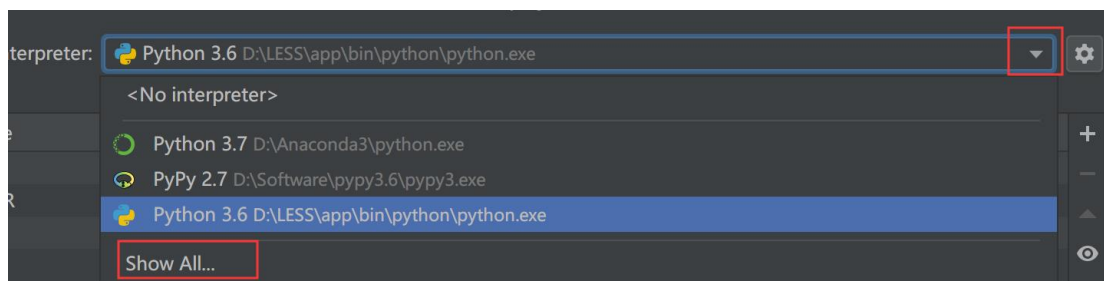


图 2-39 PyCharm 配置



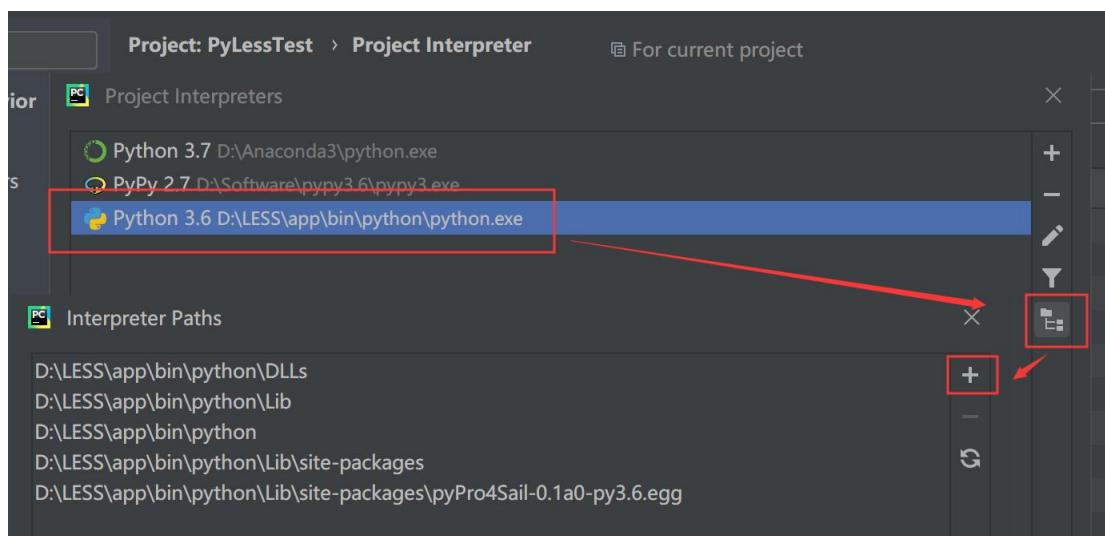


图 2-40 PyCharm 中配置 LESS SDK

### 2.4.1.1 Ex01 基本模拟

以下代码展示了利用 python 创建一个默认模拟工程，并直接进行模拟的示例。首先导入 SimulationHelper 和 Simulation 两个类，前者是模拟工程的辅助类，后者代表整个模拟工程。Simulation 包含一个 SimulatinHelper 和一个 Scene 成员变量。SimulatinHelper 主要提供创建新的模拟工程以及获取一些系统参数（例如 python 的路径等）等功能，例如 create\_new\_sim 函数根据一个输入的目录路径创建一个新的模拟工程，并用标准模板初始化该工程（设定默认参数等），如果模拟工程已经存在，则跳过。Simulation 实例从一个模拟工程路径和一个 SimulatinHelper 实例构造而来，得到 Simulation 实例后（即 sim），需要通过 read\_sim\_project 函数读取模拟工程的参数到 sim 对象中，然后可以调用 save\_sim\_project() 函数保存工程（此处没有修改参数，也可以不保存），最终调用 start () 函数执行模拟。

```
# 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") # 创建 SimulationHelper, 参数为 LESS
的安裝根目录
sim_helper.create_new_sim(sim_dir) # 新建模拟工程
```

```

sim = Simulation(sim_dir, sim_helper) # 初始化 Simulation 对象
sim.read_sim_project() # 读取模拟工程的内容
sim.save_sim_project() # 保存工程
sim.start() # 开始模拟

```

以上代码是一个最简单的模拟过程，无需打开 GUI 程序即可实现。最终的模拟结果是一副图像（`spectral_VZ=0_VA=180`），保存的位置为 `D:\LESS\simulations\Ex01\Results\`，默认保存为 ENVI 格式，像素值的类型为辐亮度（Radiance）。

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 |

图 2-41 模拟得到辐亮度图像

为了将辐亮度转为反射率，可以使用 `PostProcessing` 中的 `radiance2brf` 函数，该函数具有三个参数，第一个为模拟工程的路径，用以获取辐照度等信息；第二个为输入文件路径，即辐亮度文件，也是上一步中模拟得到的文件；第三个为输出文件路径，即 BRF 文件。模拟结束后得到图 2-42 所示 BRF 文件。（代码中的高亮加粗代码为新增代码）

```

# 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") # 创建 SimulationHelper, 参数为 LESS
的安裝根目录
sim_helper.create_new_sim(sim_dir) # 新建模拟工程
sim = Simulation(sim_dir, sim_helper) # 初始化 Simulation 对象
sim.read_sim_project() # 读取模拟工程的内容
# 模拟的输出文件
sim.save_sim_project() # 保存工程
sim.start() # 开始模拟

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

```

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

| <input type="checkbox"/> 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 |

