武汉大学
WUHAN UNIVERSITY

武汉大学遥感信息工程学院
School of Remote Sensing and Information Engineering, Wuhan University

笃志 敦行 和协 拓新

本科课程论文（设计）

——2024~2025 学年第二学期遥感原理与应用课程设计实习报告

应用 ERDAS、开放地球引擎服务平台 OGE 与 Python GDAL 库的《遥感原理与应用课程设计》课程设计实习报告

姓　　名：	杨 丹 阳
学　　号：	2023302131259
学　　校：	武汉大学
学　　院：	遥感信息工程学院
专　　业：	遥感科学与技术
课　　程：	遥感原理与应用课程设计
指导教师：	杨 代 琴

二〇二五 年 六 月

原创性声明

本人郑重声明：所呈交的论文（设计），是本人在指导教师的指导下，严格按照学校和学院有关规定完成的。除文中已经标明引用的内容外，本论文（设计）不包含任何其他个人或集体已发表及撰写的研究成果。对本论文（设计）做出贡献的个人和集体，均已在文中以明确方式标明。本人承诺在论文（设计）工作过程中没有伪造数据等行为。若在本论文（设计）中有侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

作者签名：杨丹阳

日　　期：2025 年 6 月 7 日

版权使用授权书

本人完全了解武汉大学有权保留并向有关部门或机构送交本论文（设计）的复印件和电子版，允许本论文（设计）被查阅和借阅。本人授权武汉大学将本论文的全部或部分内容编入有关数据进行检索和传播，可以采用影印、缩印或扫描等复制手段保存和汇编本论文（设计）。

作者签名：杨丹阳

日　　期：2025 年 6 月 7 日

摘　要

本文是武汉大学遥感信息工程学院遥感原理与应用课程设计的实习报告。本次实习内容包括应用 ERDAS 软件进行遥感分类专题信息提取与制图、基于开放地球引擎服务平台（OGE）的特征指数遥感专题信息提取，以及使用 Python GDAL 库编程实现遥感影像镶嵌。通过 ERDAS 软件，深入探讨了监督分类和非监督分类方法在地物分类中的应用效果，并完成了几何校正、影像融合、分类精度评定和专题制图等任务。在 OGE 平台上，研究了植被指数、水体指数和建筑指数等特征指数的提取技术，能够快速准确地提取专题信息。此外，通过 Python GDAL 编程，实现了遥感影像的镶嵌技术，探索了先进的镶嵌算法，优化了镶嵌效果。本研究旨在提高遥感数据的应用价值和分析精度，为相关领域的科学研究和实际应用提供技术支持。

关键词：遥感原理与应用；ERDAS 软件；开放地球引擎服务平台（OGE）；Python GDAL 库；遥感影像镶嵌；地物分类；特征指数提取；专题制图

ABSTRACT

This paper is an internship report for the Remote Sensing Principles and Applications course design practice at the School of Remote Sensing and Information Engineering, Wuhan University. The internship content includes applying ERDAS software for remote sensing classification and thematic information extraction and mapping, feature index-based remote sensing thematic information extraction using the Open Geospatial Engine (OGE) service platform, and programming implementation of remote sensing image mosaicking using Python GDAL library. Through ERDAS software, the application effects of supervised classification and unsupervised classification methods in land cover classification were thoroughly explored, and tasks such as geometric correction, image fusion, classification accuracy assessment, and thematic mapping were completed. On the OGE platform, extraction techniques for feature indices including vegetation index, water body index, and building index were studied, enabling rapid and accurate extraction of thematic information. Additionally, through Python GDAL programming, remote sensing image mosaicking technology was implemented, advanced mosaicking algorithms were explored, and mosaicking effects were optimized. This research aims to improve the application value and analysis accuracy of remote sensing data, providing technical support for scientific research and practical applications in related fields.

Keywords: Remote Sensing Principles and Applications; ERDAS Software; Open Geospatial Engine (OGE) Platform; Python GDAL Library; Remote Sensing Image Mosaicking; Land Cover Classification; Feature Index Extraction; Thematic Mapping

目 录

0 写在前面

我知道这篇实习报告确实写得非常长，但是是基于“把问题解释清楚”的目的写的，每一个字都是我认认真真写的，没有水字数，还请老师能够看完🥺

如果有需要的话，您可以让我线下向您简要解释报告的内容，以帮助您更快速地了解报告各部分的主要内容😊

1 绪论

随着遥感科学与技术的飞速发展，其在地球科学、环境监测、资源管理、城市规划等诸多领域的应用日益广泛[1]。遥感技术能够快速、高效地获取大范围地表信息，为科学研究和实际应用提供了丰富的数据支持[2]。然而，如何从海量的遥感数据中提取出有价值的信息，一直是遥感应用研究的核心问题之一[3]。本课程设计旨在通过综合应用多种遥感数据处理技术，加深和巩固对《遥感原理与方法》理论课程中的基本理论与方法的理解，培养运用遥感软件、开放地球引擎云平台 OGE 与编程技术来分析、解决问题的能力，深入探索遥感影像的分类、专题信息提取以及镶嵌等遥感影像处理关键技术的原理与实现，以提高遥感数据的应用价值和分析精度，自主建立遥感模型，最终培养解决遥感实际应用问题的能力。

在遥感影像分类方面，通过应用 ERDAS IMAGINE 软件，本课程设计将深入探讨监督分类和非监督分类方法在地物分类中的应用效果。监督分类方法依赖于已知类别样本的光谱特征，通过建立分类模型来识别和区分地表覆盖类型；而非监督分类方法则通过分析影像数据的内在结构和分布，自动将像素划分为不同的类别[4][5]。通过比较不同分类方法的精度和适用性，本研究将为土地利用、环境监测和资源管理等领域提供科学的分类技术支持[6]。

在遥感专题信息提取方面，本课程设计将借助开放地球引擎服务平台（OGE）深入研究基于特征指数的遥感专题信息提取技术。特征指数，如植被指数、水体指数和建筑指数，是基于地物光谱反射特性差异而构建的数学表达式，能够有效增强特定地物信息，抑制其他地物干扰[7][8]。通过计算和分析这些特征指数，本研究将能够快速、准确地提取出植被覆盖度、水体分布和建筑用地等专题信息，为生态环境保护、水资源管理和城市规划等提供重要的地理信息支持[9][10]。

在遥感影像镶嵌方面，本课程设计将通过 Python GDAL 库编程实现遥感影像的镶嵌技术[11][12]。影像镶嵌是将多幅具有重叠区域的遥感影像拼接成一幅完整、连续影像的

技术过程，对于扩大观测范围、提高空间覆盖度具有重要意义。本研究将深入研究几何校正与配准、辐射校正与色彩平衡、重叠区域处理等关键技术环节[13][14]，探索先进的镶嵌算法，以实现高质量的影像镶嵌效果，为大范围地理信息获取提供基础数据支撑[15]。

综上所述，本研究将通过系统地应用 ERDAS IMAGINE 软件、OGE 开放地球引擎服务平台和 Python GDAL 库编程，深入探索遥感影像的分类、专题信息提取和镶嵌等关键技术，旨在提高遥感数据的应用价值和分析精度，为相关领域的科学研究和实际应用提供有力的技术支持，为遥感技术的发展和应用提供新的思路和方法，推动遥感技术在更多领域的更广泛应用。

图1-1 应用ERDAS、开放地球引擎服务平台OGE与Python GDAL库的《遥感原理与应用课程设计》课程设计实习内容示意图

2 应用 ERDAS 软件进行遥感分类专题信息提取与制图

2.1 实习目的

本次实习中，应用 ERDAS 软件进行遥感分类专题信息提取与制图部分的目的旨在通过熟练掌握 ERDAS IMAGINE 软件中常用功能的操作方法，并应用 ERDAS IMAGINE 软件完成遥感分类专题信息提取与制图的全流程任务，深入理解并掌握遥感图像处理的基本流程和关键技术。

本部分具体包括以下任务目标：通过波段叠加，将多波段影像集成到一个数据集中，

便于后续分析；通过几何校正，消除影像的几何变形，使其与地理坐标系统一致，为精确的地理信息分析提供基础；将不同分辨率或不同传感器的影像数据综合处理，生成兼具高空间分辨率和高光谱信息的新影像；掌握并理解遥感影像分类方法，包括监督分类和非监督分类，能够识别和区分地表覆盖类型，为土地利用、环境监测和资源管理等提供重要地理信息支持；通过分类精度评定，评估分类方法的准确性和可靠性，选择合适的精度评定指标，为后续研究和应用提供依据；通过专题制图，将分类结果或其他地理信息以地图的形式直观展示，增强地图的可读性和美观性，为城市规划、环境保护、土地利用研究等提供决策支持。

2.2 实习基本原理

本部分实习“应用 ERDAS 软件进行遥感分类专题信息提取与制图”的总体流程如图 2.2-1 所示：

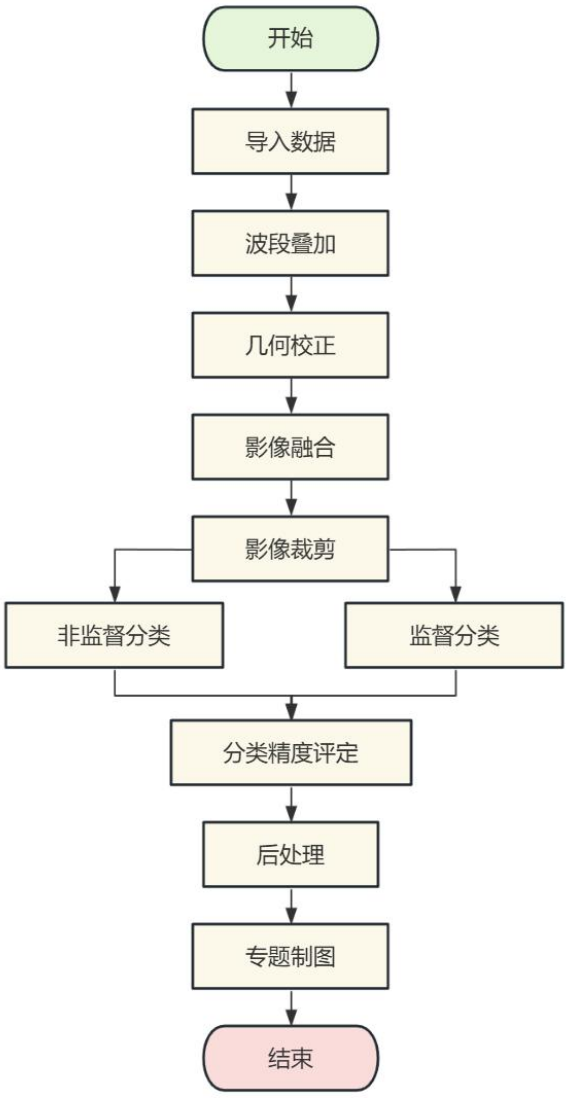


图2.2-1 应用ERDAS软件进行遥感分类专题信息提取与制图的总体流程图

2.2.1 波段叠加的基本原理

波段叠加（Band Stacking）是指将多个单波段影像（或多波段影像的单个波段）按照一定的顺序排列，形成一个多波段影像的过程。每个波段代表了影像在特定波长范围内的反射或辐射信息。通过波段叠加，可以将不同波长范围的信息集成到一个影像中，便于后续的分析和处理。

假设有n个单波段影像$B_1, B_2, \cdots, B_n$，每个波段影像的大小均为为 M×N（即 M 行和 N 列），那么波段叠加的过程可以表示为：

$$B_{stacked} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix}$$

需要注意的是，波段叠加的顺序会影响后续的分析和处理。例如，在 ERDAS IMAGINE 软件中显示时，选择哪几个波段进行 RGB 显示决定了生成的彩色影像的颜色表现；将波段叠加后的影像用于遥感指数的提取时，相应的波段代号更需要严格对应才能得到正确的结果；等等。建议严格按照拍摄影像的卫星设计的波段顺序来进行波段叠加。

通过波段叠加，可以将多个波段的信息集成到一个影像中，为后续的遥感影像分析提供了更丰富的数据基础。

2.2.2 遥感影像几何校正的基本原理

遥感影像几何校正是通过数学模型将影像中的像素坐标转换为地理坐标，消除影像中的几何变形。几何校正的目的是使影像与地图或其他地理数据在空间位置上保持一致。几何校正通常包括以下步骤：

1.选择控制点

在影像和参考数据（如地图或高分辨率影像）上选择若干对同名点，这些点的坐标在参考数据中是已知的。

2.建立几何模型

根据控制点的坐标，建立几何模型。常用的几何模型包括多项式模型、仿射变换模型和投影变换模型。多项式模型的表达式为：

$$\begin{cases} X = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 + a_5 y^2 + \cdots \\ Y = b_0 + b_1 x + b_2 y + b_3 xy + b_4 x^2 + b_5 y^2 + \cdots \end{cases}$$

其中，(X, Y)是地理坐标，(x, y)是影像坐标，a_i和b_i是多项式系数。

3.计算变换参数

将几何模型应用于整个影像，生成校正后的影像。

2.2.3 遥感影像融合的基本原理

遥感影像融合是将不同分辨率或不同传感器的影像数据进行综合处理，生成兼具高空间分辨率和高光谱信息的新影像。常用的影像融合方法包括基于光谱减法的融合方法、基于小波变换的融合方法和基于傅里叶变换的融合方法等。

基于光谱减法的融合方法（Subtractive Resolution Merge）通过从高分辨率影像中提取高频空间信息，并将其注入低分辨率的多光谱影像中。其数学表达式为：

$$F_{merged} = F_{MS} + \alpha(F_{PAN} - F_{MS})$$

其中，F_{merged}是融合后的影像，F_{MS}是多光谱影像，F_{PAN}是高分辨率全色影像，α是融合系数。

基于小波变换的融合方法（Wavelet Resolution Merge）对高分辨率影像和低分辨率影像进行小波分解，在不同频率子带上选择性融合，最后重构影像。其数学表达式为：

$$F_{merged} = W^{-1}(W(F_{PAN}) + W(F_{MS}))$$

其中，W和W^{-1}分别是小波变换和逆小波变换。

基于傅里叶变换的融合方法（Ehlers Fusion）通过频域分离光谱和空间信息减少失真。其数学表达式为：

$$F_{merged} = \Phi^{-1}(\Phi(F_{PAN}) \cdot \Phi(F_{MS}))$$

其中，Φ和Φ^{-1}分别是傅里叶变换和逆傅里叶变换。

2.2.4 基于遥感影像进行地物分类的基本原理

遥感影像分类是通过分析影像数据中的光谱信息、空间信息和纹理信息，将影像中的像素或区域划分为不同的类别。分类的目的是识别和区分地表覆盖类型（如森林、农田、城市等），从而为土地利用、环境监测和资源管理等提供重要的地理信息支持。遥感影像分类主要分为监督分类和非监督分类两种方法。

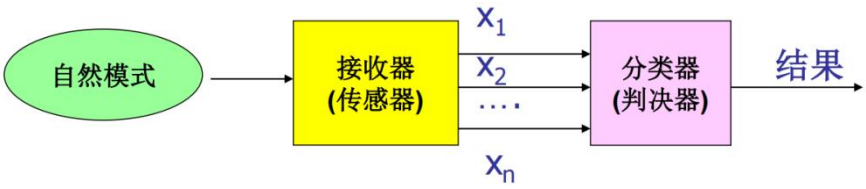


图2.2.4-1 基于遥感影像进行地物分类的模式识别系统的模型

2.2.4.1 非监督分类

非监督分类是一种不需要已知类别样本的分类方法。它通过分析影像数据的内在结构和分布，自动将像素划分为不同的类别。常用的非监督分类方法包括 K-Means 算法和 ISODATA 算法。

2.2.4.1.1 K-Means 算法

K-Means 算法通过迭代优化的方式，将像素划分为 k 个类别。其目标函数为：

$$\min_{\{C_i\}} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2$$

其中，μ_i是类别C_i的均值向量。

K-Means 算法的具体流程如图 2.2.4.1.1-1、图 2.2.4.1.1-2 所示：

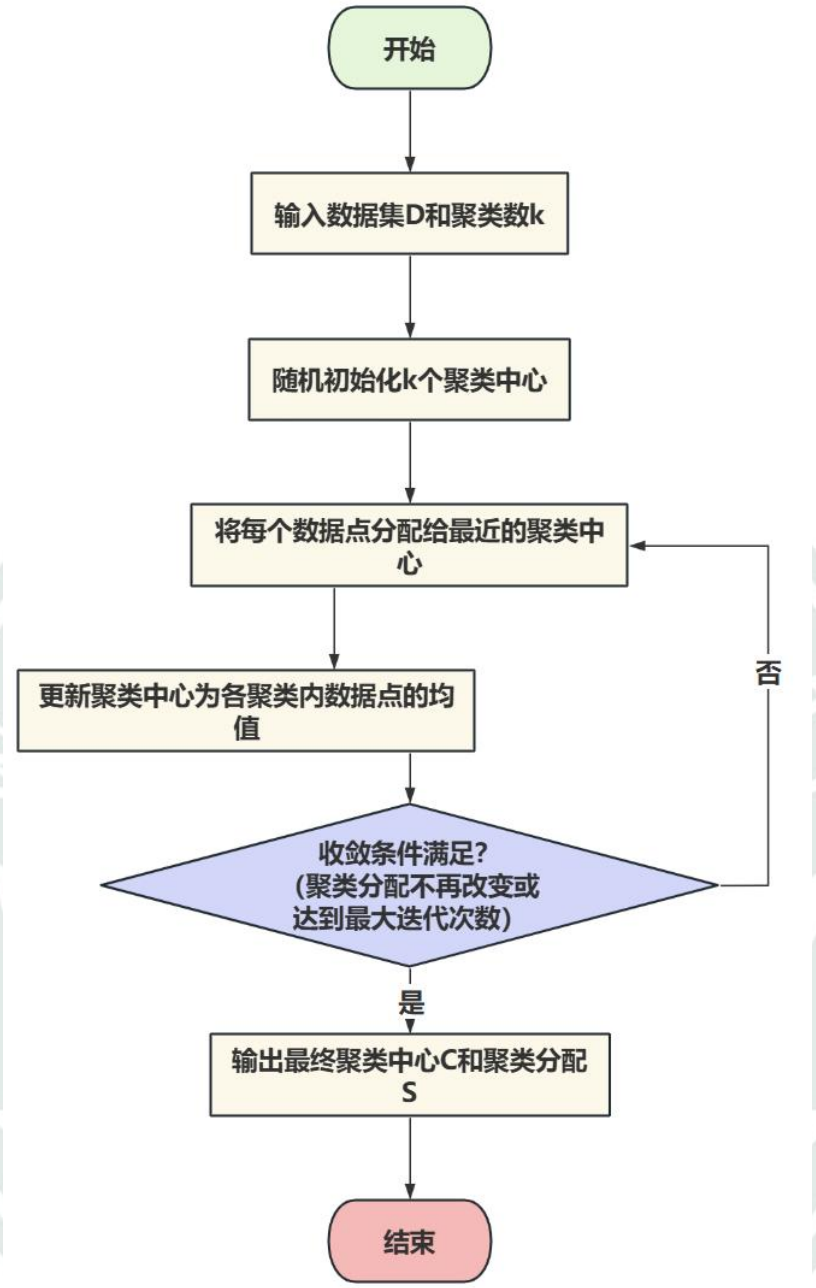


图2.2.4.1.1-1 K-Means 算法的流程图

Algorithm 1 K-Means 聚类算法

输入： 数据集 $D = \{x_1, x_2, \ldots, x_n\}$，聚类数 k

输出： 聚类中心 $C = \{c_1, c_2, \ldots, c_k\}$，聚类分配 $S = \{S_1, S_2, \ldots, S_k\}$

随机初始化 k 个聚类中心：$C^{(0)} = \{c_1^{(0)}, c_2^{(0)}, \ldots, c_k^{(0)}\}$
重复
  将每个数据点分配给最近的聚类中心：
  对于 $i = 1$ 到 n **执行**
    $S_j^{(t)} \leftarrow \arg\min_{1 \le j \le k} \|x_i - c_j^{(t-1)}\|$
  结束循环
  更新聚类中心：
  对于 $j = 1$ 到 k **执行**
    $c_j^{(t)} \leftarrow \frac{1}{|S_j^{(t)}|} \sum_{x \in S_j^{(t)}} x$
  结束循环
  $t \leftarrow t + 1$
直到 满足收敛条件（聚类分配不再改变或达到最大迭代次数）
返回 $C^{(t)}$, $S^{(t)}$

图2.2.4.1.1-2 K-Means 算法的伪代码

2.2.4.1.2 ISODATA 算法

ISODATA 算法是 K-Means 算法的一种改进算法，在 K-Means 算法的基础上增加了允许进行类别的分裂、合并与消失的规则。其目标是通过动态调整类别数量和类别中心，优化分类结果。

ISODATA 算法的具体流程如图 2.2.4.1.2-1、图 2.2.4.2.1-2 所示：

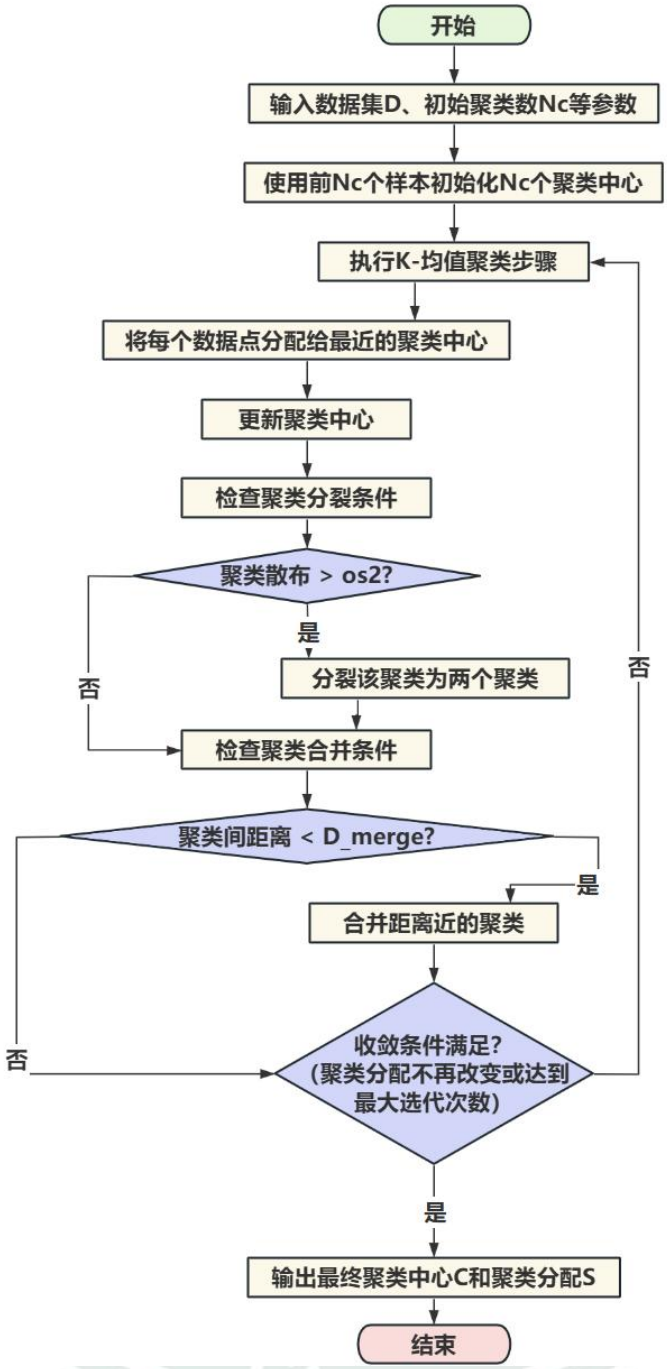


图2.2.4.1.2-1 ISODATA 算法的流程图

Algorithm 2 ISODATA 聚类算法

输入： 数据集 $D = \{x_1, x_2, \ldots, x_n\}$，初始聚类数 N_c，每个聚类的最小样本数 N_{min}，分裂的最大散布参数 $os2$，合并的最大距离分离 D_{merge}，可合并的最大聚类数 N_{merge}

输出： 最终聚类中心 $C = \{c_1, c_2, \ldots, c_k\}$，聚类分配 $S = \{S_1, S_2, \ldots, S_k\}$

使用前 N_c 个样本初始化 N_c 个聚类中心：$C^{(0)} = \{c_1^{(0)}, c_2^{(0)}, \ldots, c_{N_c}^{(0)}\}$
重复
  执行 K-均值聚类：
  对于 $i = 1$ 到 n **执行**
    $S_j^{(t)} \leftarrow \arg\min_{1 \le j \le N_c} \|x_i - c_j^{(t-1)}\|$
  结束循环
  更新聚类中心：
  对于 $j = 1$ 到 N_c **执行**
    $c_j^{(t)} \leftarrow \frac{1}{|S_j^{(t)}|} \sum_{x \in S_j^{(t)}} x$
  结束循环
  分裂样本差异足够大的聚类：
  对于 $j = 1$ 到 N_c **执行**
    如果 聚类 j 的散布 $> os2$ **那么**
      将聚类 j 分裂为两个聚类
    结束如果
  结束循环
  合并距离足够近的聚类：
  对于 $j = 1$ 到 N_c **执行**
    对于 $k = j + 1$ 到 N_c **执行**
      如果 聚类 j 和聚类 k 之间的距离 $< D_{merge}$ **那么**
        合并聚类 j 和聚类 k
      结束如果
    结束循环
  结束循环
  $t \leftarrow t + 1$
直到 满足收敛条件（聚类分配不再改变或达到最大迭代次数）
返回 $C^{(t)}$, $S^{(t)}$

图2.2.4.1.2-2 ISODATA 算法的伪代码

2.2.4.2 监督分类

监督分类是一种基于已知类别样本的分类方法。在监督分类中，首先需要从影像中选取一定数量的训练样本，这些样本的类别是已知的。然后，通过统计分析这些训练样本的光谱特征，建立分类模型。本次实习中作者试验的监督分类方法包括最大似然分类

法（Maximum Likelihood Classifier, MLC）、平行六面体法（Parallelepiped）和特征空间法（Feature Space）。

2.2.4.2.1 最大似然分类法（MLC）

最大似然分类法（MLC）的原理是：假设每个类别的光谱数据服从多元正态分布，通过计算像元属于各类别的概率，选择概率最高的类别作为分类结果。其数学表达式为：

$$P(C_i|x) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)}$$

其中，$P(C_i|x)$是像元x属于类别C_i的概率，μ_i是类别C_i的均值向量，Σ_i是类别C_i的协方差矩阵，d是波段数。

2.2.4.2.2 平行六面体法（Parallelepiped）

平行六面体法（Parallelepiped）的原理是：通过定义多维的“盒子”来分类，若像元的光谱值落在某类别的盒子内则被归为该类别。其数学表达式为：

$$\text{Classify}(x) = \begin{cases} C_i, \text{if } L_{i,j} \le x_j \le U_{i,j} \text{ for all } j \\ \text{Unassigned, otherwise} \end{cases}$$

其中，$L_{i,j}$和$U_{i,j}$分别是类别C_i在第j个波段的下界和上界。

2.2.4.2.3 特征空间法（Feature Space）

同名地物点在不同波段图像中亮度的观测量构成一个多维的随机向量$\boldsymbol{X}$，称为光谱特征向量：

$$\boldsymbol{X} = [x_1, x_2, \cdots, x_n]^T$$

其中，n是遥感影像的总波段数，x_i是地物图像点在第i波段图像中的亮度值。

特征空间是光谱特征向量所在的、用来描述和区分不同地物类型的多维数学空间，其中每个维度代表一个特征参数。

使用特征空间法（Feature Space）进行地物分类的原理是：基于像元在特征空间中的位置进行分类，特征空间由选择的波段或波段的组合值（如遥感指数、主成分分析得到的各个主成分，等等）构成。其数学表达式为：

$$\text{Classify}(x) = \arg\min_i \|\boldsymbol{X} - \boldsymbol{\mu}_i\|$$

其中，$\boldsymbol{\mu}_i$是类别C_i的均值向量，$\|\cdot\|$表示在特征空间中的一种距离，可以使用曼哈顿距离（Manhattan Distance）$\sqrt{\sum_{i=1}^{n}(x_i - y_i)}$、欧氏距离（Euclidean Distance）$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$、马氏距离（Euclidean Distance）$\sqrt{(A-B)^T S^{-1}(A-B)}$等（$n$是特征向量的维度，$A$、$B$是特征空间中两点的向径，$S$是与某一类样本分布形态有关的协方差矩

阵）。

2.2.5 分类精度评定的基本原理

类精度评定是通过比较分类结果与真实地物类别，评估分类方法的准确性和可靠性。常用的精度评定指标包括总体分类精度（Overall Classification Accuracy, OCA）、Kappa系数（Kappa Coefficient）和用户精度（User's Accuracy）与生产者精度（Producer's Accuracy）。

1.**总体分类精度（OCA）**：表示分类正确的像素占总像素的比例，计算公式为：

$$\mathrm{OCA} = \frac{\sum_{i=1}^{k} n_{ii}}{\sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij}}$$

其中，n_{ii}是第i类别分类正确的像素数，n_{ij}是第i类别被分类为第j类别的像素数。

2.Kappa 系数：考虑了分类结果的随机性，计算公式为：

$$\kappa = \frac{\sum_{i=1}^{k} n_{ii} - \sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij}\, p_i p_j}{N - \sum_{i=1}^{k} p_i^2}$$

其中，p_i是第i类别的先验概率，N是总像素数。

3.用户精度（User's Accuracy）：表示被分类为第i类别的像素中，实际属于第i类别的比例，计算公式为：

$$\text{User's Accuracy}_i = \frac{n_{ii}}{\sum_{j=1}^{k} n_{ij}}$$

4.生产者精度（Producer's Accuracy）：又称制图精度，表示实际属于第i类别的像素中，被正确分类为第i类别的比例，计算公式为：

$$\text{Producer's Accuracy}_i = \frac{n_{ii}}{\sum_{j=1}^{k} n_{ji}}$$

通过这些精度评定指标，可以全面评估分类方法的性能，为后续的研究和应用提供依据。

2.2.6 专题制图的基本原理

专题制图是将分类结果或其他地理信息以地图的形式展示出来，以便直观地表达地理现象的分布特征。专题制图的基本步骤包括：

1.选择地图投影和坐标系统：根据研究区域的地理位置和制图目的，选择合适的地图投影和坐标系统。

2.设计地图布局：包括地图标题、比例尺、图例、指北针等元素的设计。

3.符号化和颜色选择：根据不同的地物类别，选择合适的符号和颜色，以增强地图的可读性和美观性。

4.添加辅助信息：如网格、注记等，以提供更多的地理信息。

5.输出地图：将设计好的地图输出为 PDF 或其他格式，以便打印或在线展示。

专题制图不仅能够直观地展示地理信息，还可以为城市规划、环境保护、土地利用研究等提供重要的决策支持。

2.3 实习步骤

这一节将详细介绍我在本次实习过程中应用 ERDAS 软件进行遥感分类专题信息提取与制图的操作步骤等。其中许多过程与结果文件中的后缀“_1259”是我（杨丹阳）的学号后 4 位，用于证明相关成果是我本人制作出来的。**请注意：本节不重点阐述相关操作的原理（对相关原理的解释请参见 2.2 节），也不对结果展开具体分析（对所得结果的分析请参见 2.4 节）。**

2.3.1 ERDAS IMAGINE 2015 软件的安装与配置

ERDAS IMAGINE 2015 软件的安装与配置不属于本次实习的重点，在此不展开介绍，其步骤可以参考这篇教程：https://blog.csdn.net/mrib/article/details/107354557。

2.3.2 对 SPOT 影像进行投影与坐标系设置

先介绍一些 ERDAS IMAGINE 2015 软件中的一些基本的图层操作：

导入数据：导入数据的方法有两种：一种是直接把要导入的数据拖到软件窗口中，另一种是右键视图视图 2D View #1-Open Raster Layer，然后选择要打开的数据。建议把实习所用的数据和产生的数据都放到英文文件夹下，否则在进行影像镶嵌那一步的时候会报错。

移除图层：右键点击相应的图层，选择 Remove Layer 即可，也可以按住 Ctrl 多选几个图层然后一起点击 Remove Layer，实现同时移除多个图层。

导出图层为文件：

同时显示两个视图：首先选择 File-Window-Add Views，选择 Display Two Views，这样就打开了左右两个视图，可以在左侧的 Contents 中看见两个视图（View）。可以在两边的视图中分别选择 Fit Layer To Window 来将图层缩放到屏幕。如果要关闭视图，点击右上角的×即可。

首先导入用于专题图制作的 数据 A/rsdata/sp_yc.tif （SPOT 卫星的宜昌遥感影像）到软件中。浏览要打开的数据时有一个小技巧：如图 2.3.2-1，右侧的 Recent 可以找到最近，打开或者保存的文件。

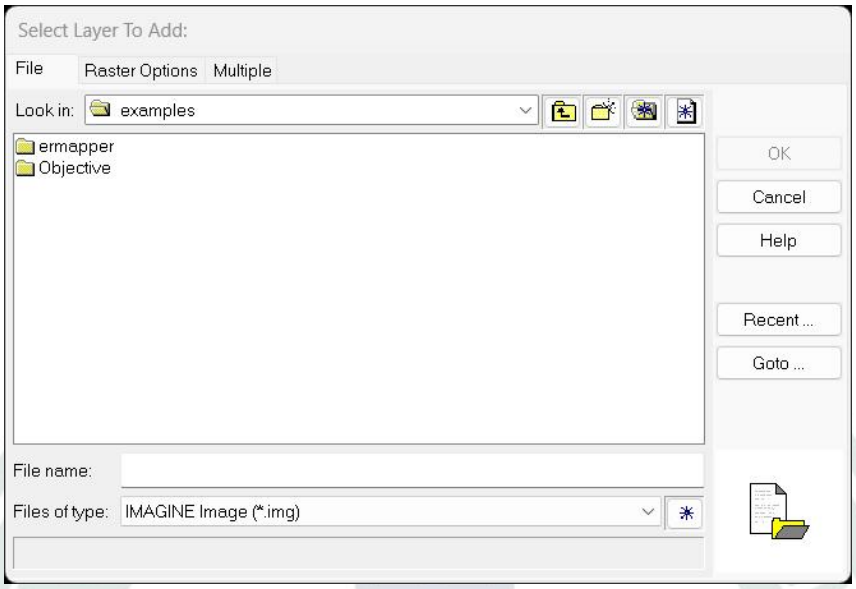


图2. 3. 2-1 浏览数据

打开文件时，注意设置 Files of type 为相应的数据文件类型。

打开 数据 A/rsdata/sp_yc.tif 之后，右键点击左侧 Contents 中的 sp_yc.tif，选择 Fit Layer To Window，可以实现缩放到图层（图 2.3.2-2）.

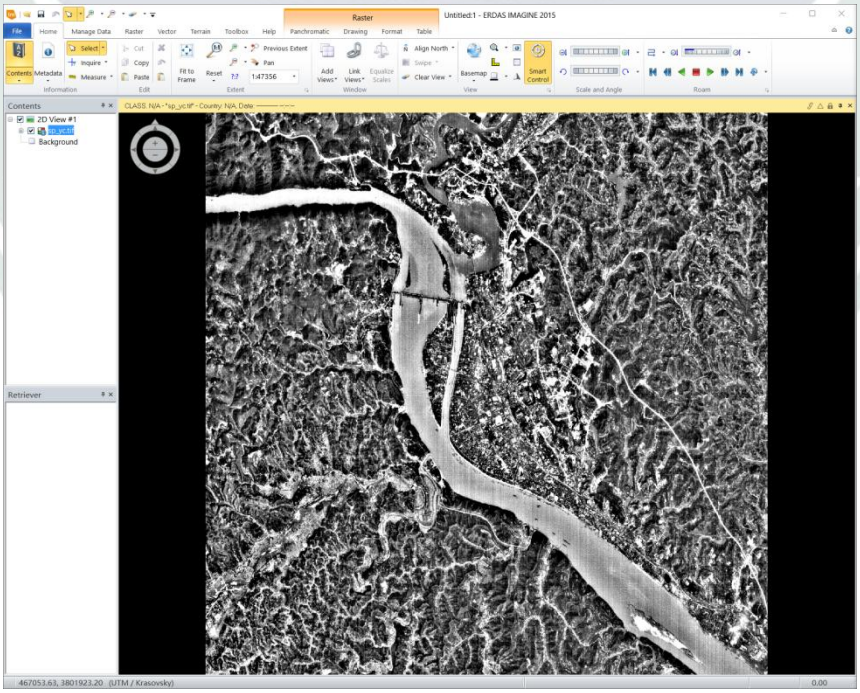


图2. 3. 2-2 显示遥感影像数据

找到上方的 Home-Information-Metadata，单击  图标，根据 数据

A/rsdata/readme.txt 中的信息填写（图 2.3.2-3）。

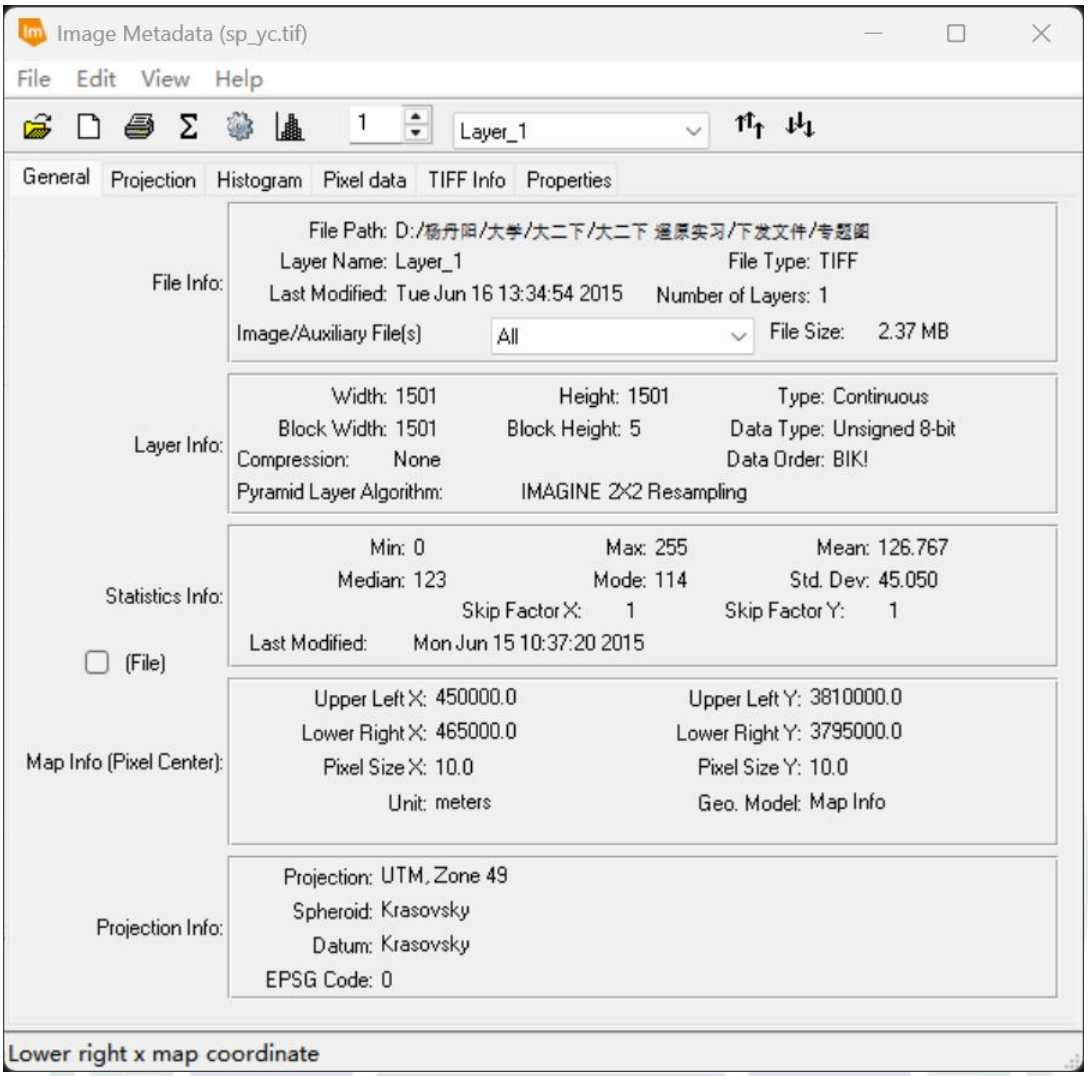


图2. 3. 2-3 配置投影与坐标系信息

2.3.3 对 TM 影像进行多波段合成

打开上方的 Raster-Spectral-Layer Stack，如图 2. 3. 3-1，在 Input File 里面选择 left 各个波段，注意每次选择一个波段的 tif 之后都要点击 Add，输出文件格式为 img，这是 ERDAS IMAGINE 软件内部使用的数据格式，点击 OK 即可输出。同样地输出 right 的各个波段。左右两景影像的 stack 需要分别输出。