图 2-42 模拟得到反射率图像

上述代码使用的是默认的输出文件名 `spectral_VZ=0_VA=180`（通过函数 `sim.get_dist_file` 得到），实际上，为了方便使用，我们在模拟时，可以设定自己的输出文件名，在 `read_sim_project` 和 `save_sim_project` 之间通过修改参数实现，代码如下：

```
sim.read_sim_project() # 读取模拟工程的内容
sim.set_dist_file(os.path.join(sim.get_sim_dir(), "Results",
"output_radiance_file"))
# 模拟的输出文件
sim.save_sim_project() # 保存工程
```

#### 2.4.1.2 Ex02 添加场景元素

在 Ex01 中，虽然我们实现了整个模拟流程，但是默认场景中只包含了一个简单 100m\*100m 的平地。本例中，我们将在场景中添加一些树木，并设定其光谱。

##### A. 导入场景 OBJ 文件

在 LESS 中，目前所有的场景元素（树木、房子等）均以三角面片的 OBJ 格式文件作为输入，树模型等可以通过第三方软件如 OnyxTree 等生成。

所有的场景元素均在类 `Scene` 中进行管理，当创建一个 `Simulation` 后，可以通过 `sim.get_scene()` 得到 `Scene` 的实例，进而访问或者修改场景中的元素。示例代码如下：

```
# 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") # 创建 SimulationHelper, 参数为 LESS 的
安装根目录
sim_helper.create_new_sim(sim_dir) # 新建模拟工程
sim = Simulation(sim_dir, sim_helper) # 初始化 Simulation 对象
sim.read_sim_project() # 读取模拟工程的内容
sim.set_dist_file(os.path.join(sim.get_sim_dir(), "Results",
"output_radiance_file"))

#####添加场景元素 start #####
scene = sim.get_scene() # 得到 Scene
landscape = scene.get_landscape() # 得到 LandScape 对象
landscape.clear_landscape_elements() # 清除工程已有的场景元素 (重复运行程序时, 必须
执行)

obj_tree01 = SceneObject("Tree01") # 定义一个场景物体, 名叫 Tree01
# 添加一个 obj 文件作为 tree1 的一个组分, 对于包含多个 group 的 obj 文件, 首先需要拆分并获取每
个组的路径, 再分别导入, 每个 group 设置不同的光学属性
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 包含两个组分, 第一个为 leaves, 第二个为 branch, 因此分别导
入。
# birch_leaf_green 和 birch_branch 为默认已定义的内置光学属性, 自定义光学属性将在下一个案
例讲解
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)

# 当定义好一个 obj 文件时, 还需要将其放置在不同的位置, 在 LESS 中称之为 instance
# 随机放置在场景中, 场景默认大小为 100m*100m
for i in range(1000):
    x = random.random()*100
    y = random.random()*100
    landscape.place_object("Tree01", x=x, y=y)

```

```
##### 添加场景元素 end #####
# 模拟的输出文件
sim.save_sim_project() # 保存工程
sim.start() # 开始模拟
PostProcessing.radiance2brf(sim.get_sim_dir(), sim.get_dist_file(),
sim.get_dist_file()+"_BRF")
```

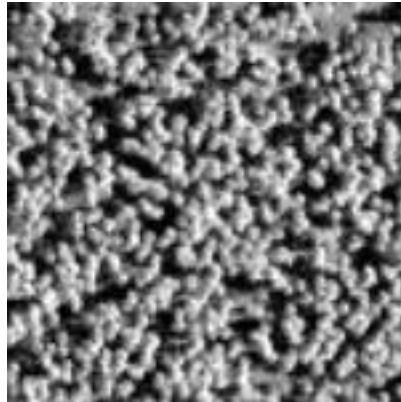


图 2-43 在场景中添加树模型

## B. 自定义光学属性

在 LESS 中，一个光学属性通常是指一对反射率和一个透射率，即正面和反面的反射率以及透射率（假设两面的透射率相同）。定义光学属性的类为 `OpticalItem`。在上例的基础上，需要改变的代码为：

```
scene = sim.get_scene() # 得到 Scene
landscape = scene.get_landscape() # 得到 Landscape 对象
landscape.clear_landscape_elements() # 清除工程已有的场景元素（重复运行程序时，必须执行）

# 定义一个光学属性（正面反射；背面反射；透射）
landscape.clear_user_defined_optical_properties() # 清空用户自定义的光学属性
# LESS 初始时为两个波段
op_item01 = OpticalItem("op_leaves", "0.05,0.4;0.05,0.4;0.05,0.4")
landscape.add_op_item(op_item01) # 添加到场景库
op_item02 = OpticalItem("op_branch", "0.2,0.45;0,0;0,0") # LESS 初始时为两个波段
landscape.add_op_item(op_item02) # 添加到场景库

obj_tree01 = SceneObject("Tree01") # 定义一个场景物体，名叫 Tree01
```

```

# 添加一个 obj 文件作为 tree1 的一个组分, 对于包含多个 group 的 obj 文件, 首先需要拆分并获取每个组的路径, 再分别导入, 每个
# group 设置不同的光学属性
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 包含两个组分, 第一个为 leaves, 第二个为 branch, 因此分别导入
# birch_leaf_green 和 birch_branch 为默认已定义的内置光学属性, 自定义光学属性将在下一个案例讲解
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. 自定义传感器

在例子 Ex02-A 和 Ex02-B 中, 我们使用的是默认的传感器, 即正射投影, 影像的大小为 100 像素\*100 像素, 在本例中, 我们将通过 SDK 修改传感器的参数。代码示例如下:

```

from Observation import ObservationOrthographic
...
# 当定义好一个 obj 文件时, 还需要将其放置在不同的位置, 在 LESS 中称之为 instance
# 随机放置在场景中, 场景默认大小为 100m*100m
for i in range(1000):
    x = random.random()*100
    y = random.random()*100
    landscape.place_object("Tree01", x=x, y=y)
##### 添加场景元素 end #####