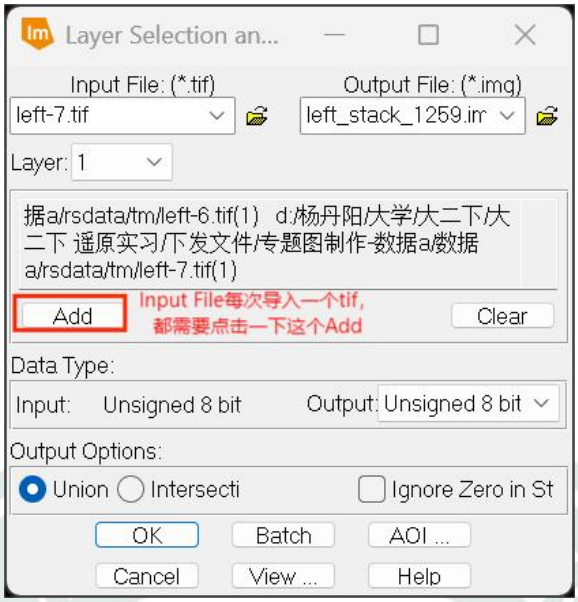


图2.3.3-1 对TM影像进行多波段合成

参与多波段合成的文件（在 数据 A/rsdata/tm 文件夹下）如图 2.3.3-2：

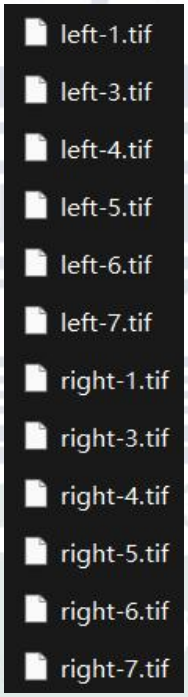


图2.3.3-2 参与多波段合成的文件

得到的文件如图 2.3.3-3：

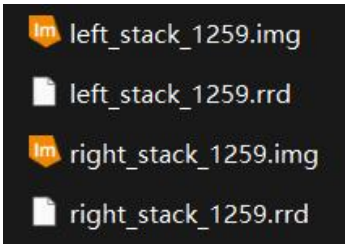


图2.3.3-3 对TM影像进行多波段合成得到的文件

2.3.4 对得到的多光谱的 stack 进行几何校正

导入 2.3.3 得到的 left_stack_1259.img 文件至软件，选择 Raster-Multispectral-Transform & Orthocorrect-Control Points（如图 2.3.4-1），然后会弹出一个窗口。注意弹出的窗口可能在主程序窗口的后面，被盖住了。

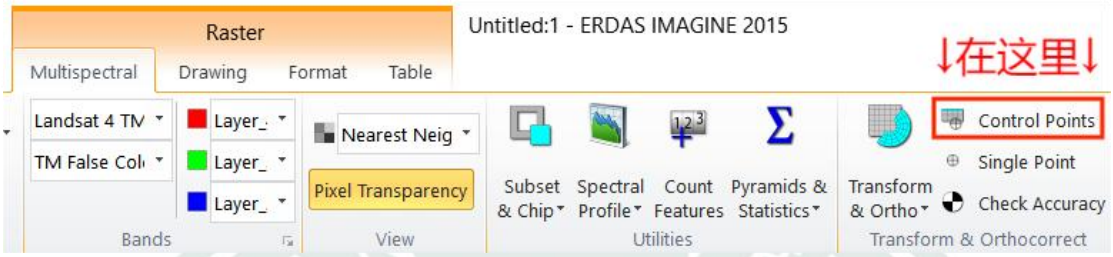


图2.3.4-1 打开几何校正工具

弹出的窗口中选择多项式校正方法 Polynomial（图 2.3.4-2），若是有那个闲情雅致去多戳几个点的话也可以去选其他模型，比如二次函数模型。我去年在 MATLAB 课上就注意到，控制点越多、用的拟合次数越高，效果却不一定更好（那个作业里面我用的是模式匹配的方法选择控制点，绝对准确，但是控制点越多、用的拟合次数越高，PSNR 却不一定更高）；在今年的地信实习里面，我手戳控制点的时候，由于戳得不准，如果戳太多或者用的函数模型次数太高的话，图像甚至会扭曲起来。这次实习就不为难自己了，一次多项式有(1+1)×(1+2)=6 个待定参数，需要至少 3 对同名点。

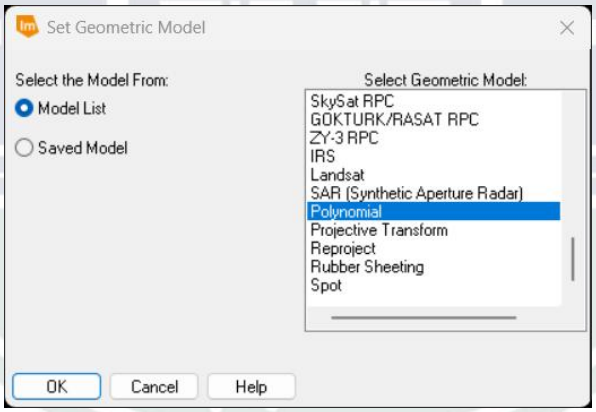


图2.3.4-2 设置几何校正所用的模型

然后弹出图 2.3.4-3 的窗口：

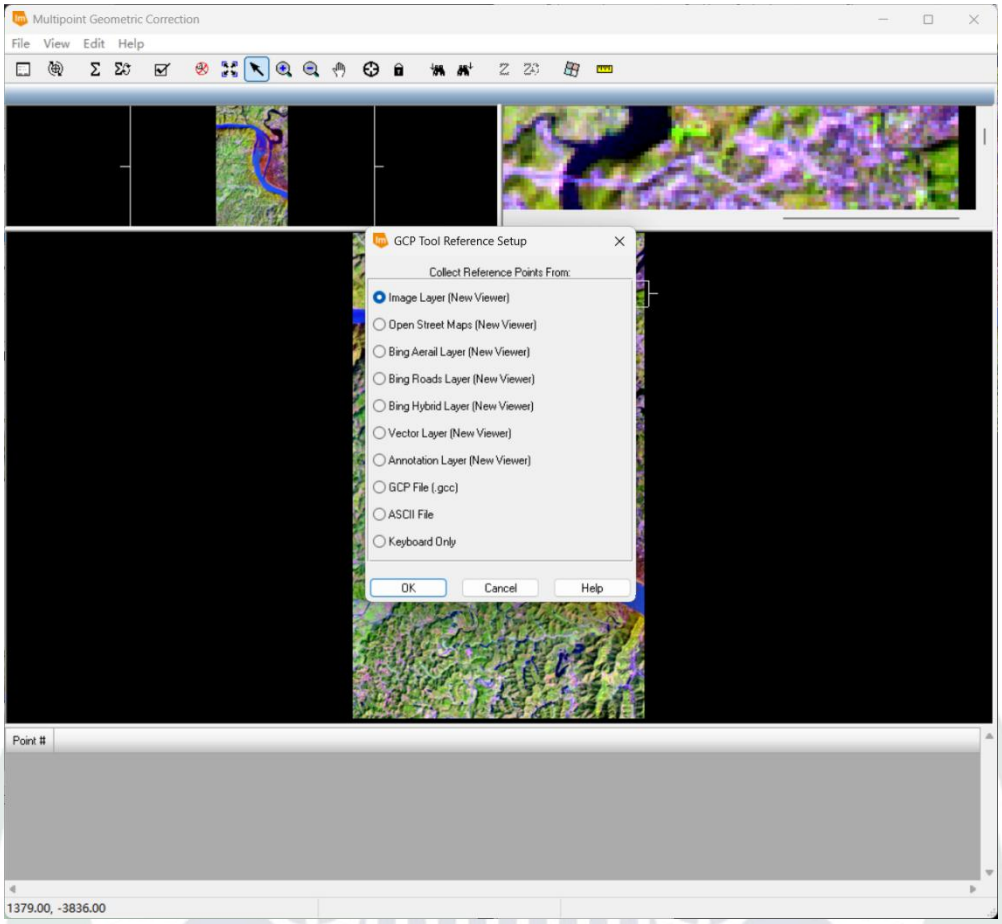


图2.3.4-3 弹出的几何校正工具界面

接下来的操作如图 2.3.4-4~图 2.3.4-7 所示：

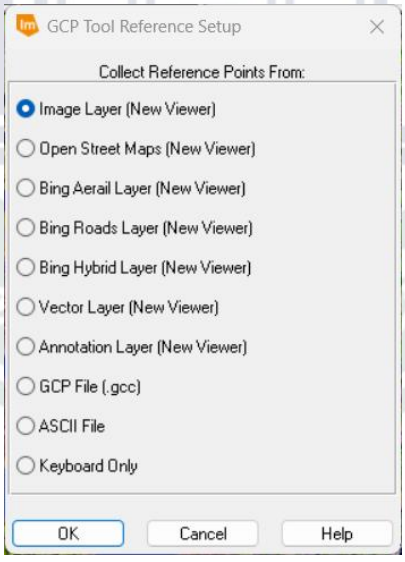


图2.3.4-4 点击OK

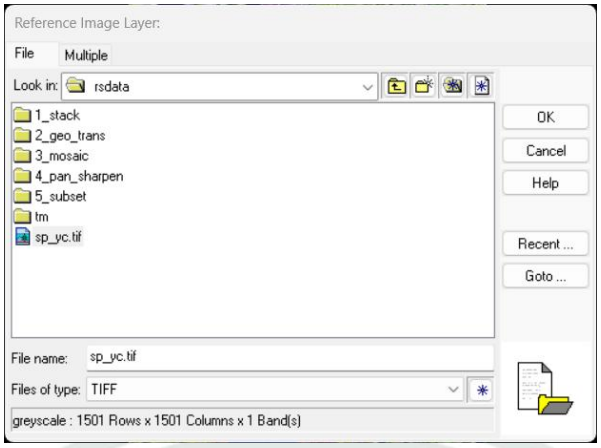


图2.3.4-5 选择sp-yc.tif，点击OK。也就是说，我们是把stack_1259.img的几何位置转换到sp-yc.tif的几何位置

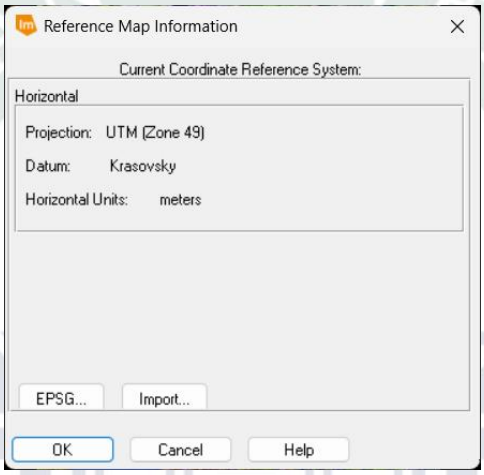


图2.3.4-6 点击OK

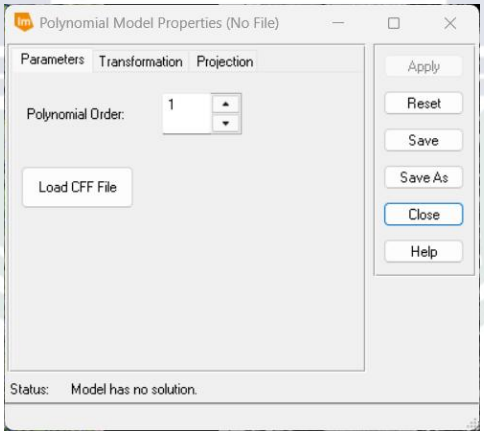


图2.3.4-7 点击Close或者直接点×

然后点击上方的  分别在左右两边创建同名点（视图操作方式：右键有 Fit Layer To Window 可以实现缩放到图层，长按中键拖动，滚轮放大缩小），同名点应该尽可能分布在图幅全幅并且分布均匀、易于辨识。去年暑假测绘实习就搞过这个事情，只不过这个图更难戳。先不管什么控制点、检查点的，先戳 6 对点。戳完之后，按住 Ctrl 选中

其中两个（图 2.3.4-8），然后点击上方的 Edit-Set Point Type-Check，这样就是把控制点（Control）变成检查点（Check）。

Point #	Point ID	Color	X Input	Y Input	Color	X Ref.	Y Ref.	Type	X Residual	Y Residual	RMS Error	Contrib.	Match
1	GCP #2		960.497	-3966.404		455589.085	3799400.724	Control	0.124	0.348	0.369	1.584	
2	GCP #3		919.244	-3990.280		454627.599	3798615.003	Control	-0.086	-0.242	0.257	1.103	
3	GCP #4		1037.777	-3888.126		457305.624	3801827.430	Control	-0.041	-0.115	0.122	0.522	
4	GCP #5		1076.248	-4004.874		458511.858	3798379.047	Check					
5	GCP #6		950.484	-3878.632		455154.921	3802001.202	Check					
6	GCP #1		826.903	-3631.481		451570.774	3809220.853	Control	0.003	0.009	0.010	0.041	
7	GCP #7							Control					

图2.3.4-8 按住Ctrl可以批量选中同名点数据

这里说明一下：一次多项式有(1+1)×(1+2)=6 个待定参数，需要至少 3 对同名点，为了产生多余观测发现误差，选择 4 对控制点，控制点是用于产生转换模型的参数的。检查点不参与产生转换模型的参数，而是根据控制点产生的转换模型的参数来计算转换模型的精度。

点击上方的  将会计算**控制点**的精度；点击  将会在左边戳点之后在右边自动计算出一个大致坐标（并计算**控制点**的精度），鼠标长按图上的同名点是可以拖动改变位置的；点击  将计算**检查点**的精度。精度将显示在窗口的右下角，分别有 X、Y、Total 的精度。本次实习技术指标要求 4 个控制点和 2 个检查点的点位误差（Total）均不超过 1.0000m。

为了在图上能够更好地显示同名点的分布，同样按住 Ctrl 多选控制点/检查点，用不同的颜色显示控制点/检查点（图 2.3.4-9）。注意左右两图需要分别设置颜色。

Point #	Point ID	Color	X Input	Y Input	Color	X Ref.	Y Ref.	Type	X Residual	Y Residual	RMS Error	Contrib.	Match
1	GCP #1		1096.238	-3713.163		458332.765	3807176.207	Control					
2	GCP #2		1162.835	-3814.688		460198.647	3804173.246	Control					
3	GCP #3		1170.560	-3881.190		460568.270	3802197.113	Control					
4	GCP #4		1136.284	-4095.164		460239.109	3795771.344	Control					0.397
5	GCP #5		1166.535	-3839.720		460355.420	3803437.026	Check	-0.151	0.205	0.255	0.342	0.810
6	GCP #6		1094.115	-4053.881		459082.940	3796987.140	Check	-1.005	0.187	1.023	1.372	0.786
7	GCP #7							Control					

图2.3.4-9 用不同的颜色显示控制点/检查点

图 2.3.3-10~2.3.4-13 显示了我对 left/right_stack_1259.img 进行几何精度校正的控制点分布与精度效果：

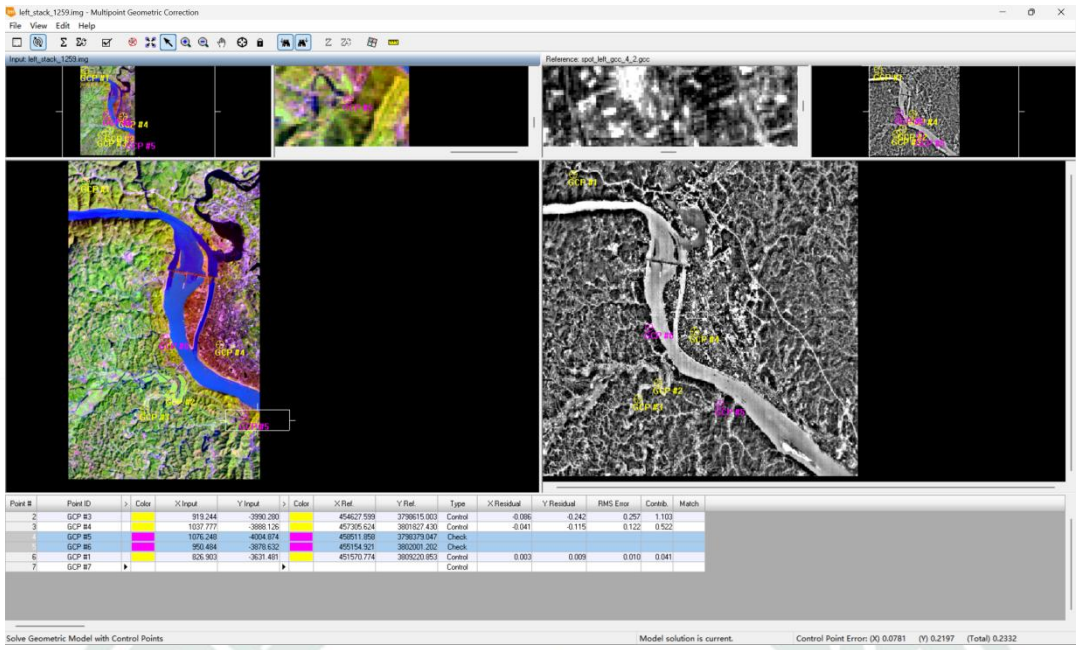


图2.3.4-10 对left_stack_1259.img进行几何精度校正的控制点精度（0.2332<1）

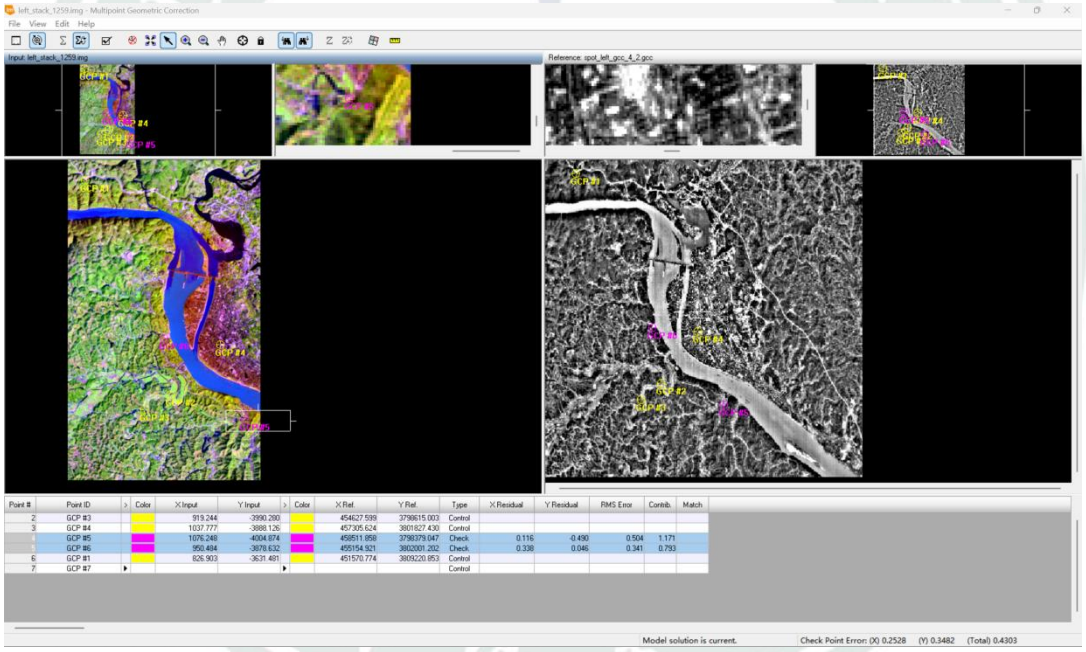


图2.3.4-11 对left_stack_1259.img进行几何精度校正的检查点精度（0.4303<1）

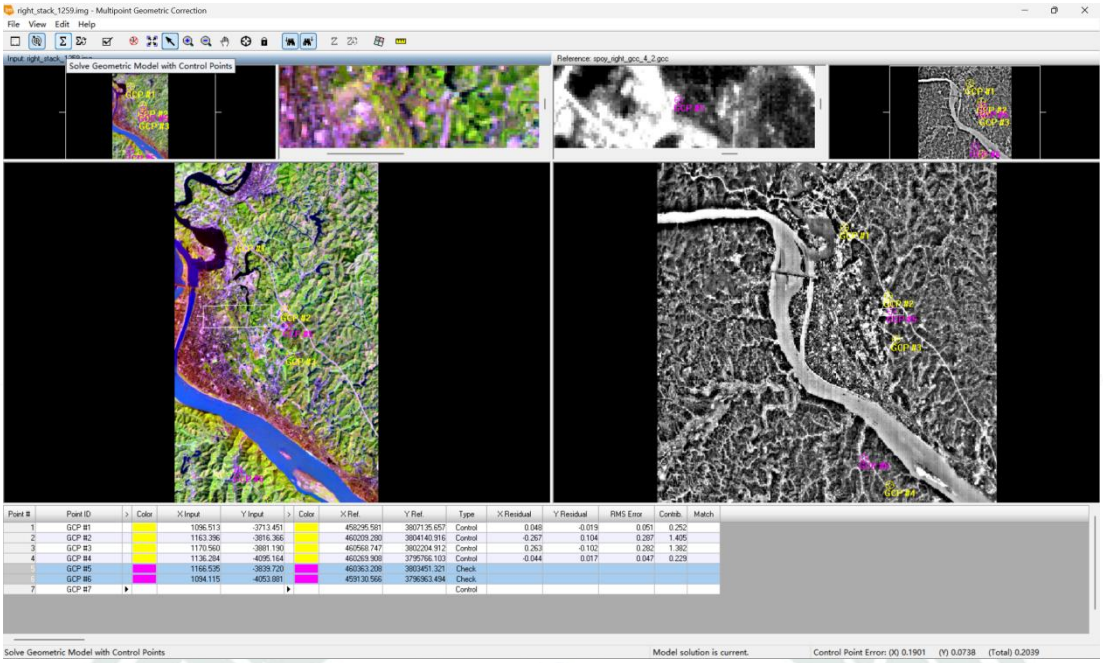


图2.3.4-12 对right_stack_1259.img进行几何精度校正的控制点精度（0.2039<1）

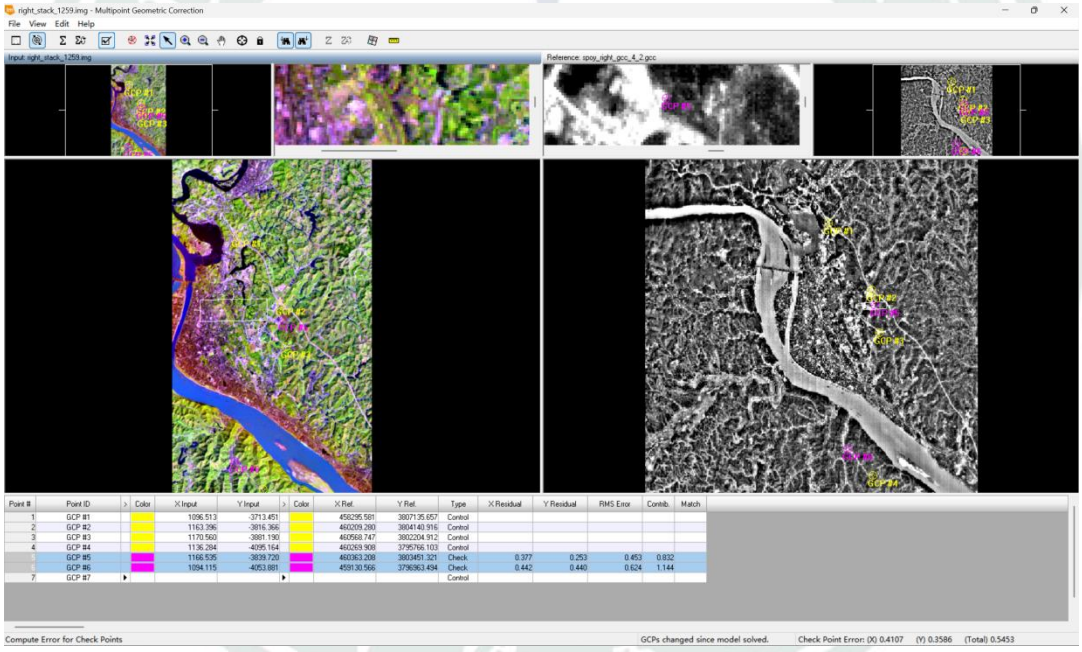


图2.3.4-13 对right_stack_1259.img进行几何精度校正的检查点精度（0.5453<1）

确认精度符合要求之后先别急着关。点击上方的File，你会看到如图2.3.4-14所示的菜单：

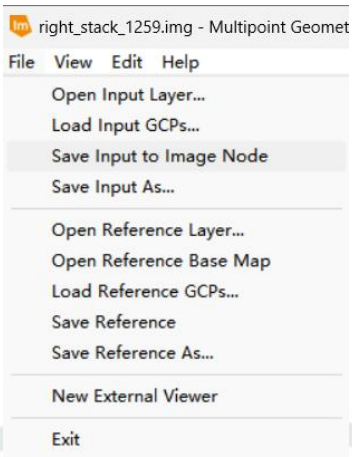


图2.3.4-14 File菜单中有各种保存几何转换控制点数据的功能

其中，Save Input to Image Node 可以把待转换的图（左边的图）上的点保存（类似于“关联”，别担心，不是绘制哦）到左边的图上，下次打开的时候就会自动带上点的数据；Save Input As 就是把左边图上的点的数据保存到本地；Save Reference As 可以把有右边的图（转换的目标图）上的点的数据保存到本地。强烈建议保存，毕竟戳一次点不容易。下次要用的时候，使用 Load Input GCPs 和 Load Reference GCPs 导入即可。

点击上方的  ，设置参数（图 2.3.4-15，注意像元大小 Output Cell Sizes 设置为 30），点击 OK 导出几何校正之后的 img。用同样的方法处理 left_stack_1259.img。

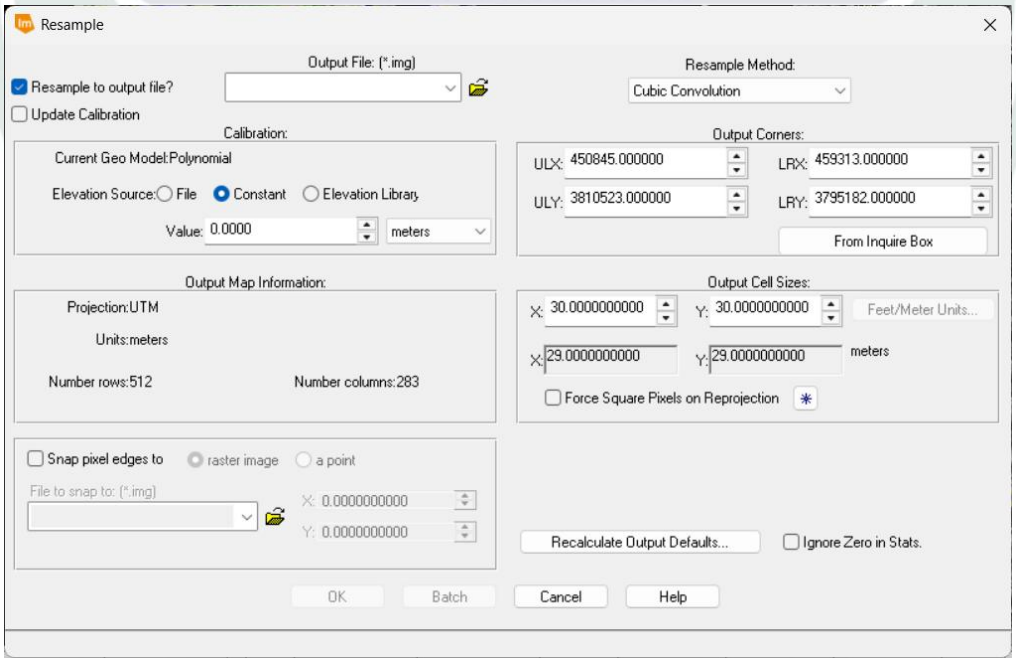


图2.3.4-15 设置导出几何校正后的img的参数

最后点击×的时候会弹出一个提示（图 2.3.4-16）：

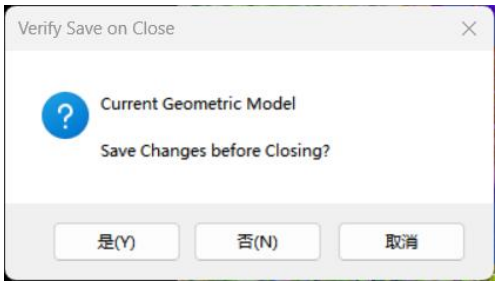


图2.3.4-16 提示是否保存几何校正的模型

我建议点击“是”，然后找个地方存好（图 2.3.4-17）：

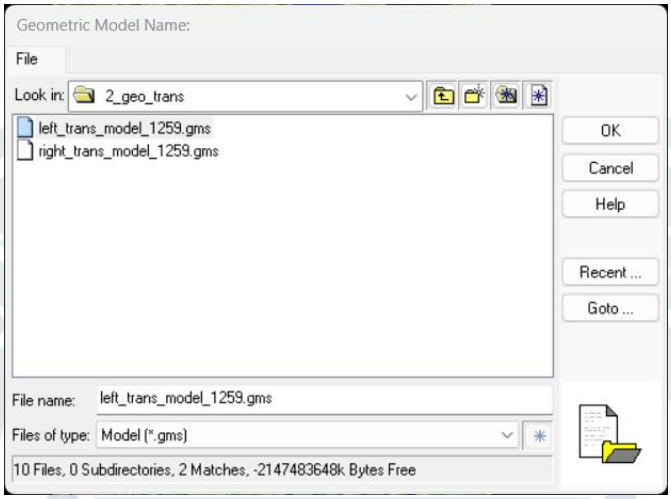


图2.3.4-17 保存几何校正的模型

然后你之前保存到图上的点的信息当然要点击保存“是”（图 2.3.4-18）

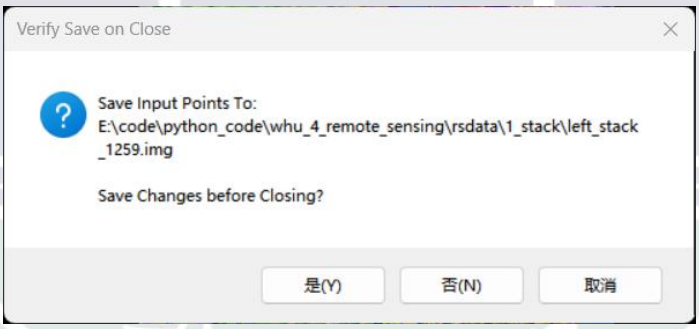


图2.3.4-18 保存图上的点的信息

用同样的方法对 right 的图像进行几何校正。然后把校正后的 left&right_stack_processed_1259.img 导入到主界面图层中查看（图 2.3.4-19），如果没出现明显的错位，看起来是一个整体的平行四边形，那校正效果还是可以的：

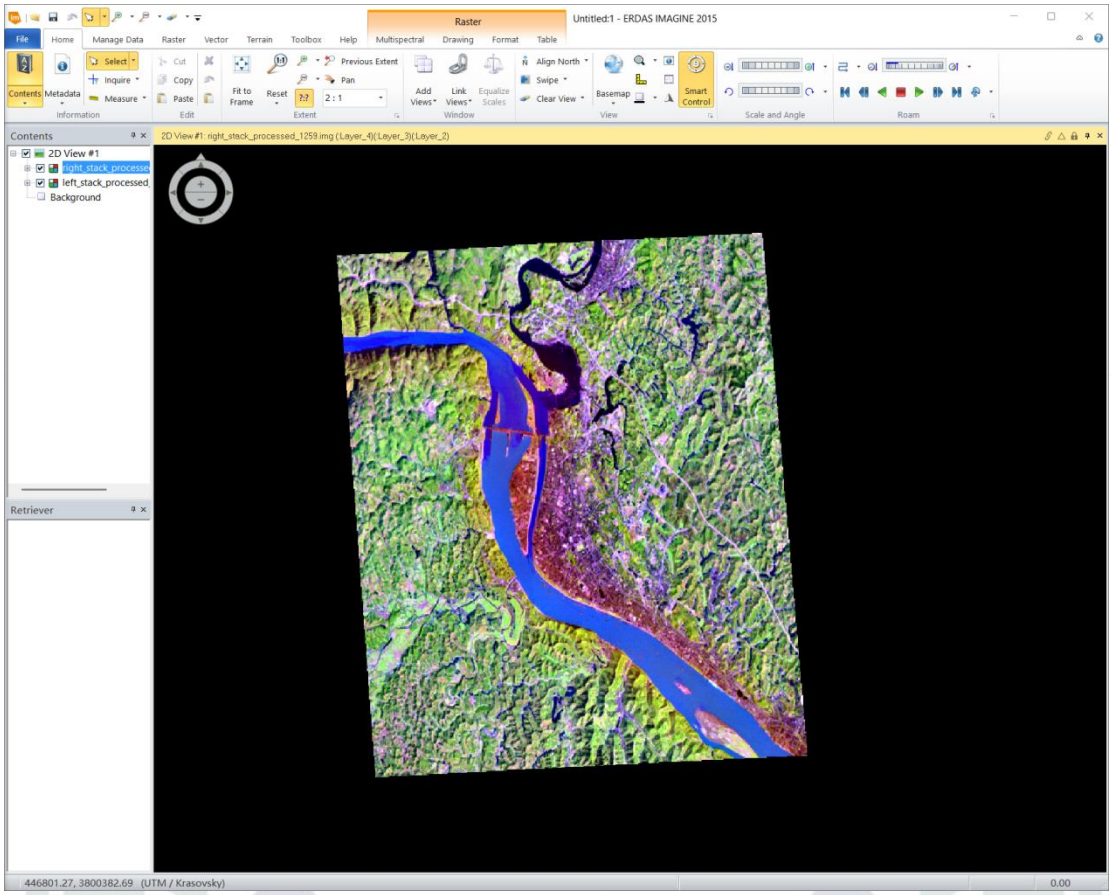


图2.3.4-19 查看几何校正的效果，我校正的效果还是可以的

当然还有另一种查看几何校正效果的方法：

首先在视图中导入几何校正得到的 left/right_stack_processed_1259.img 和几何校正的目标影像 sp_yc.tif，在左侧的 Contenes 里面拖动图层顺序，使得 left/right_stack_processed_1259.img 在 sp_yc.tif 的上面（图 2.3.4-20）；

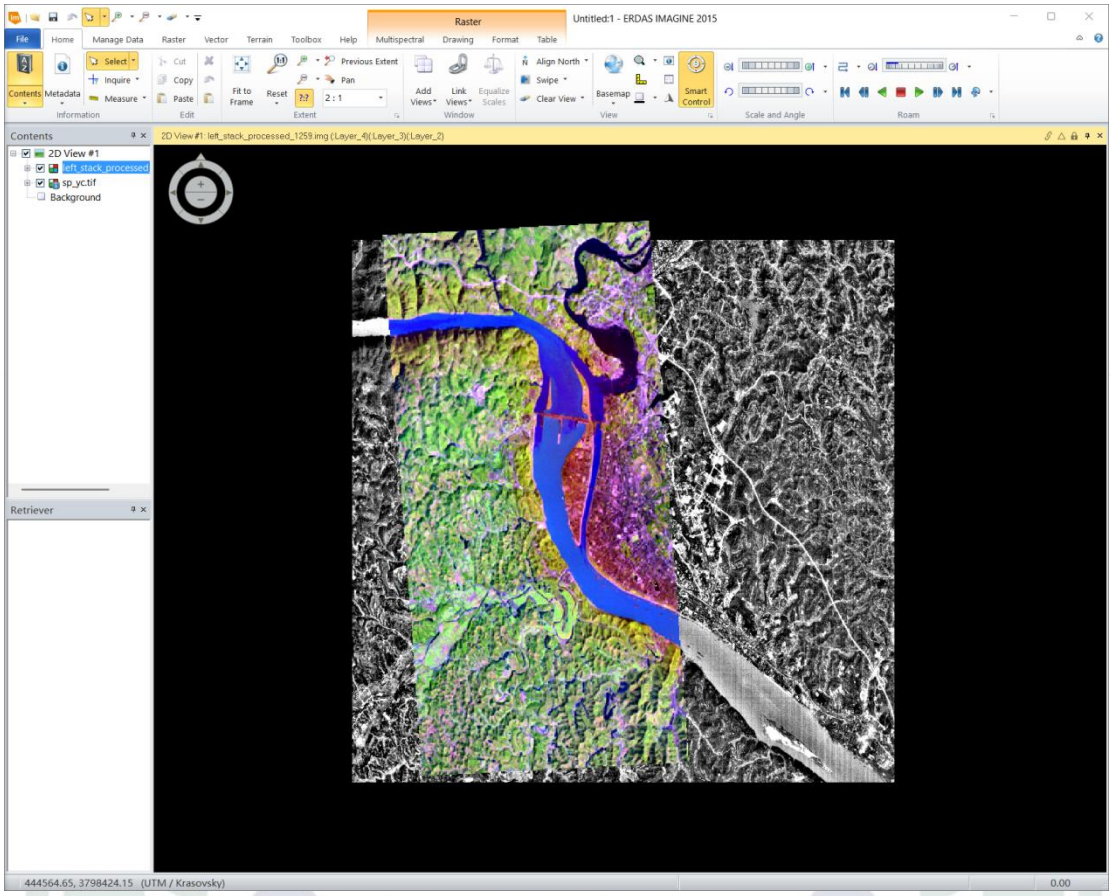


图2.3.4-20 在视图中导入几何校正得到的left/right_stack_processed_1259.img和几何校正的目标影像sp_yc.tif

然后需要使用 Blend 相关工具。由于这个工具不是很好找，点击上面的 Help，在 Search Commands 里面输入“blend”，按下 Enter，在右侧查询到 Blend 相关工具（图 2.3.4-21）：

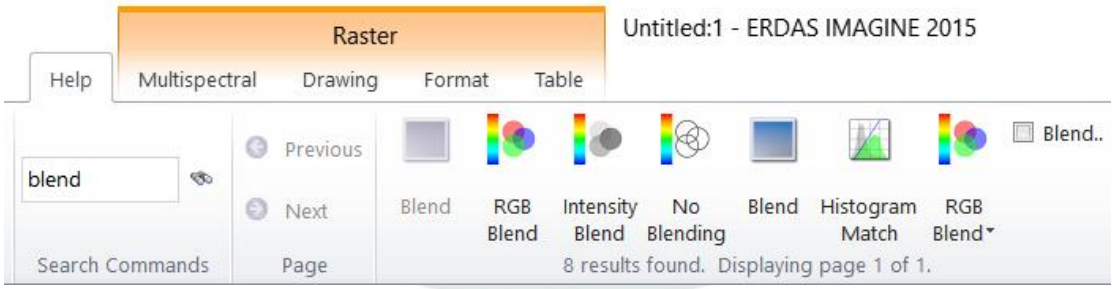


图2.3.4-21 使用Help中的搜索功能查找Blend工具

点击右边的 Blend，可以打开 Utility 相关工具（图 2.3.4-22）：

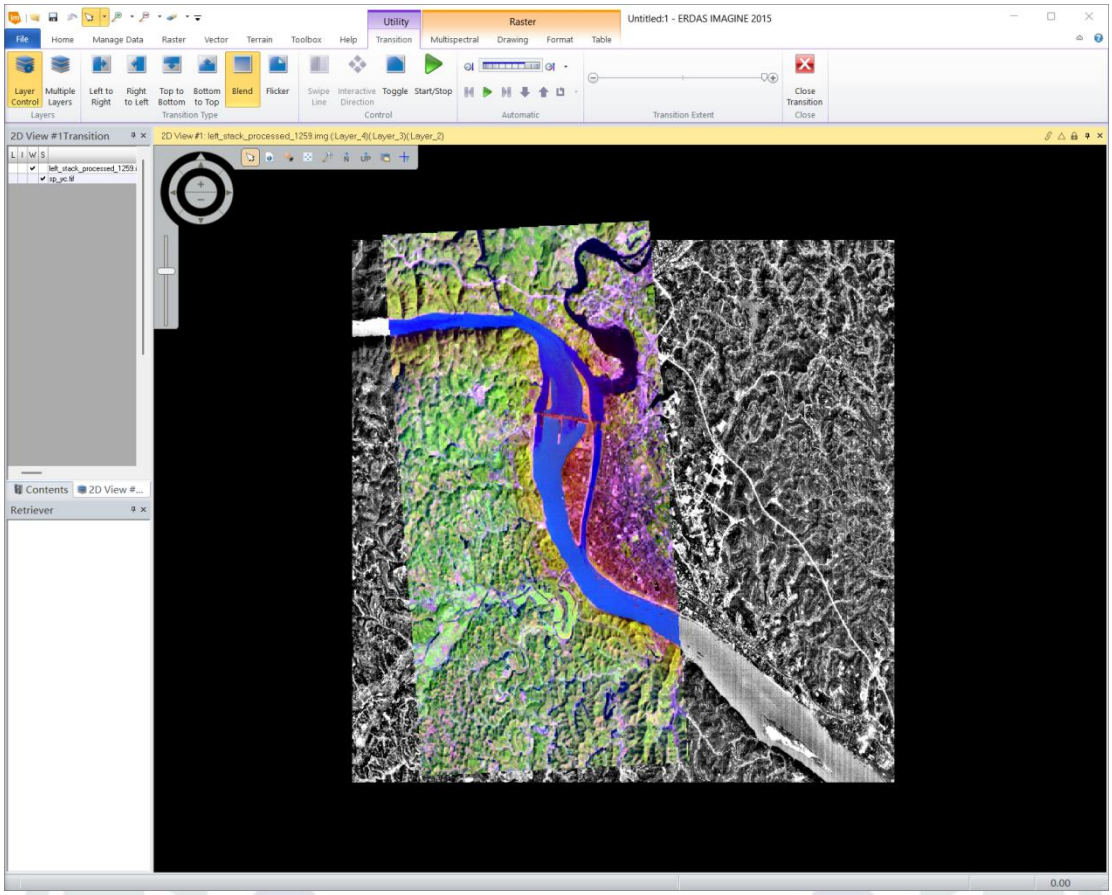


图2.3.4-22 打开Utility相关工具

选择上方的 Blend/Flicker/Toggle 等效果选项，点击上方的 Start/Stop，可以得到渐变/不断闪烁变化等动态效果，从而能够更加直观地检查几何校正的精度（图 2.3.4-23）。

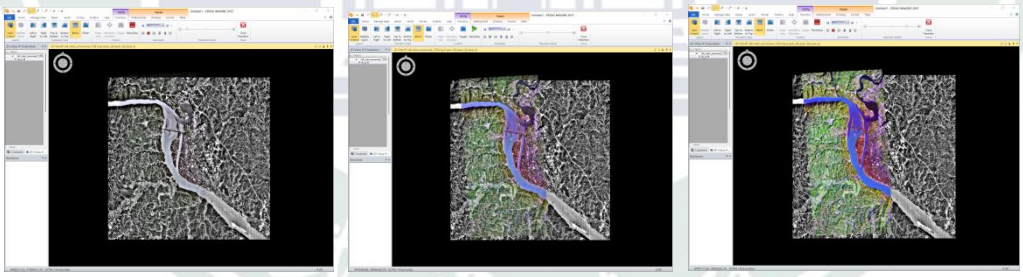
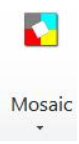