# 修改传感器参数
sensor = scene.get_sensor() # 首先得到 Sensor
sensor.set_film_type("rgb") # 设置为 rgb, 可直接输出 png 图像, 如果设置为 spectrum, 则输出一个 envi 格式的多波段图像
sensor.set_image_width(300)
sensor.set_image_height(300)
sensor.set_sample_per_pixel(16)

```

```

# 新建一个 Observation, 然后设置到场景中, 也可以直接读取, 再修改
obs = ObservationOrthographic() # 默认观测天顶角为 0, 方位角为 180
scene.set_observation(obs)

# 模拟的输出文件
sim.save_sim_project() # 保存工程
sim.start() # 开始模拟

```

采用以上代码模拟得到的图像如图 2-44 所示, 可以看到模拟图像的分辨率提高了, 树冠的轮廓也更清晰了。

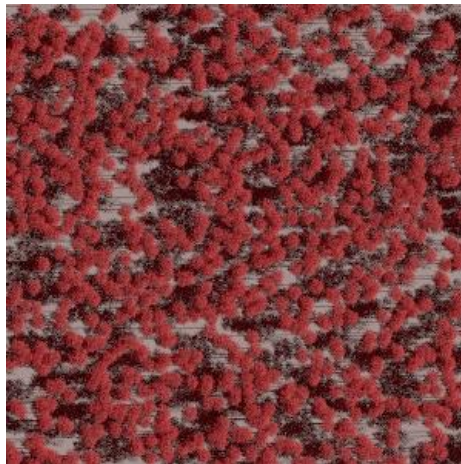


图 2-44 修改传感器参数

上述改动只是修改了默认的 SensorOrthographic 的参数 (例如图像大小), 如果需要设置其他类型的传感器 (SensorPerspective、SensorFisheye、SensorPhotonTracing), 则需要新建, 再添加到场景, SensorPerspective 示例代码如下:

```

# 添加新的传感器
sensor = SensorPerspective()
sensor.set_fov_x(40) # X 方向视场角 (单位: degree)
sensor.set_fov_y(30) # Y 方向视场角 (单位: 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)

# 添加新的观测, 与 SensorPerspective 相对应的是 ObservationPerspective

```

```

obs = ObservationPerspective()
obs.set_origin((50, 50, 150)) # 设置相机的位置
obs.set_target((50, 50, 0)) # 设置相机的观测目标，目标和原点之间的向量决定相机观测方向
scene.set_observation(obs)

```

**SensorPhotonTracing** 代表前向光线追踪，并没有实际的相机实体，因此 **SensorPhotonTracing** 对应的 **ObservationPhotonTracing** 不需要设置具体的参数，只需要构建一个空的实例即可，示例代码如下：

```

# 添加前向追踪的 sensor，模拟多角度反射率
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)

```

#### 2.4.1.3 Ex03 激光雷达模拟

激光雷达模拟与影像模拟的方式大体相同，不同在于传感器的参数有所差异。如果要启用激光雷达模拟，需要调用 `scene.enable_lidar(True)`。目前，内置的激光雷达传感器有 **ALSLiDAR** 和 **TLSLiDAR**。每个传感器有三类参数，分别为 **Beam**、**Device** 和 **Platform**。**Beam** 包括 **Axial Division** 和 **Max scattering order**，调用方式为 `liarSensor.beam.axialDivision` 和 `lidarSensor.beam.maxOrder`（`lidarSensor` 为创建的 LiDAR 对象，例如 `lidarSensor = ALSLiDAR()`）。**Device** 为仪器硬件参数，包括光斑能量等，调用方式为 `lidarSensor.device`。**Platform** 为平台参数，调用方式为 `lidarSensor.platform`。`liarSensor.set_lidar_sim_mode(LiDARSimMode.MultiRayPointCloud)` 函数设置模拟的模式，有三种 **Single ray point cloud**, **multi ray point cloud** 以及 **multi ray waveform**（与 GUI 一一对应）。需要注意的时，创建好传感器之后，需要调用 `scene.set_lidar_sensor(liarSensor)` 来设置传感器到场景，之后保存工程并进行模拟。

```

# 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") # 创建 SimulationHelper, 参数为 LESS
的安裝根目录
sim_helper.create_new_sim(sim_dir) # 新建模拟工程
sim = Simulation(sim_dir, sim_helper) # 初始化 Simulation 对象
sim.read_sim_project() # 读取模拟工程的内容
scene = sim.get_scene() # 得到 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() # 保存工程
sim.start() # 开始模拟

```

## 2.4.2 树冠生成

在 SDK for Python 中提供了一套可以生成各种形状树冠（树冠内部填充随机分布叶片）的工具。树冠的基本参数如下图所示：

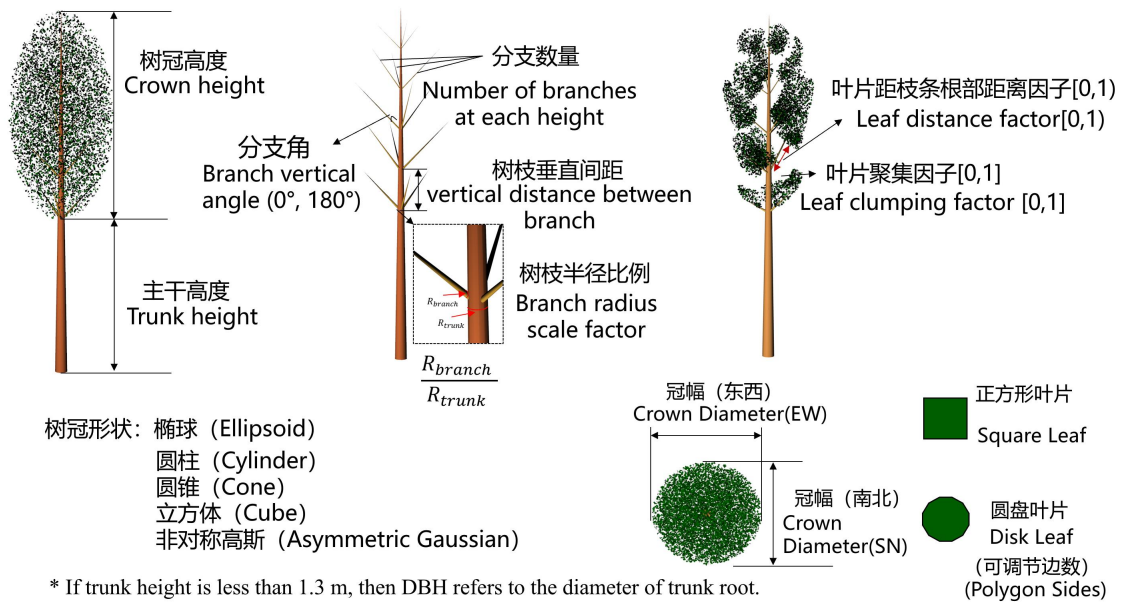


图 2-45 树冠基本参数

以下是代码示例:

```
from ObjectGenerator import CrownGenerator, CrownShape, LeafShape, LAD
cg = CrownGenerator()
cg.lvd = 10000 # 叶面积体密度
cg.crown_diameter_EW = 2 # 东西冠幅
cg.crown_diameter_SN = 2 # 南北冠幅
cg.crown_height = 4 # 树冠高度
cg.crown_shape = CrownShape.ELLIPSOID # 树冠形状
cg.leaf_angle_dist = LAD.SPHERICAL # 叶倾角分布
cg.single_leaf_area = 0.001 # 单个叶片面积
cg.leaf_shape = LeafShape.SQUARE # 叶片形状, 若为 DISK, 可设置叶片三角形个数
# (leaf_number_triangles, 即多边形边数)
cg.has_trunk = True # 是否包含主干
cg.trunk_height = 3 # 如果包含, 可设置树干高度
cg.dbh = 0.2 # 主干胸径
cg.generate_crown(r"D:\crown.obj") # 保存 obj 文件
```

2.5 GUI 界面自动化 AutoGUI

在 LESS 中，一共有两种方式来执行模拟，第一种的最常规的界面 GUI，即通过鼠标点击操作，第二种是通过 Python SDK（即 2.4 节内容）来执行批处理。GUI 一般只能执行单次模拟，想要对一些参数进行批处理一般需要通过 BatchTool 来实现（即 2.2.2 节内容来处理），BatchTool 是一个比较灵活地工具，能够对几乎界面上所有的参数进行控制。本节内容介绍在 GUI 界面上的另一个工具 AutoGUI，即通过在界面输入 Python 脚本的方式来控制参数，可以对一些参数进行控制，通过 AutoGUI 的好处是，可以不用写完整的 SDK 脚本，同时又能发挥脚本文件的灵活性（图 2-46）。

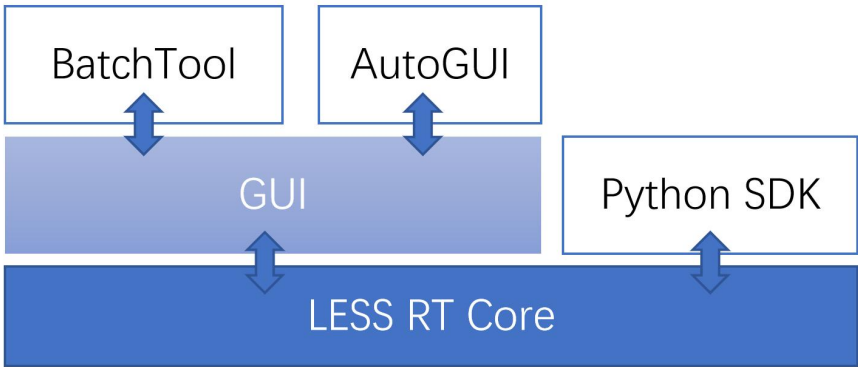


图 2-46 LESS 执行模拟的接口

在打开或者新建一个工程之后，可以在界面的左下角点击 Python Console 按钮，然后回弹出一个新的 python 命令行窗口，在此处即可以通过调用 python 脚本来修改界面的参数。默认情况下，LESS 提供了一个 less 对象，通过该对象可以实现对 LESS 界面种一些参数的控制（目前只开放了部分参数，如有需要可以添加）。例如，在窗口种输入 less.run\_all() 表示执行 LESS 模拟，其效果与手动点击界面种的【Run】【Run all】的完全一致的。目前开放的接口如下（可以通过 print(less.help())打印所有支持的函数）：

|                                                                                       |            |
|---------------------------------------------------------------------------------------|------------|
| print(less.help())                                                                    | 打印所有支持的函数  |
| run_all()                                                                             | 执行 LESS 模拟 |
| process_brf()                                                                         | 生成 BRF 图像  |
| add_op_item(String op_name, String op_ref_front, String op_ref_back, String op_trans) | 添加光谱属性     |
| del_op_item(String op_name)                                                           | 删除一条光谱属性   |
| modify_op_item(String op_name, String op_ref_front,                                   | 修改一条光谱属性   |

|                                                                                                                                                                                                                                                                        |                                                                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| String op_ref_back, String op_trans)                                                                                                                                                                                                                                   |                                                                   |
| set_sun_zenith_angle(double zenith)<br>set_sun_azimuth_angle(double azimuth)<br>set_obs_zenith_angle(double zenith)<br>set_obs_azimuth_angle(double azimuth)<br>get_sun_zenith_angle()<br>get_sun_azimuth_angle()<br>get_obs_zenith_angle()<br>get_obs_azimuth_angle() | 设置太阳和观测角度                                                         |
| add_object(String obj_name)                                                                                                                                                                                                                                            | 添加一个 object                                                       |
| add_object_component(String obj_name, String comp_path, String op_name)                                                                                                                                                                                                | 为某个 object 添加组分，并设置光谱名称                                           |
| set_object_component_optical(String obj_name, String comp_name, String op_name)                                                                                                                                                                                        | 设置某个 object 的某个组分的光谱                                              |