图2.3.4-23 渐变/不断闪烁变化等动态效果，能够更直观地检查几何校正的精度

2.3.5 影像镶嵌



选择上方的 Raster-Mosaic-MosaicPro（ ），点击后打开影像拼接的工具，将 2.3.4 得到的 left/right_stack_processed_1259.img 拖入其中（图 2.3.5-1）：

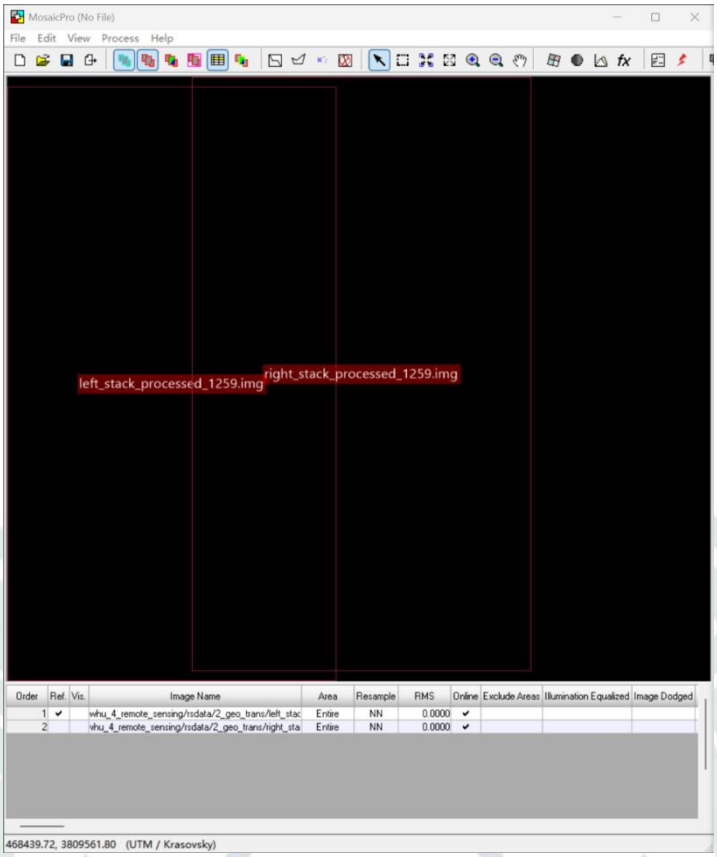
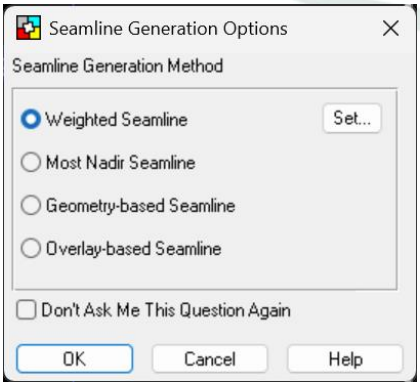


图2.3.4 影像拼接窗口

点击  可以设置镶嵌边：



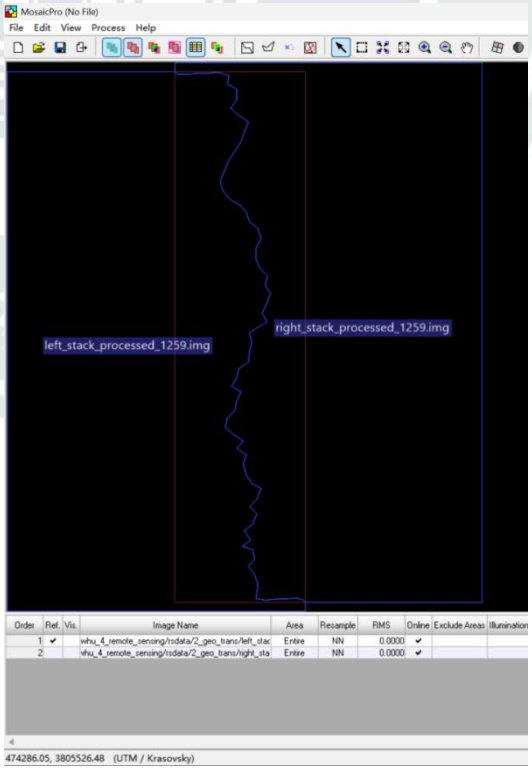


图2.3.4-2 设置镶嵌边（右图示Weigted Seamline镶嵌边）

这里我选择“Weighted Seamline”，然后再点击上方的  ，设置是否需要平滑（Smoothing）及羽化（Feathering）（图 2.3.4-3）：

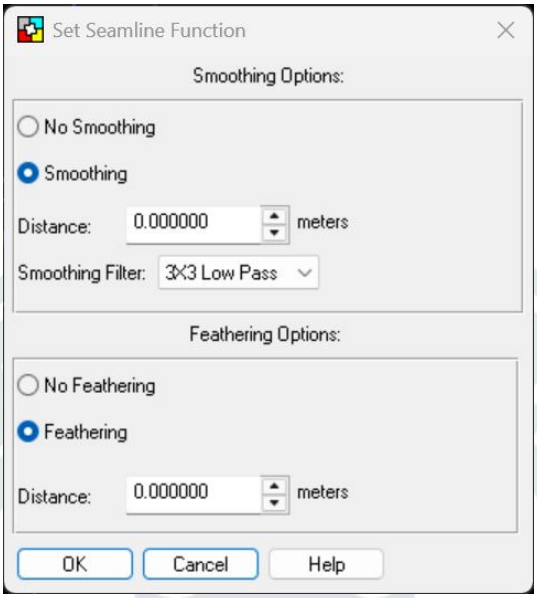


图2.3.4-3 设置是否需要平滑（Smoothing）及羽化（Feathering）

点击  可以设置重采样方法（图 2.3.4-4）：

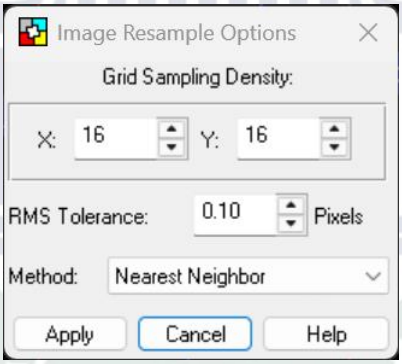


图2.3.4-4 设置重采样方法

点击  可以进行直方图匹配（图 2.3.4-5）：

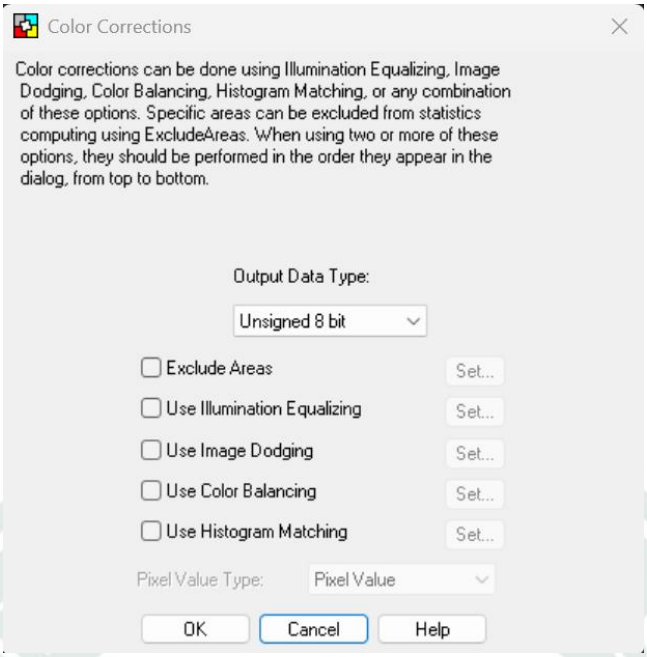


图2.3.4-5 进行直方图匹配

其中，Output Data Type 是输出数据类型，例如 Unsigned 8bit 表示输出图像为 8 位无符号整型（标准 RGB 图像格式，范围 0~255）；Exclude Areas 是排除区域，Set...允许用户指定图像中某些区域（如背景、水印等）不参与颜色校正的统计计算，避免无关区域影响校正结果（例如排除过亮/过暗的干扰区域）；各种颜色校正方法包括（可多选或组合使用）：Use Illumination Equalizing（光照均衡）、Use Image Dodging（图像减淡）、Use Color Balancing（色彩平衡）、Use Histogram Matching（直方图匹配）等等。

点击  可以导出影像拼接的结果：

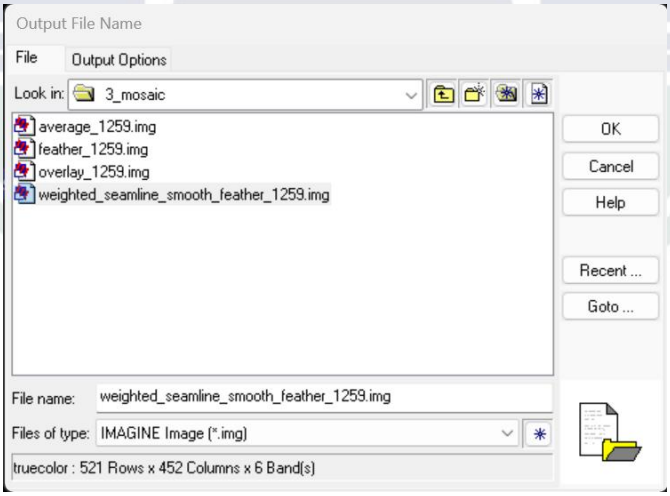


图2.3.5-6 导出影像拼接的结果

点击上方的 Output Options 可以设置“Ignore Input Values”为 0，避免融合得到的影像中存在阴影（图 2.3.5-7）：

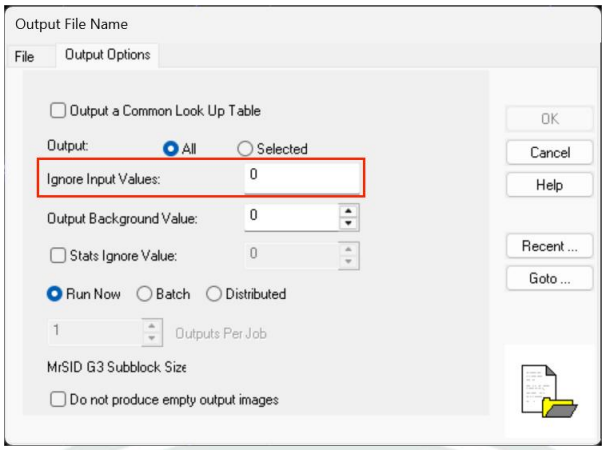


图2.3.5-7 设置“Ignore Input Values”为0

点击  也就是不使用镶嵌边，那么此时点击上方的  设置的就是各种在重叠区域的“融合”算法（图 2.3.5-8）：

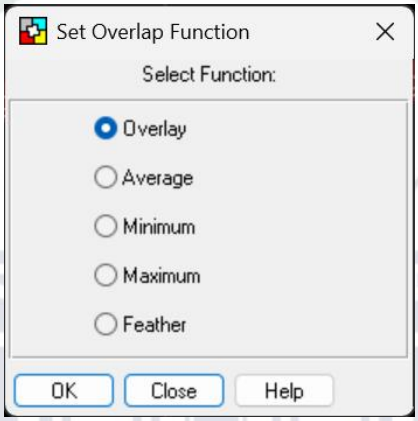
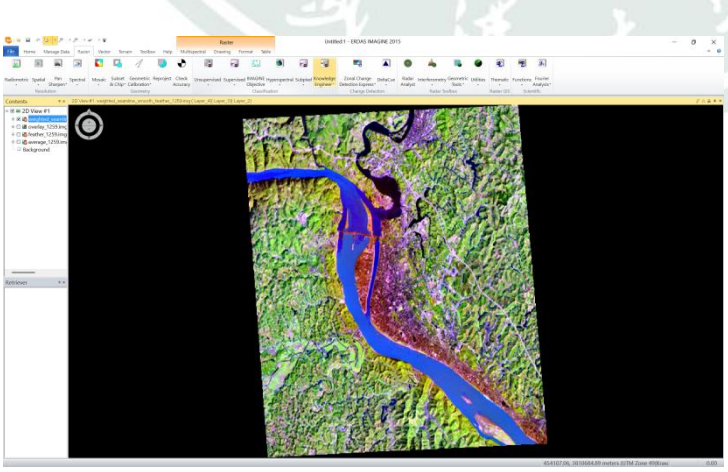


图2.3.5-8 在不使用镶嵌边的情况下，各种在重叠区域的“融合”算法

可以选择一种重叠混合模式，决定重叠部分的像素值如何计算：Overlay（叠加）、Average（平均）、Minimum（最小值）、Maximum（最大值）、Feather（羽化，在重叠区域边缘创建平滑的渐变过渡）。

作者尝试使用了几种方法（图 2.3.5-9~2.3.5.12）：



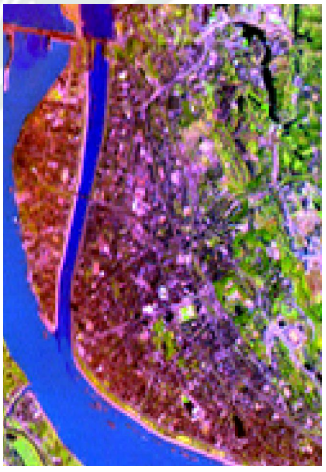


图2.3.5-9 使用Weighted Seamline镶嵌边，并使用平滑（Smoothing）和羽化（Feathering），右图示重叠处放大细节

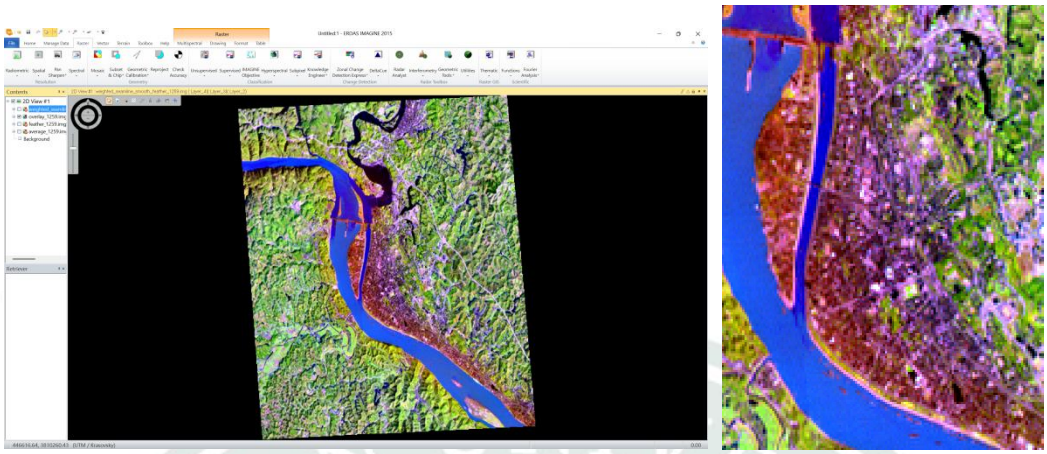


图2.3.5-10 未使用镶嵌边，使用覆盖（Overlay），右图示重叠处放大细节

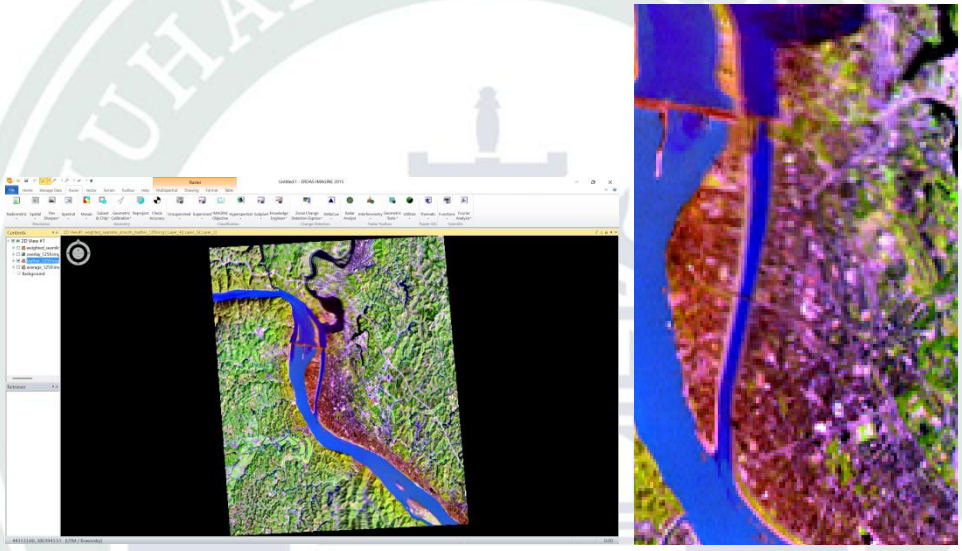


图2.3.5-11 未使用镶嵌边，使用羽化（Feather），右图示重叠处放大细节（有明显模糊）

图2.3.5-12 未使用镶嵌边，使用平均（Average），右图示重叠处放大细节（有明显模糊）

由于两幅影像在进行几何校正时的细小精度差异等原因，使用羽化（Feather）与平均（Average）方法得到的影像存在明显的模糊，使用覆盖（Overlay）镶嵌的结果中可

以看出一条拼接痕迹，因此作者选择使用 Weighted Seamline 镶嵌边，并使用平滑（Smoothing）和羽化（Feathering）（图 2.3.5-9）所示的图形完成接下来的步骤。

2.3.6 影像融合

点击上方的 Raster-Pensharpen，选择下拉菜单（图 2.3.6-1）中的一种方法进行影响融合。这些方法都是遥感影像融合（Resolution Merge 或 Pan-sharpening）中常用的技术，旨在将高分辨率的全色（Panchromatic, PAN）影像与低分辨率的多光谱（Multispectral, MS）影像融合，生成兼具高空间分辨率和高光谱信息的新影像。Subtractive Resolution Merge 基于光谱减法原理，通过从全色影像中提取高频空间信息（如边缘、纹理），并将其注入多光谱影像中；HPF Resolution Merge（High-Pass Filtering）使用高通滤波器提取全色影像的高频细节，直接叠加到多光谱影像上；Modified IHS Resolution Merge（改进的强度-色度-饱和度融合）：传统 IHS 将多光谱影像从 RGB 空间转换到 IHS 空间，用全色影像替换强度分量（I），再逆变换回 RGB 的方法，改进版通过局部自适应或加权减少光谱失真；Wavelet Resolution Merge（小波变换融合）对全色和多光谱影像进行小波分解，在不同频率子带（低频近似、高频细节）上选择性融合，最后重构影像；Ehlers Fusion 是由 Klaus Ehlers 提出的基于傅里叶变换的融合方法，通过频域分离光谱和空间信息，减少失真；HCS Resolution Merge（Hyperspherical Color Space）将多光谱影像转换到超球体色彩空间，替换亮度分量后逆变换，适用于高光谱数据；Resolution Merge（通用术语）可能指传统方法（如 PCA、Brovey 变换等）；Projective Resolution Merge 基于投影变换的融合方法，通过数学投影将全色影像信息映射到多光谱空间。实际选择哪一种方法需结合数据特性（传感器类型、波段数）和应用需求。