| place_object(String obj_name, double x, double y, double z, double rotation_degree)                                                                                                                                                                                    | 将某个 object 放置在 xyz 的位置，并设置旋转角度。注意：此处输入的 xyz 和角度需要为浮点数，即 0.0，而不是 0 |
| del_object_place(String obj_name, int index)                                                                                                                                                                                                                           | 删除某个 object 的某个位置                                                 |
| del_object(String obj_name)                                                                                                                                                                                                                                            | 删除一个 object                                                       |
| get_object_list()                                                                                                                                                                                                                                                      | 获取 object 列表                                                      |
| get_object_component_list(String obj_name)                                                                                                                                                                                                                             | 获取某个 object 的所有组分列表                                               |
| set_terrain_optical(String op_name)                                                                                                                                                                                                                                    | 设置地形的光谱                                                           |
| save()                                                                                                                                                                                                                                                                 | 保存工程，                                                             |
| run_py(String script_path)                                                                                                                                                                                                                                             | 运行一个 python 脚本，可以把上述函数写在一个脚本文件种，方便写自己的执行逻辑，也方便保存代码。               |

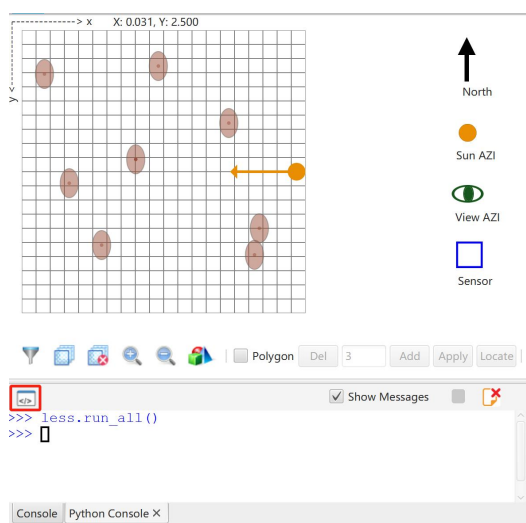


图 2-47 LESS 种 AutoGUI 模块

## 第 3 章 LESS 扩展模块

### 3.1 LESS 1D 模型

冠层辐射传输模型是植被定量遥感反演的基础，当前常用的一维模型由于具有较高的计算效率，因而得到了广泛的应用。三维模型虽然能够得到更准确的模拟结果，但是其输入参数较为复杂、操作不够便捷，因而限制了其应用。为了充分利用三维模型计算准确这一特点，本研究在一维与三维之间进行了折衷处理，发展了 LESS1D 模块，LESS1D 是 LESS 三维辐射传输模型的一维“简化版”，其主要特征是提供像一维模型的输入参数（例如叶面积指数、叶倾角分布以及株密度等），但具有三维模型的计算精度（例如无需进行热点的近似处理）。LESS1D 目前提供了 LESS-Hom, LESS-Row 以及 LESS-Forest 共三种场景模拟（通过场景自动生成，可以很容易扩展到其他更复杂的场景类型），分别表示水平均匀分布场景、垄行结构场景以及离散森林冠层场景。LESS1D 的辐射传输计算过程与 LESS 模型完全一致，即辐射传输过程完全以光线追踪的形式进行，具有较高的计算精度，因此可以说 LESS1D 具有一维模型的简洁，三维模型的精度。LESS1D 提供了易用的图形界面，支持参数批量输入以及批量模拟等，是进行大量参数组合模拟的理想工具。

打开 LESS 主程序，在【Tools】中打开【LESS1D】主界面，点击【Browse】选择模拟结果输出路径并新建一个模拟工程，也可以通过主界面的【Tools】→【Load】载入已有的模拟工程，设置完全部参数后，点击【Tools】→【Save】保存模拟参数。

### ① 基本参数设置

在【General】【wavelengths】设置模拟的光谱波长，设置的多个波长数据之间用“，”隔开。在【num\_of\_cores】中根据电脑性能设置相应处理核数。

### ② 叶片反射率模拟

为了方便使用，LESS1D 额外提供了一个 PROSPECT 模型，可以通过输入叶片参数，实现叶片尺度的反射率和透过率模拟。每个参数的数据类型有【LIST】和【RANGE】两种，【LIST】不同数据间用“/”隔开；【RANGE】采用“下限值：步长：上限值”的方式，最终模拟结果是每个参数的组合，因而可以方便地进行批处理。在【Prospect5D】中设置叶片参数，点击【Browse】设置输出目录，点击【Run】即可模拟得到叶片光谱反射率和透过率。

### ③ 冠层光谱模拟

LESS1D 提供了 LESS-Hom, LESS-Row 以及 LESS-Forest 共三种冠层场景，分别表示水平均匀分布场景、垄行结构场景以及离散森林冠层场景。

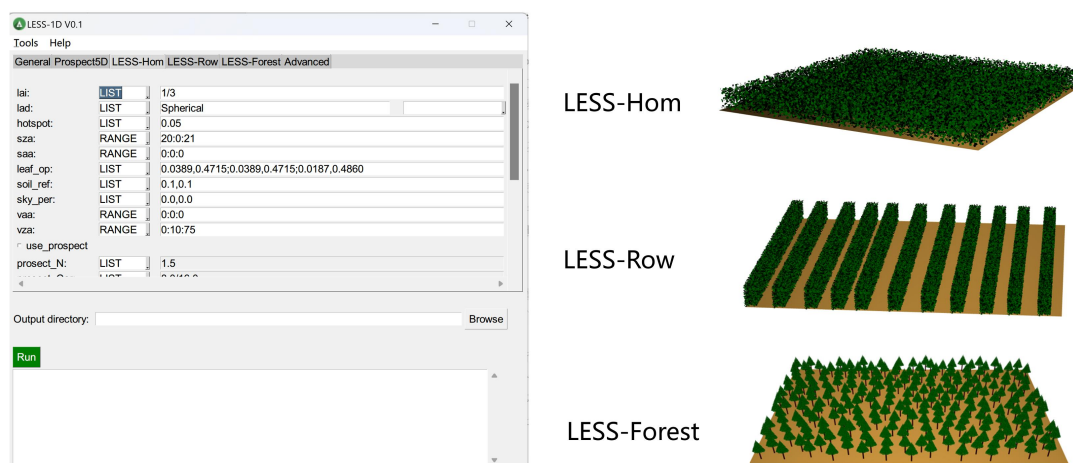


图 3-1 LESS1D 主界面及树冠基本参数

### (1) 水平均匀场景【LESS-Hom】

| 参数  | 说明                                      |
|-----|-----------------------------------------|
| lai | LESS-Hom 的 lai 是场景 LAI 值。               |
| lad | 叶倾角分布类型：Erectophile 竖直型、Exemophile 极端型、 |

|                  |                                                                                                |
|------------------|------------------------------------------------------------------------------------------------|
|                  | Plagiophile 倾斜型、Planophile 平面型；Spherical 球面型、Uniform 统一型。                                      |
| hotspot          | 热点计算公式为叶片长度/冠高。                                                                                |
| leaf_op          | 叶片光谱。当勾选“ <input checked="" type="checkbox"/> use_prospect”后，LESS1D 将根据 prospect 参数自动计算叶片光谱参数。 |
| soil_op          | 土壤光谱（波段数量需要与植被等一致）                                                                             |
| sky_per          | 天空光比例（波段数量需要与土壤以及植被等一致）                                                                        |
| sza/saa, vza/vaa | 太阳天顶角/方位角，观测天顶角/方位角                                                                            |

## （2）垄行结构场景【LESS-Row】

| 参数            | 说明  |
|---------------|-----|
| row_width     | 垄宽  |
| plant_height  | 垄高  |
| ridge_spacing | 垄间距 |

## （3）垄行结构场景【LESS-Forest】

| 参数              | 说明                                                 |
|-----------------|----------------------------------------------------|
| lai_single_tree | LESS-Forest 的 lai 是单株 LAI 值。                       |
| crown_shape     | 树冠冠形：Ellipsoid 椭球形、Cylinder 圆柱形、Corn 圆锥形、Cube 立方形。 |
| stem_density    | 株密度，即场景中树木株数。（场景面积 100*100m）                       |
| tree_height     | 树高                                                 |
| crown_diameter  | 冠幅                                                 |
| crown_length    | 冠长                                                 |
| dbh             | 胸径                                                 |

## ④【Advanced】设置

未勾选【has\_fpar】，模拟结果仅输出 BRF 数值；勾选【has\_fpar】，模拟结果同时

输出 BRF 和 FPAR 数值。

在【fpar\_layers】设置冠层,表示对场景进行分层,以得到每一层的 fPAR。设置格式为“起始高度:步长:结束高度”。

在【sun\_spec】和【sky\_spec】中分别输入场景的太阳直射辐射数值和散射辐射数值。

⑤ 模拟结果

参数设置完毕,在【Browse】设置存储与输出路径,点击所模拟场景页面左下角的【Run】进行模拟,模拟结束后在设置路径的【Result】中查看结果。

(1) BRF 结果

在【Result】中打开【less\_BRF\_info】txt 文本可查看输入参数。BRF 模拟结果数据存储在【less\_BRF】的 txt 文本中。在本练习中设置了三个波段(600nm, 700nm, 800nm),观测角度范围为“-75: 2: 75”。由图可得在不同观测角度下各波段的 BRF 值。

(2) FPAR 结果

在【Result】中打开【less\_FPAR】txt 文本可查看 FPAR 模拟结果。

|                               |                                    |
|-------------------------------|------------------------------------|
| 模拟结果                          | 说明                                 |
| TfPAR                         | 表示场景总 FPAR 值, 包括叶片总 FPAR 和土壤总 FPAR |
| Unnamed (组分名称)                | 表示叶片总 FPAR                         |
| terrain                       | 表示土壤总吸收                            |
| Total Absorption              | 表示场景各波段 FPAR 值                     |
| Absorption of unnamed/terrain | 表示叶片/土壤各波段的 FPAR 值                 |

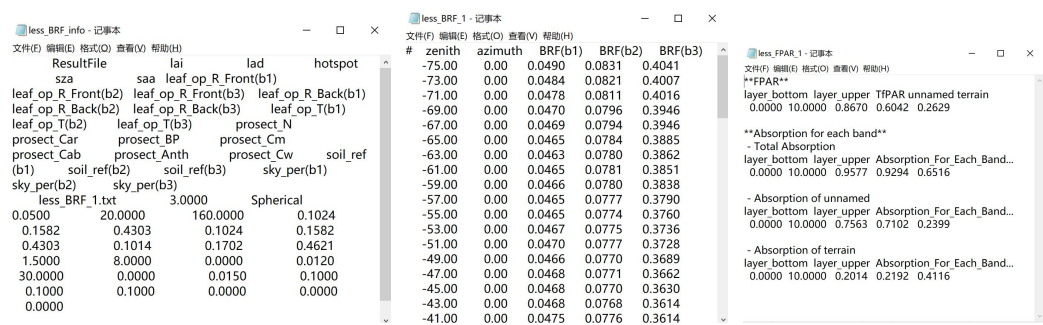


图 3-2 LESS1D 模拟结果

## 3.2 LESS Plugins

### 3.2.1 插件的使用

LESS 提供了一个开发插件的模块,通过该模块可以快速的基于 LESS 开发具有图形界面的插件, LESS 插件可以方便地集成第三方应用,例如在 LESS 模拟结果上进行进一步的处理、构建复杂的三维场景、提取 LESS 场景结构参数等,从而方便用户的使用。LESS 插件位于【Tools】【LESS Plugins】,打开插件之后,会弹出一个插件对话框,如下图所示。该对话框左侧列出了所有的插件,右侧为该插件的描述信息,以及作者信息。选中一个插件之后,可以点击【Run Plugin...】进入插件的每个插件的详细界面,该界面展示了插件的所有参数,根据需要进行相关测试,然后点击 Run,则可以运行该插件。

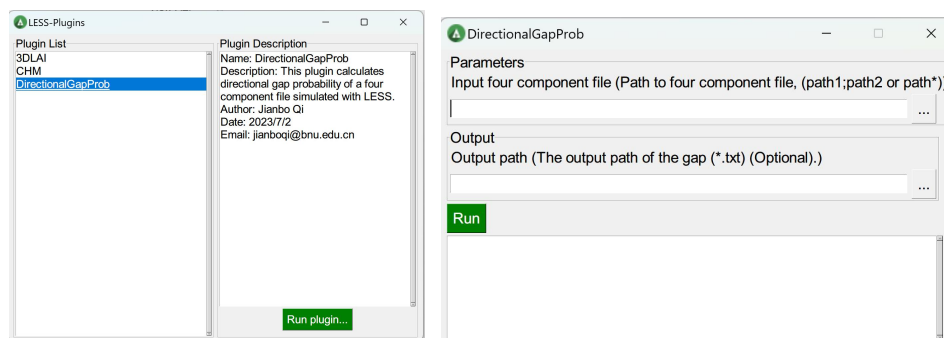


图 3-3 LESS 插件使用

### 3.2.2 插件的开发

在 LESS 中开发插件十分容易,无需处理 GUI 相关问题,只需要写数据处理逻辑部分的代码即可。首先打开 LESS 插件所在的目录,通常位于安装目录下(例如 D:\LESS\app\Python\_script\LESSPlugins),在该目录下有一个文件夹 StandardPlugins,该文件夹下保存了所有的插件,每个插件为一个文件夹,插件名即为文件夹名。如过要创建一个名为 Demo 的 Plugin,则在 StandardPlugins 文件夹下首先创建一个文件夹 Demo,进入 Demo 文件夹,创建名称分别为 info.json 和 Demo.py 的两个文件,文件内容可以直接从已有的其他插件中复制。info.json 名称固定,不可修改,内容如下图所示



图 3-4 LESS 插件使用

程序文件（**Demo.py**），文件名与插件名一致，只需要实现一个 **run** 函数即可，所有的计算逻辑均位于 **run** 函数中，具体可以参考 **APluginDevDemo** 插件或者已有的插件。

### 3.2.3 批量调用插件

如果需要反复调用插件以进行批处理，LESS 提供了 **Python** 接口，只需要将 **LESSPlugins** 加入到路径即可，具体可以参考以下代码：