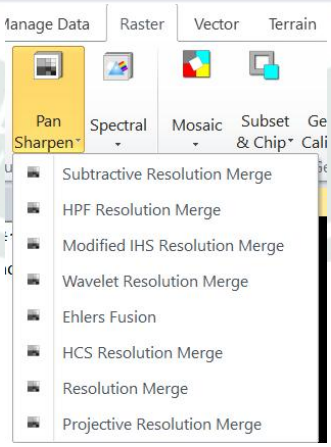


图2. 3. 6-1 影响融合算法选项

在打开的窗口中的配置示例如图 2.3.6-2 所示：

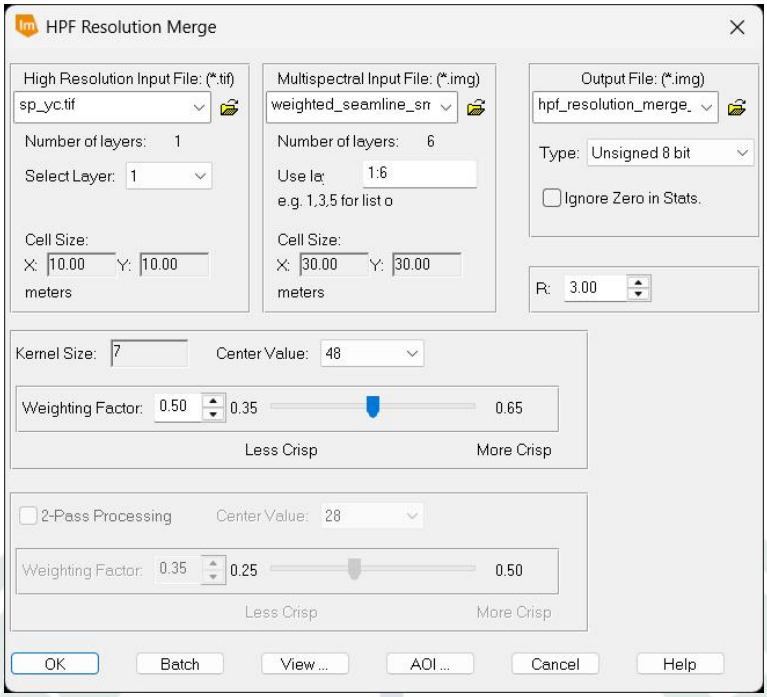


图2.3.6-2 影像融合的配置

作者比较了 HPF Resolution Merge 和普通的 Resolution Merge 的效果（图 2.3.6-3、图 2.3.6-4）：

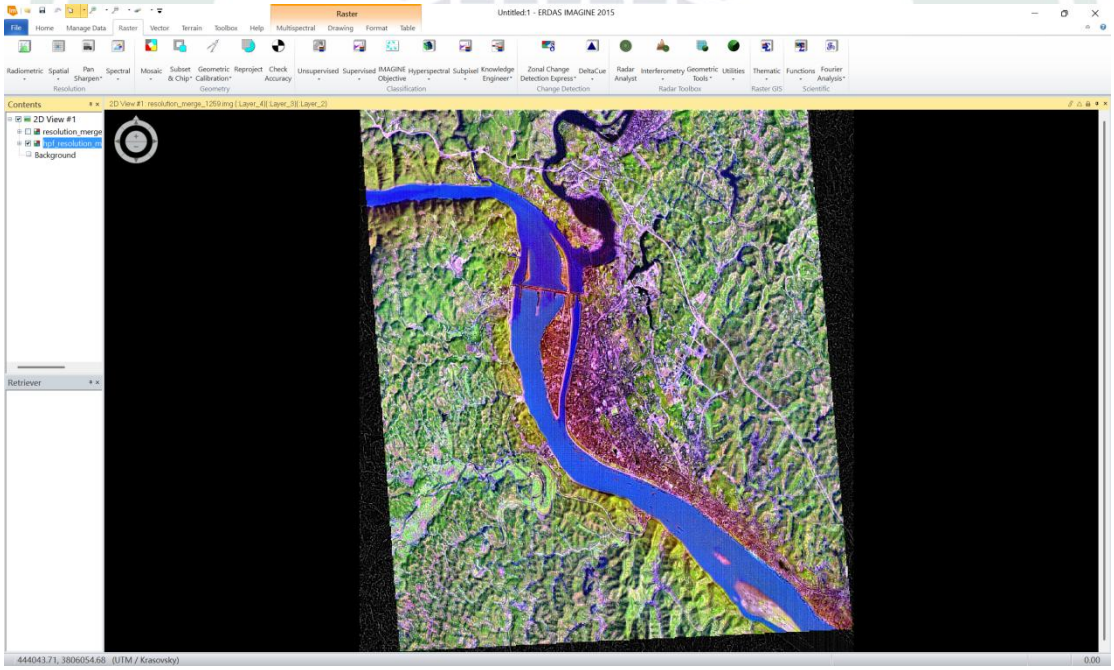


图2.3.6-3 HPF Resolution Merge的效果

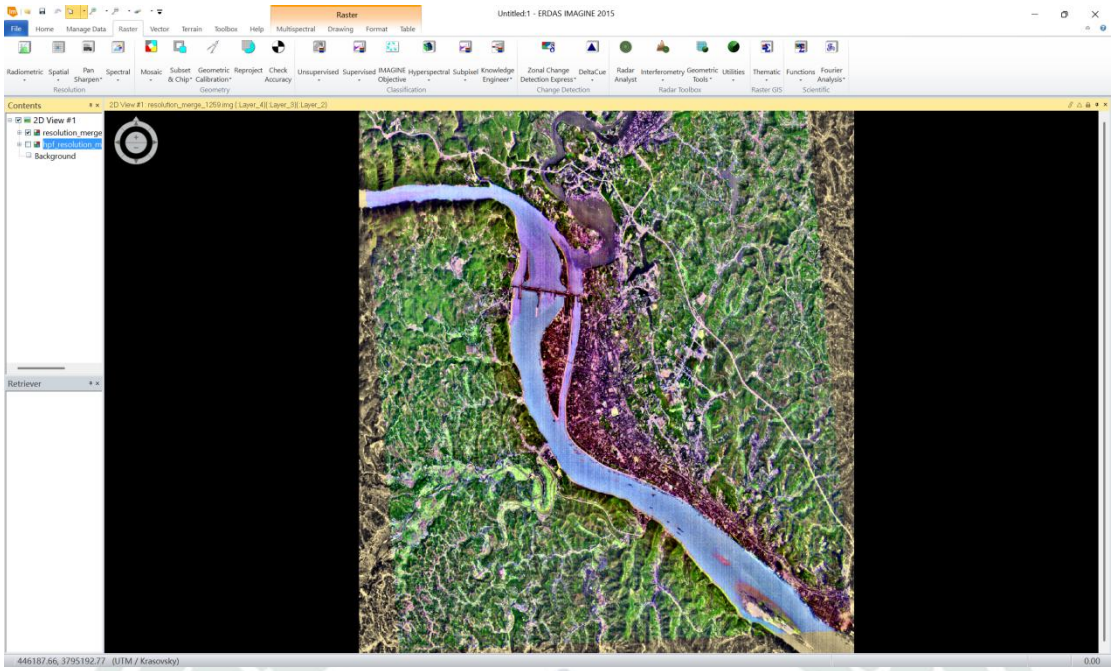


图2.3.6-4 Resolution Merge的效果

作者认为 HPF Resolution Merge 能将高分辨率的 sp_yc.tif 影像的空间信息更加清晰地呈现在 TM 影像的光谱信息中，效果更好，所以后续操作基于 HPF Resolution Merge 的结果进行。

2.3.7 影像裁剪

先将 2.3.6 得到的 hpf_resolution_merge_1259.img 拖到软件中，然后选择上方的 Raster-Drawing，点击 Insert Geometry 中的 ，开始绘制 AOI（图 2.3.7-1）：

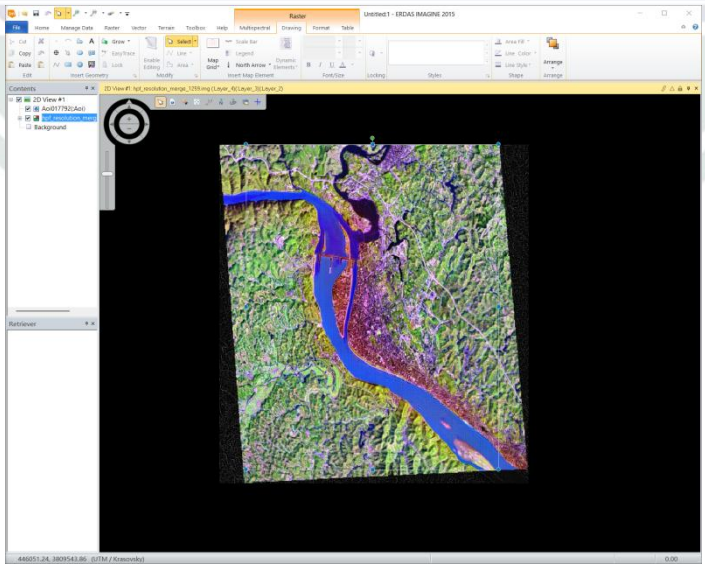


图2.3.7-1 绘制AOI（那个框就是绘制的矩形AOI）

绘制完成后，可以右键左侧 Contents 中的 AOI 图层，将选择 Save Layer As，将绘

制的裁剪范围保存为本地的.aoi 文件（图 2.3.7-2）：

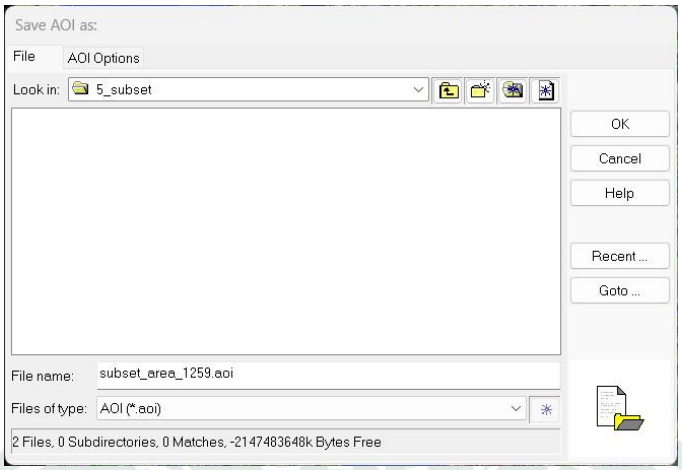


图2.3.7-2 保存绘制的裁剪范围为本地的.aoi文件

然后点击上方的 Raster-Subset & Chip-Create Subset image，在打开的窗口中配置如下（图 2.3.7-3）：

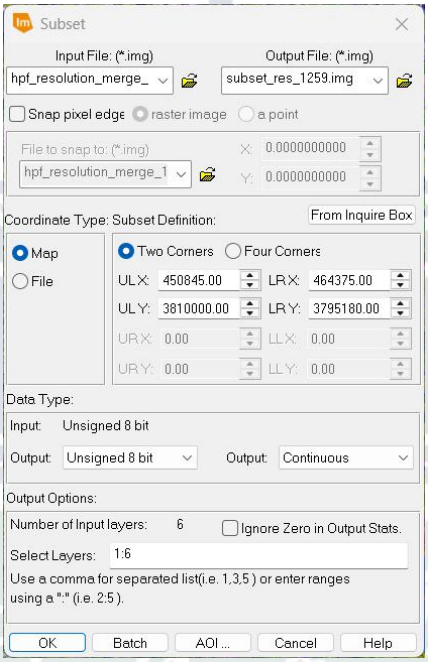


图2.3.7-3 影像裁剪窗口的配置

点击下方的 AOI...按钮，在打开的窗口中可以选择 Viewer（视图上绘制的 AOI 范围），或者选择 AOI File 来加载刚刚保存的.aoi 文件（图 2.3.7-4）：

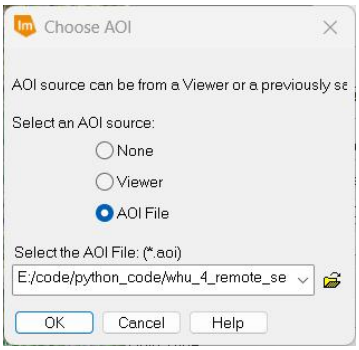


图2.3.7-4 选择AOI范围

点击 OK，再点击前面的那个窗口的 OK，运行完成后将得到的 subset_res_1259.img 拖到软件中，可以查看裁剪的效果（图 2.3.7-5）：

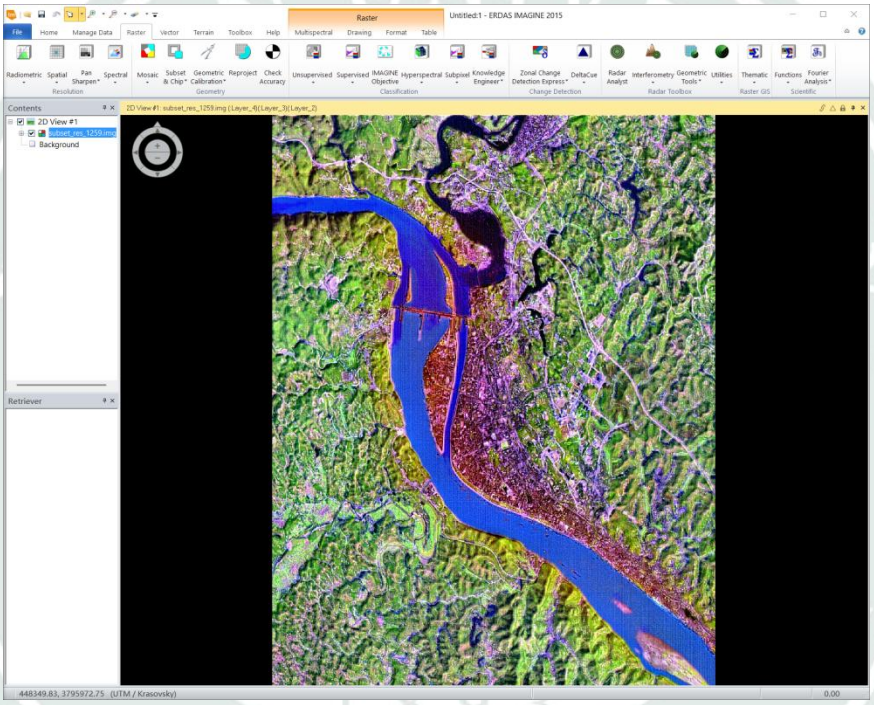


图2.3.7-5 裁剪效果

2.3.8 非监督分类

导入 2.3.7 所得的 subset_res_1259.img，点击 Raster-Classification-Unsupervised-Unsupervised Classification，在打开的窗口中进行如下配置（图 2.3.8-1，注意分类数设置为 6）：

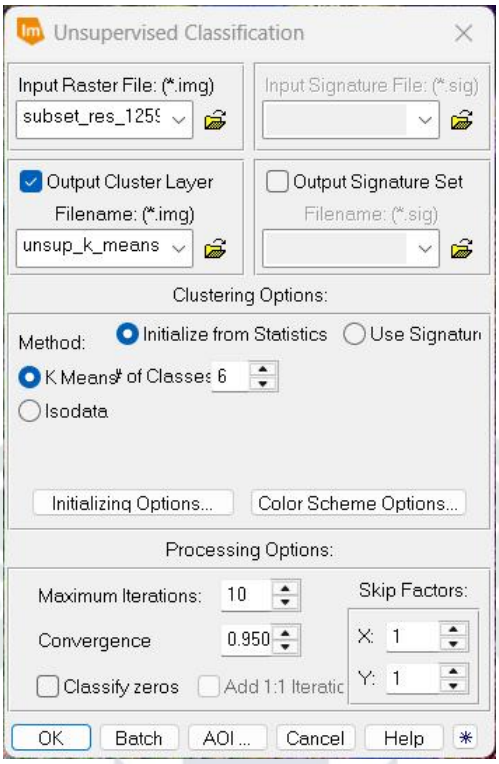


图2.3.8-1 非监督分类的配置

点击其中的 Initializing Options，可以设置初始撒点的位置（图 2.3.8-2）：

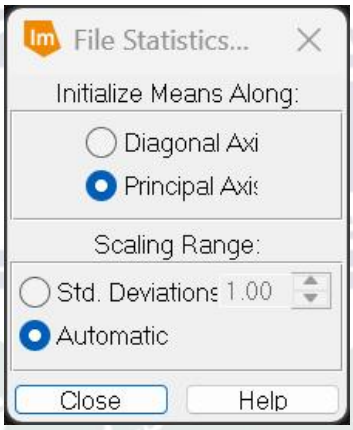


图2.3.8-2 设置聚类分析的初始化配置

其中，Diagonal Axis 表示初始化均值点沿数据空间的对角线分布，Principal Axis 表示沿主成分方向初始化均值点（常用于 PCA 预处理后的聚类）。

点击 OK 即可运行，运行的时候可以在图 2.3.8-3 所示的窗口中看见迭代的过程。

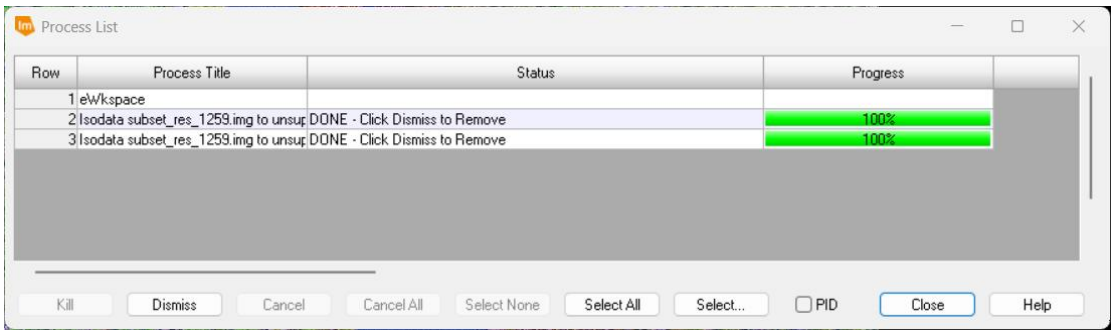


图2.3.8-3 运行任务窗口，可以从进度条中看出不断的迭代过程

作者尝试了两种方法进行非监督分类：K-Means 和 ISODATA，效果分别如图 2.3.8-4、图 2.3.8-5 所示：

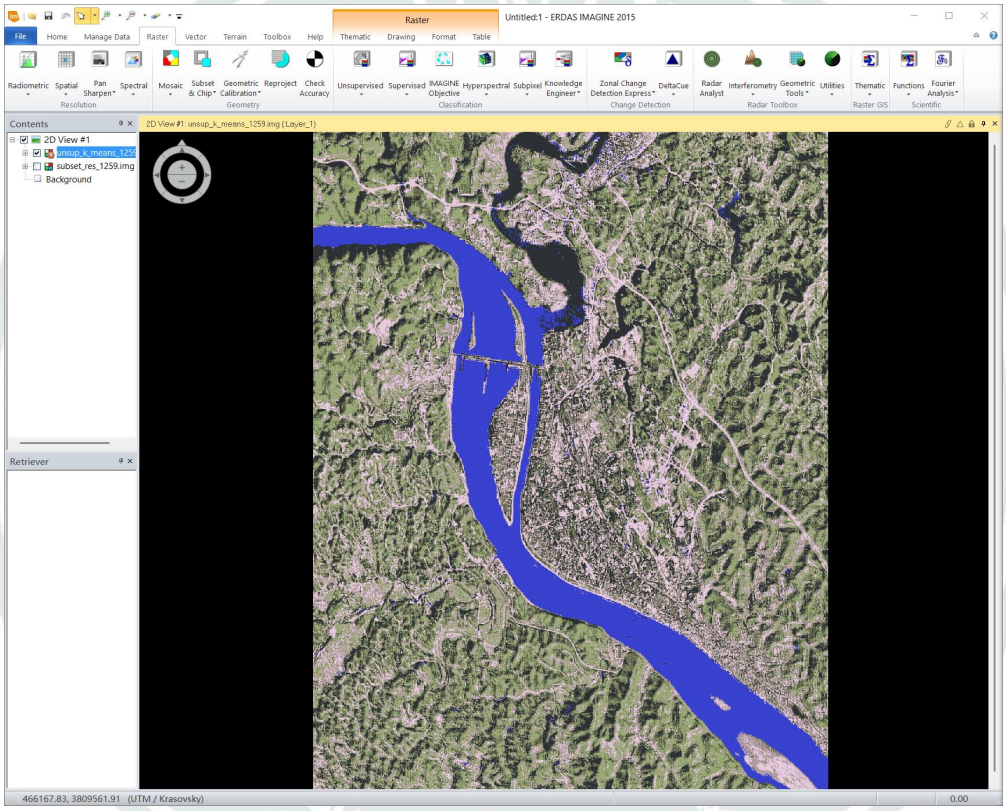


图2.3.8-4 K-Means分类的效果

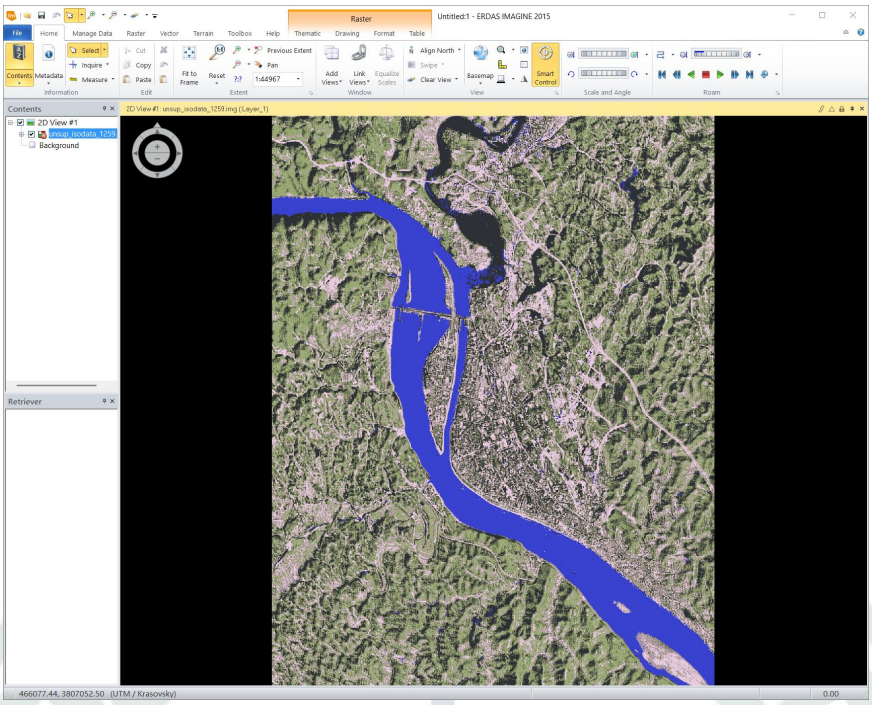


图2.3.8-5 ISODATA分类的效果

选择 File-Window-Add Views，选择 Display Two Views，这样就打开了左右两个视图，可以在左侧的 Contents 中看见两个视图（View）。可以在两边的视图中分别选择 Fit Layer To Window 来将图层缩放到屏幕。对比两种非监督分类方法得到的结果（图 2.3.8-6，左边是 K-Means 算法的结果，右边是 ISODATA 算法的结果）。