```

# This script demonstrate how to invoke the plugins from a python script, in order
to do batch processing

# If you write a code outside this LESSPlugins folder, please add LESSPlugins
path as sys path:

# sys.path.add([Path to LESSPlugins]) # usually in LESS installation folder,
e.g.,

# import sys

# sys.path.append(r"D:\LESS\app\Python_script\LESSPlugins")

from StandardPlugins import StandardPlugins

# this LESS simulation directory or None for plugins that do not need a simulation
folder

curr_dir = r"G:\LESS\simulations\TESTS\testLAICal"

std_plugins = StandardPlugins(curr_dir)

# print(std_plugins.get_plugin_names()) # list all the available plugins

plugin = std_plugins.plugin_dict["3DLAI"] # query a plugin for running

# preparing running params according to the plugin parameters required

# you can refer to the plugin folder or print plugin.plugin_params

```

```
run_params = {
    "Grid Size": {
        "rows": 1,
        "cols": 1,
        "layers": 1
    },
    "Computation extents": {
        "width": 100,
        "height": 100
    },
    "Components": {
        "Hide components": ""
    },
    "Output": {
        "Output path": r""
    }
}

# The second parameter window is only used for GUI, here you can set to None
plugin.run(run_params, window=None)
```

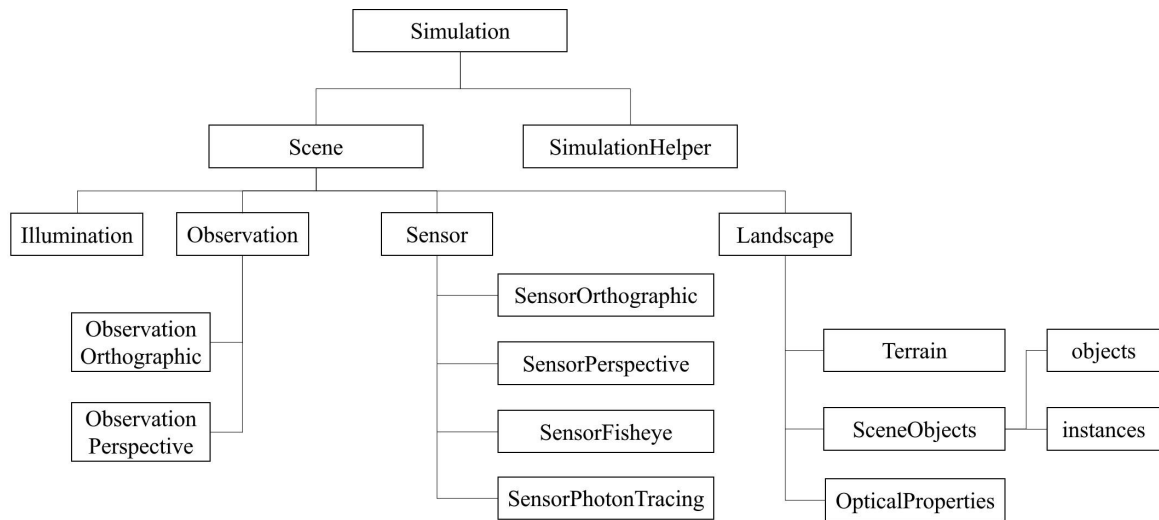
## 参考文献

Automatic citation updates are disabled. To see the bibliography, click Refresh in the Zotero tab.

## 附录

### 附录 1 SDK for Python 文档

pyLessSDK 主要的类如附图 1 所示，Simulation 包含了 Scene 和 SimulationHelper 两个类，Scene 是整个场景的主类，包含了场景的中各个要素，例如观测 Observation、传感器 Sensor 等。每个类的主要函数及功能如下文所示。



附图 1 pyLessSDK 主要类图

LESS SDK PAI 文档地址：[https://lessrt.org/sdk\\_api\\_doc/index.html](https://lessrt.org/sdk_api_doc/index.html)

#### SimulationHelper

- `SimulationHelper(less_install_dir)`: 构造函数，传入 LESS 的安装目录，例如 D:\LESS
- `create_new_sim(new_sim_path)`: 创建新的模拟工程，参数 new\_sim\_path 为模拟工程的目录，如果是空目录，则创建该模拟工程；如果该目录已经是模拟工程了，则跳过创建。

#### Simulation

- `Simulation(sim_dir, simulation_helper)`: 构造函数，传入刚才创建的模拟工程目录 sim\_dir，以及 SimulationHelper 的实例 simulation\_helper。
- `set_dist_file(dist_file)`: 设置输出文件的路径（例如：d:\test01\results\output\_radiance）
- `get_dist_file()`: 获取输出文件的路径

- `get_scene()`: 获取场景实例
- `get_sim_dir()`: 获取模拟工程的目录
- `read_sim_project`: 读取模拟工程参数
- `save_sim_project ()`: 保存模拟工程
- `start()`: 开始模拟

## Scene

- `set_sensor(sensor)`: 设置传感器 `sensor`
- `get_sensor ()`: 获取场景的 `sensor`
- `set_illumination (illumination)`: 设置场景 `Illumination`
- `get_illumination ()`: 获取场景 `Illumination`
- `set_observation(obs)`: 设置场景的观测参数
- `get_observation ()`: 获取场景观测参数

## Landscape

- `add_op_item (op_item)`: 在光谱数据库中添加一条光谱
- `get_op_item (op_name)`: 根据光谱名称获取一条光谱
- `add_object (scene_obj)`: 添加一个场景物体 `SceneObject`
- `place_object (obj_name, x=50.0, y=50.0, z=0.0, rotate=0.0)`: 将物体 `obj_name` 放置在位置 `(x,y,z)`处, 并设置其旋转角度。
- `set_terrain_op(op_name)`: 设置地形对应的光谱曲线
- `get_terrain()`: 获取场景地形 `Terrain`
- `clear_landscape_elements()`: 清空场景中的所有元素, 包括加入的物体以及他们的位置
- `clear_user_defined_optical_properties()`: 清空场景中用户自定义的光谱曲线

## Sensor (所有 Sensor 的基类)

- `set_spectral_bands(spectral_bands)`: 设置模拟的波段, 例如三个波段”600:2,700:2,800:2”
- `set_repetitive_scene(num)`: 设置场景重复次数 `num`
- `get_repetitive_scene()`: 获取场景重复次数

## SensorOrthographic

- `set_sample_per_pixel(num)`: 设置每像素的光线条数
- `get_sample_per_pixel()`: 获取每像素的光线条数
- `set_image_width(image_width)`: 设置图像宽度（像素）
- `set_image_height(image_height)`: 设置图像高度（像素）
- `set_film_type(film_type)`: 设置图像保存类型: `rgb` 或者 `spectrum`
- `set_sub_region_width(width)`: 设置相机观测范围, 例如 100m 场景只观测中心 50m
- `set_sub_region_height(height)`: 设置相机观测范围

## SensorPerspective

- `set_fov_x(fov_x)`: 设置 X 方向的视场角, 单位: `degree`
- `set_fov_y(fov_y)`: 设置 Y 方向的视场角, 单位: `degree`
- `set_sample_per_pixel(num)`: 设置每像素的光线条数
- `get_sample_per_pixel()`: 获取每像素的光线条数

## SensorFisheye

- `set_angular_fov(fov)`: 设置鱼眼相机的视场角, 例如  $170^\circ$
- `set_projection_type(projection_type)`: 设置投影类型, 例如 `equisolid`
- `set_sample_per_pixel(num)`: 设置每像素的光线条数
- `get_sample_per_pixel()`: 获取每像素的光线条数

## SensorPhotonTracing

- `set_sun_ray_resolution(self, sun_ray_resolution)`: 设置入射光线的密度, 以最小间距度量, 例如 0.02 表示每 2cm 一条光线。
- `enable_brdf_product(enabled)`: 是否启用 BRDF 模拟, 如果 `enabled=True`, 则可使用以下函数:
  - `set_number_of_directions(num)`: 设置获取 BRDF 的半球空间的方向数量
  - `virtual_detector_directions(virtual_detectors)`: 设置虚拟观测器
  - `virtual_directions(virtual_directions)`: 虚拟观测方向, 模拟 BRDF 时主要使用的方式
- `enable_updown_radiation_product(enabled)`: 是否启用上下行辐射模拟
- `enable_fpar_product(enabled)`: 是否启用 `fpar` 的模拟, 如果启用
  - `set_fpar_layer(self, fpar_layer)`: 设置 `fpar` 分层。

## ObservationOrthographic

- `set_obs_azimuth(azi)`: 设置观测方位角
- `set_obs_zenith (zenith)`: 设置观测天顶角

## ObservationPerspective

- `set_origin (origin)`: 设置相机位置, (x,y,z)
- `set_target (zenith)`: 设置相机观测目标, (x,y,z)
- `enable_relative_height(enabled)`: 是否开启相对高度, 如果开启则 z 坐标为相对于地形的高度, 而不是绝对高度。

## ObservationFisheye

- `set_origin (origin)`: 设置相机位置, (x,y,z)
- `set_target (zenith)`: 设置相机观测目标, (x,y,z)
- `enable_relative_height(enabled)`: 是否开启相对高度, 如果开启则 z 坐标为相对于地形的高度, 而不是绝对高度。

## ObservationPhotonTracing

继承自 ObservationOrthographic, 使用时保持默认参数即可, 无须修改参数。