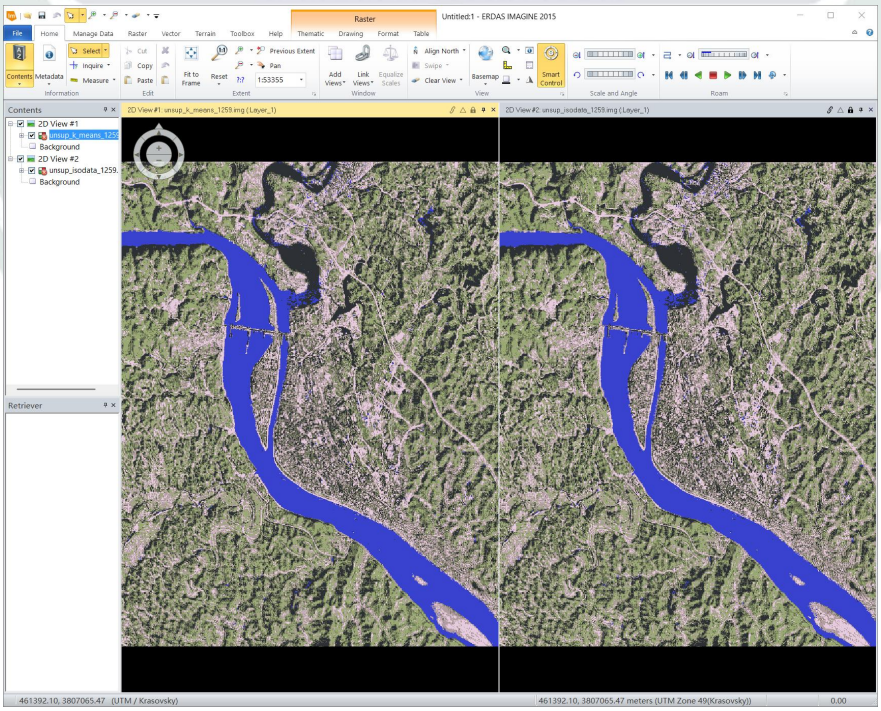


图2.3.8-6 左边是K-Means算法的结果，右边是ISODATA算法的结果

可以看出二者结果几乎是一样的。这是因为我为了保证分类数一样，设置的 ISODATA 的最大、最小类别数都是 6。事实上，ISODATA 算法是改进的 K-Means 算法，

只是在 K-Means 算法的基础上允许进行类别的分裂、合并与消失，所以如果限制 ISODATA 算法的结果中的类别数必须和 K-Means 算法一样的话，当然结果几乎一样了。

选择上方的 Raster-Table-Show Attributes，或者右键点击得到的专题图数据并选择 Display Attributes Layers 可以在下方看到出现的数据表格：

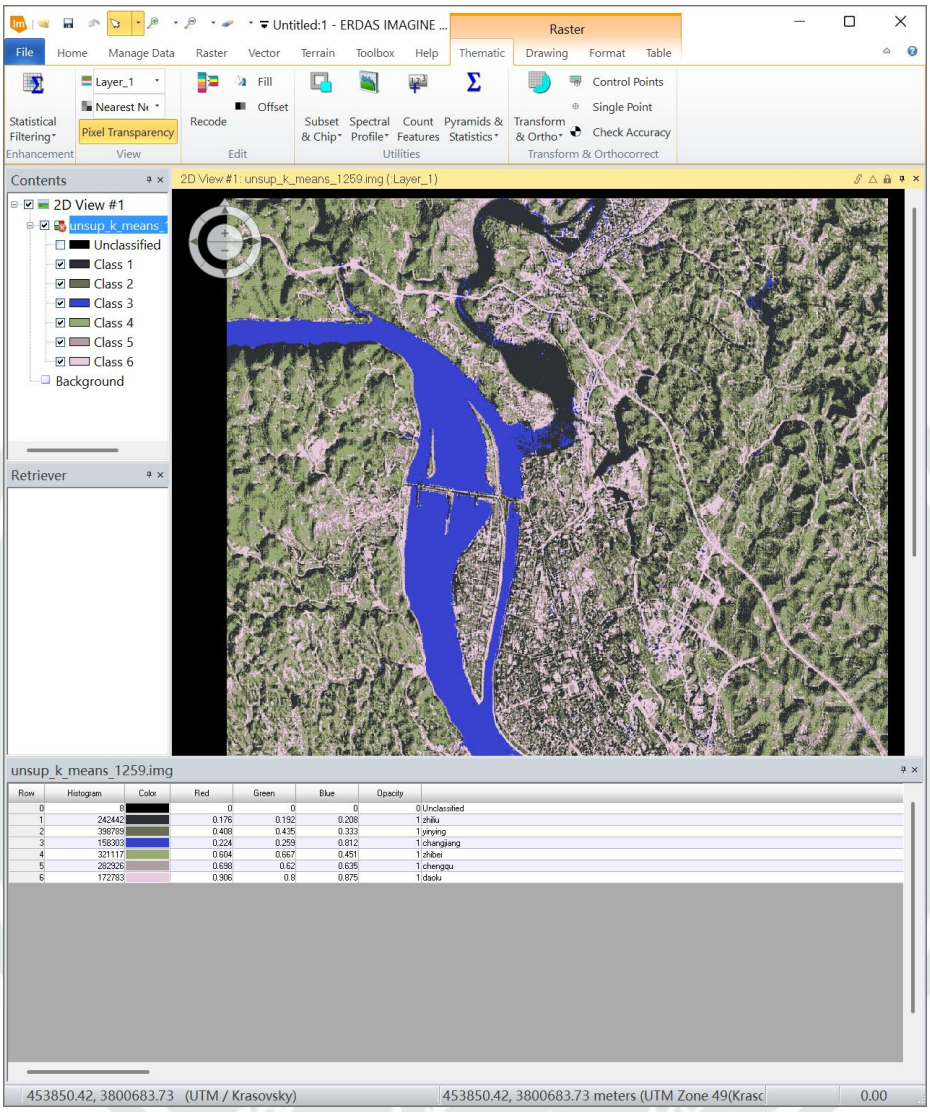


图2. 3. 8-7 在下方查看分类专题图数据的属性表

可以修改属性表中的 Opacity（透明度），首先改成 0，再改成 1，这样就可以发现这个专题图数据的这个类型到底是什么类型，然后修改 Class_Names 栏为其名称。修改完了之后，关掉下面的数据表的时候，可能会弹出一个是否确认保存的窗口（图 2.3.8-8），这个时候需要点击“是”。

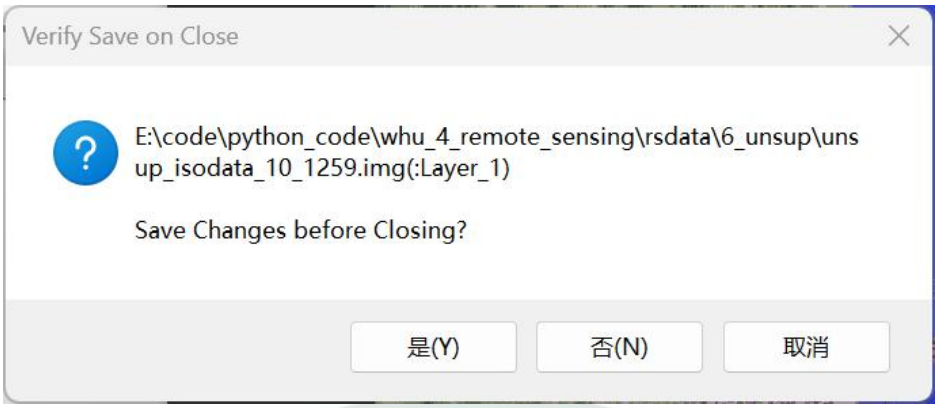


图2.3.8-8 是否保存对数据表的修改

接下来，作者尝试更改非监督分类的参数配置。例如，作者修改 ISODATA 算法的参数配置，设置最大分类数为 10（图 2.3.8-9）：

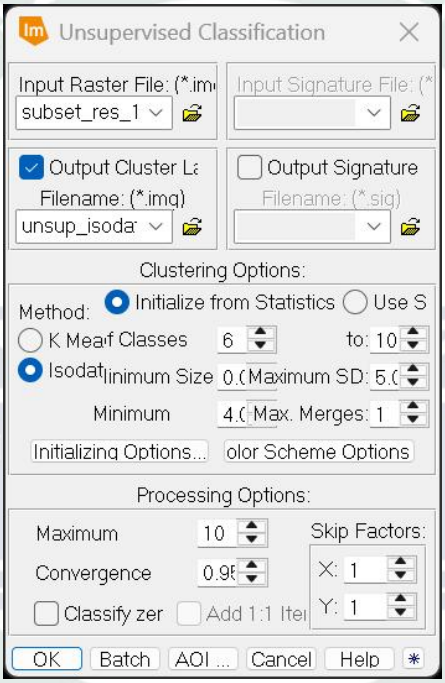


图2.3.8-9 修改ISODATA算法的参数配置

点击 OK，打开输出的 unsup_isodata_10_1259.img ，打开专题图数据的表格，会发现被分成了 10 类，其中有的类别其实属于同一类地物。首先用上文中介绍的方法修改 Class_Names 栏为其真正的名称（图 2.3.8-10），然后使用重编码（Recode）工具来合并类型，注意**一定要使用 Raster-Thematic-Recode 工具而不是其他地方的重编码工具（图 2.3.8-11）**。

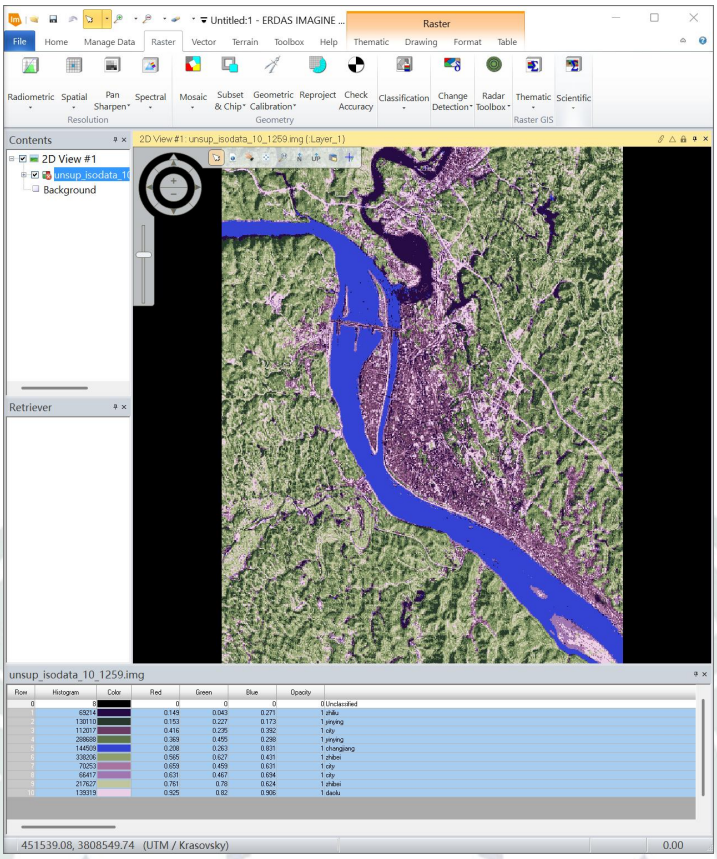


图2.3.8-10 改Class_Names栏为其真正的名称

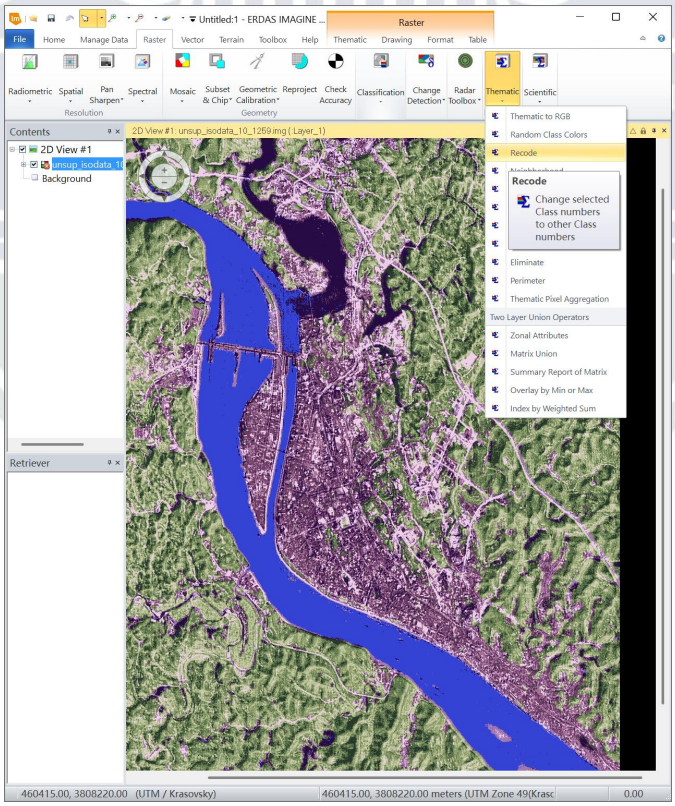


图2.3.8-11 一定要使用Raster-Thematic-Recode工具而不是其他地方的重编码工具

然后打开 Recode 窗口（图 2.3.8-12），点击 Setup Recode，将相同类别的信息的 Value

修改得一样（图 2.3.8-13）：

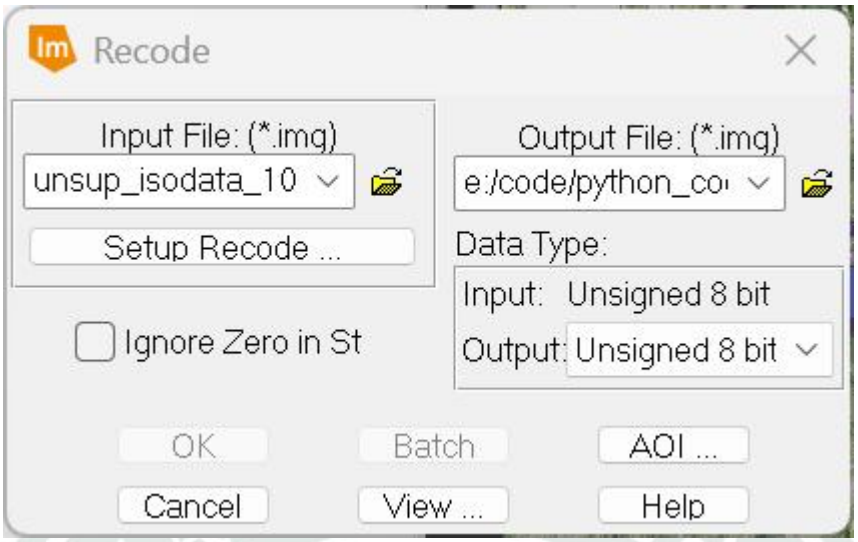


图2.3.8-12 进行Recode配置

Thematic Recode

Value	New Value	Histogram	Red	Green	Blue	Class_Names	Opacity
0	0	8.0	0.000	0.000	0.000	Unclassified	0.0
1	1	69214.0	0.149	0.043	0.271	zhiliu	1.0
2	2	130110.0	0.153	0.227	0.173	yinying	1.0
3	3	112017.0	0.416	0.235	0.392	city	1.0
4	2	288688.0	0.369	0.455	0.298	yinying	1.0
5	4	144509.0	0.208	0.263	0.831	changjiang	1.0
6	5	338206.0	0.565	0.627	0.431	zhibei	1.0
7	3	70253.0	0.659	0.459	0.631	city	1.0
8	3	66417.0	0.631	0.467	0.694	city	1.0
9	5	217627.0	0.761	0.780	0.624	zhibei	1.0
10	6	139319.0	0.925	0.820	0.906	daolu	1.0

图2.3.8-13 编码的修改：将相同类别的信息的Value修改得一样

点击 OK，然后再点击上一个窗口的 OK（注意不要再点击 Setup Recode 了，不然你要重做），然后打开得到 unsup_isodata_10_recode_1259.img，打开数据表，可以看到已经完成重编码（图 2.3.8-14），当然颜色都改变了，需要自己重新设置颜色（图 2.3.8-15）：

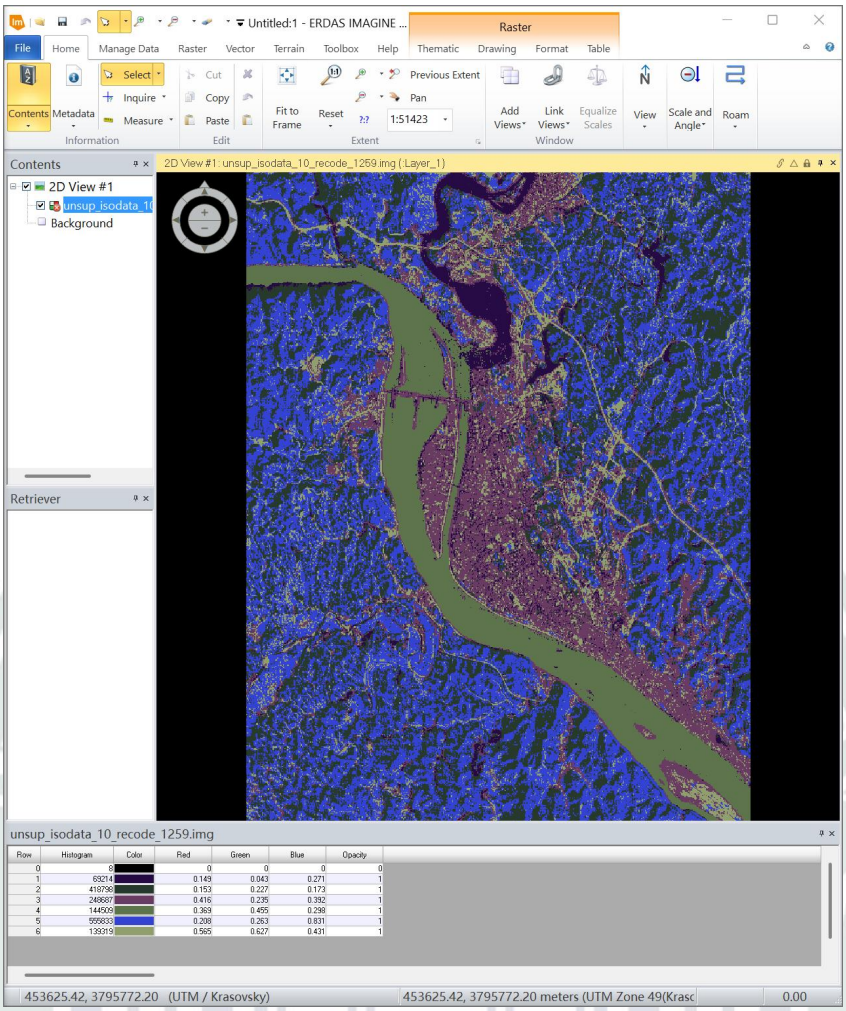


图2.3.8-14 已经完成重编码，但是还需要设置颜色

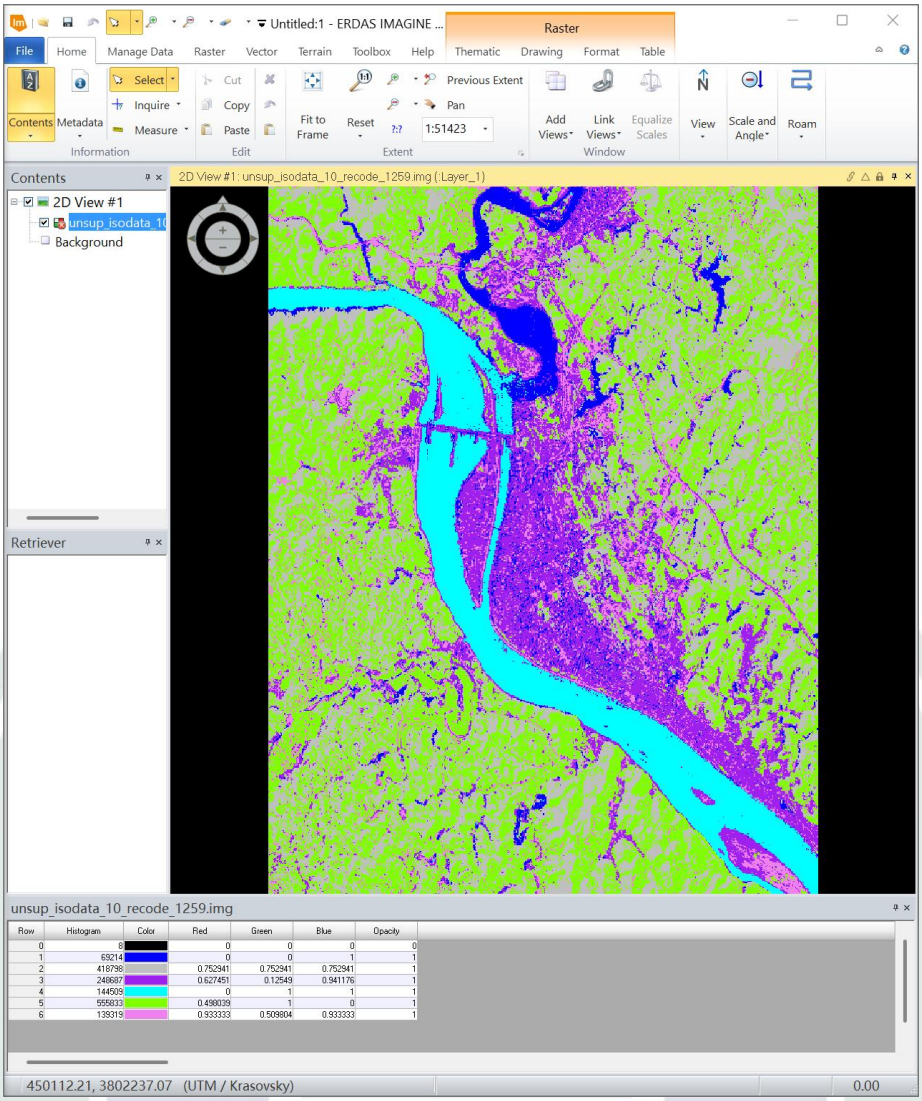


图2.3.8-15 重新设置颜色之后的效果

2.3.9 监督分类

导入2.3.7所得的subset_res_1259.img，点击Raster-Classification-Supervised-Signature Editor，打开如下界面（图 2.3.9-1）：

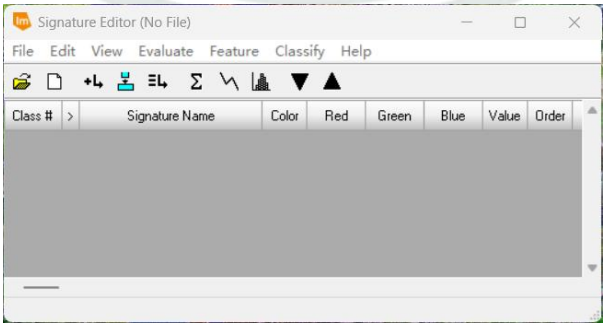


图2.3.9-1 Signature Editor的界面

点击程序主界面上方的 Raster-Drawing，点击 图标开始绘制样区，然后点击 Signature Editor 窗口中的 将某一类地物的样区添加到 Signature Editor 中。所选择的这一类的地物样区的像素数量应足够多，并且具有代表性，能够反映这一类地物样区的所有代表区域，即使其特征可能有所不同。

选择完同一类地物的所有样区之后，按住 Shift 选中这一类地物的所有样区，点击 可以将得到的样区区域合并，合并结果出现在刚刚按住 Shift 选中这一类地物的所有样区的下面。合并之后可以右键刚刚按住 Shift 选中这一类地物的所有样区的第一列（Class #），选择 Delete Selection，把合并之前的结果删除。

选择完成之后，尽可能保证 I 那一列是“√”，也就是样区数据的协方差可逆，这样后续的一些监督分类算法才可以适用。

最终得到的样区选择结果如图 2.3.9-2：

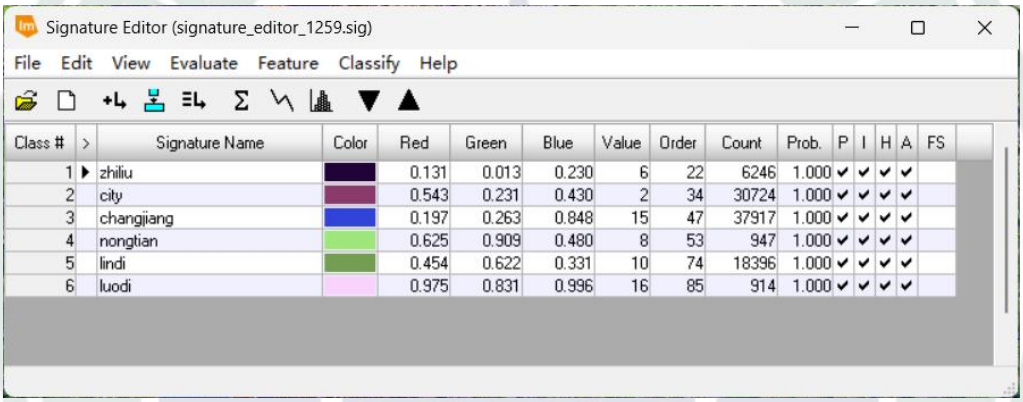


图2.3.9-2 最终得到的样区选择结果

在 Signature Editor 关闭窗口之前，可以通过 Signature Editor 窗口中 File-Save As 将选择的样区信息.sig 文件保存到本地。下次打开 Signature Editor 窗口的时候，可以通过 File-Open 打开上次保存的.sig 文件（图 2.3.9-3）

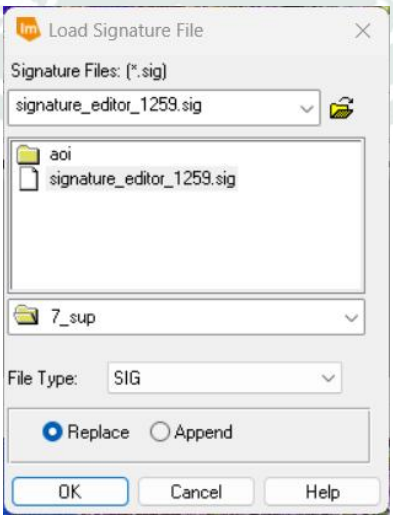


图2.3.9-3 通过File-Open打开上次保存的.sig文件

此外，还可以将每个地物绘制的 AOI 保存到本地：右键主界面 Contexts 里面的 AOI 图层，保存图层为本地的 AOI 文件即可。

关闭 Signature Editor 窗口之后，点击 Raster-Classification-Supervised Classification，在打开的窗口中设置参数（图2.3.9-4），其中修改Decision Rules中的Non-parametric Rules（非参数规则，监督分类中不依赖统计分布假设的决策方法）的选项即可修改所用的监督分类的算法。

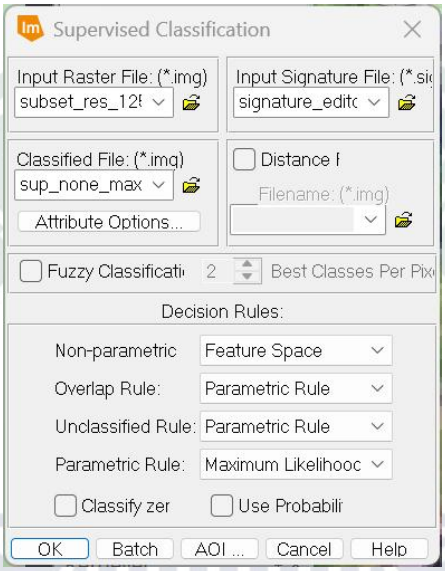


图2.3.9-4 设置监督分类的配置

作者的 Non-parametric Rules 选项使用了 None、Parallelepiped 和 Feature Space 这三种监督分类的方法。None（无规则）仅使用最大似然分类法（Maximum Likelihood Classifier, MLC），这是一种基于统计的参数化方法，假设每个类别的光谱数据服从多元正态分布，通过计算像元属于各类别的概率，选择概率最高的类别作为分类结果，需要足够的训练样本以准确估计均值、方差和协方差矩阵，当数据分布接近正态分布时效果最佳；Parallelepiped（平行六面体法）通过训练样本的光谱值范围定义多维的“盒子”（平行六面体），若像元的光谱值落在某类别的盒子内则被归为该类别，若同时落入多个盒子则按照下面其它参数设置的规则被标记为“未分类”或按优先级分配；Feature Space（特征空间法）基于像元在特征空间中的位置进行分类，特征空间由选择的波段（或主成分）构成，当类别在特定波段组合下可分性高时（如植被与水体在 NDVI 波段）效果较好。

为了首先确保证编码编号连续，导入得到的 sup_none_maximum_likelihood_1259.img，使用重编码（Recode）工具将监督分类方法得到的分类结果进行重编码（图 2.3.9-5），注意**一定要使用 Raster-Thematic-Recode 工具而不是其他地方的重编码工具（图 2.3.9-6）：**

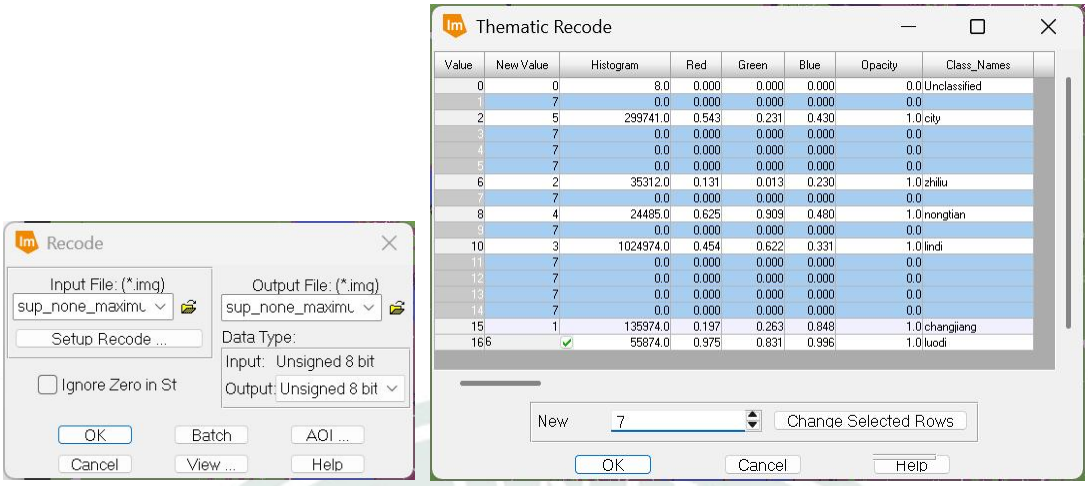


图2.3.9-5 重编码操作

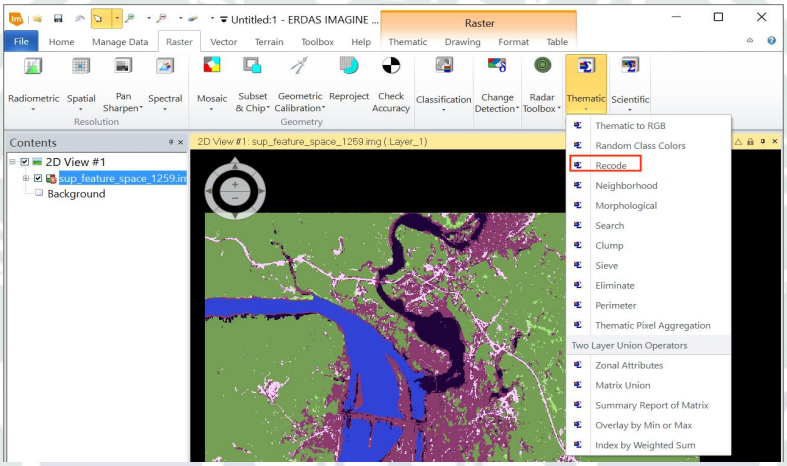


图2.3.9-6 一定要使用Raster-Thematic-Recode工具而不是其他地方的重编码工具

然后打开得到的 sup_none_maximum_likelihood_1259_recode.img，打开数据表，可以看到已经完成重编码（图 2.3.9-7），当然颜色都改变了，需要自己重新设置颜色：

sup_none_maximum _likelihood_1259_recode.img

Row	Histogram	Color	Red	Green	Blue	Opacity
0	8		0	0	0	0
1	136415		0	1	1	1
2	33697		0	0	1	1
3	1022470		0	0.392157	0	1
4	28559		0	1	0	1
5	299819		0.627451	0.12549	0.941176	1
6	55400		0.933333	0.509804	0.933333	1

图2.3.9-7 重编码完成之后的效果，需要自己重新设置颜色

类似地重编码使用其他方法得到的监督分类的结果专题图。图 2.3.9-8~2.3.9-10 展示了不同方法的分类结果（已经完成重编码）：

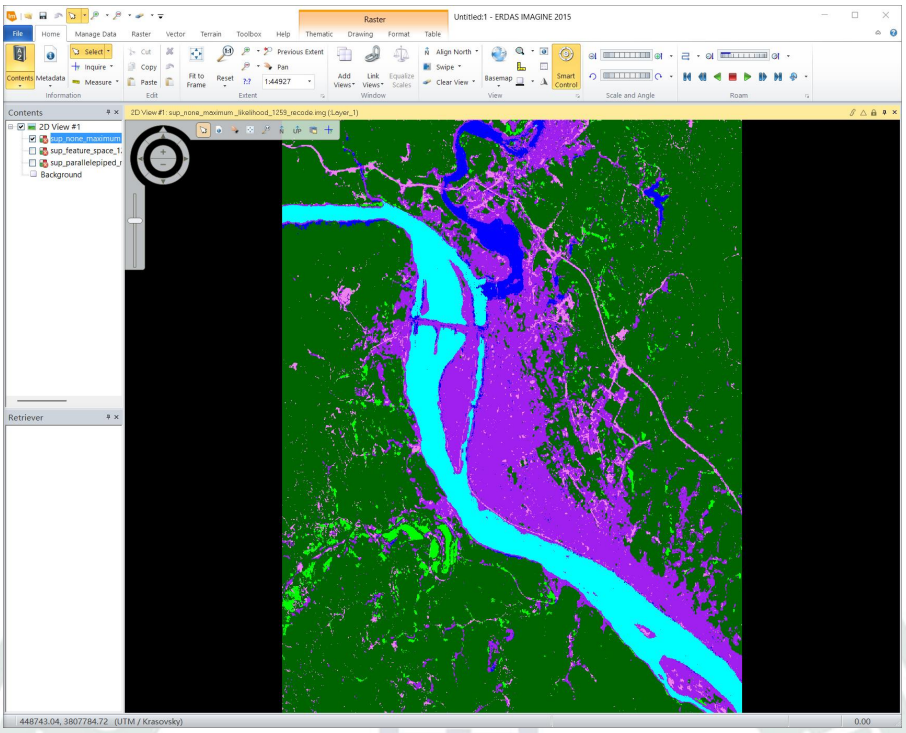


图2.3.9-8 重编码后的最大似然分类法（Maximum Likelihood Classifier, MLC）的监督分类效果

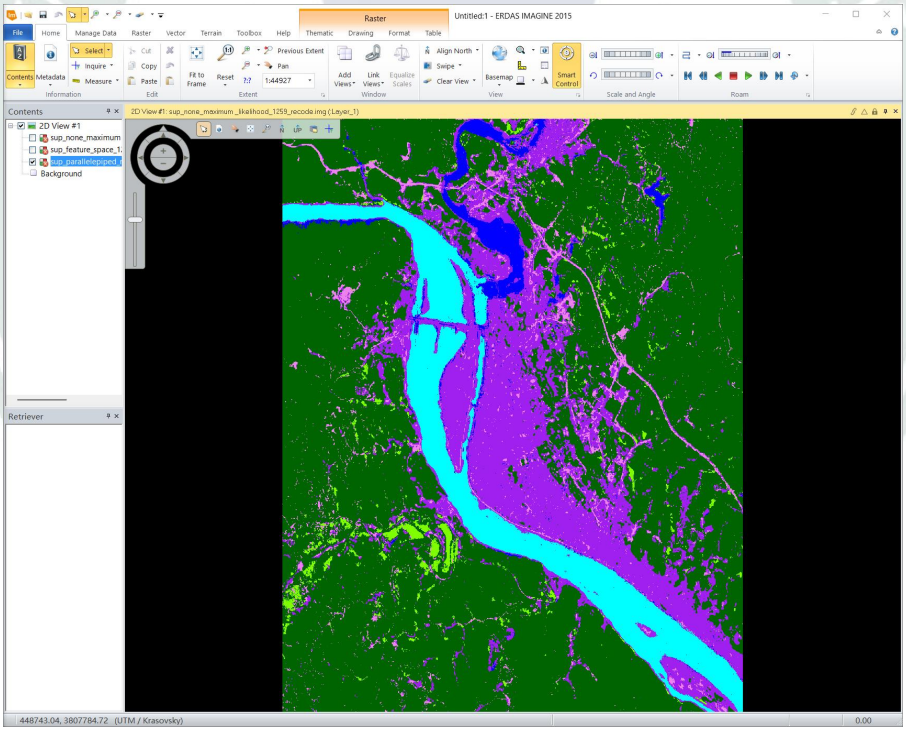


图2.3.9-9 重编码后的平行六面体法（Parallelepiped）的监督分类效果

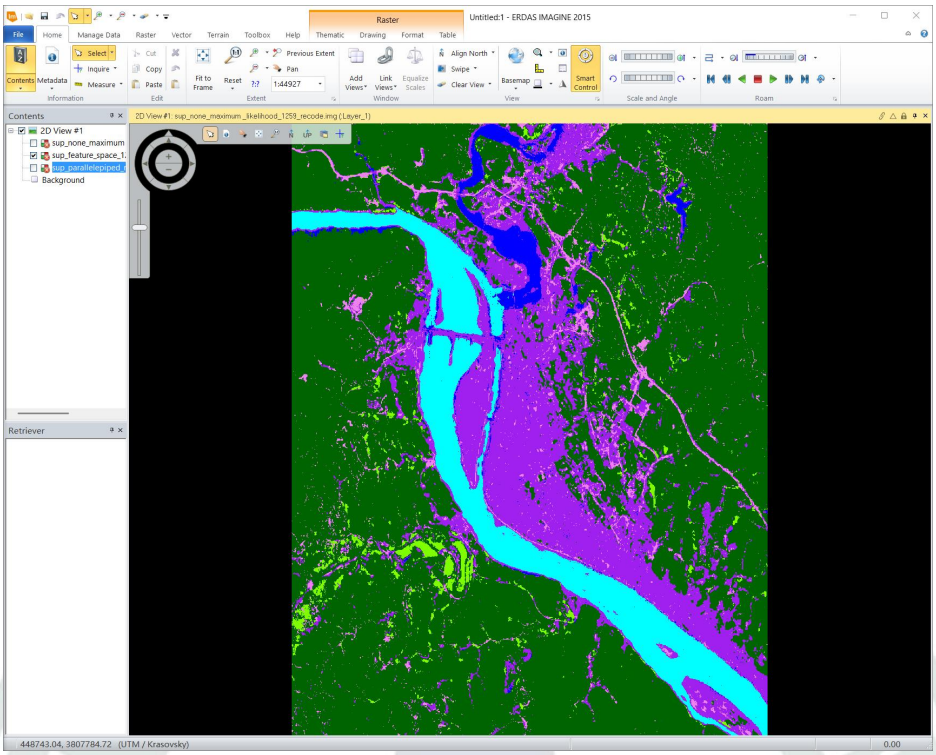


图2.3.9-10 重编码后的特征空间法（Feature Space）的监督分类效果

2.3.10 分类精度的评定

导入 2.3.7 所得的 subset_res_1259.img，然后点击 Raster-Classification-Supervised-Accuracy Assessment，打开如图 2.3.10-1 所示的窗口：

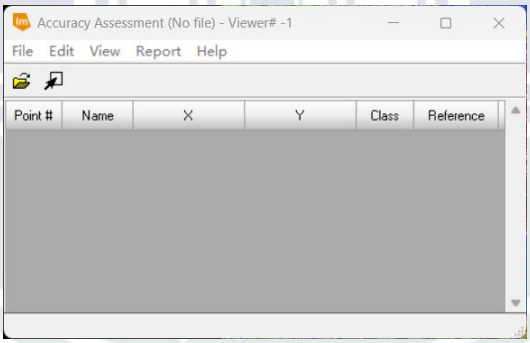


图2.3.10-1 Accuracy Assessment的窗口

点击 File-Open，打开 2.3.9 中所得重编码后的监督分类结果 sup_none_maximum_likelihood_1259_recode.img（图 2.3.10-2）。此时窗口不会有任何变化，但是窗口的表格已经和这个监督分类结果图关联起来了。

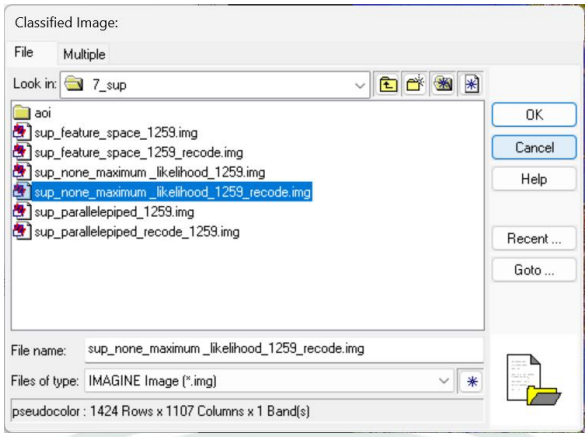


图2. 3. 10-2 打开重编码后的监督分类结果

点击上方的 Edit-Select Viewer，提示需要选择视图来显示随机产生的检查点（图 2.3.10-3），此时在出现图 2.3.10-4 的提示弹窗之后，点击一下程序主界面的视图，后面随机产生的检查点将会在这里选择的视图上展点，用于目视判读。

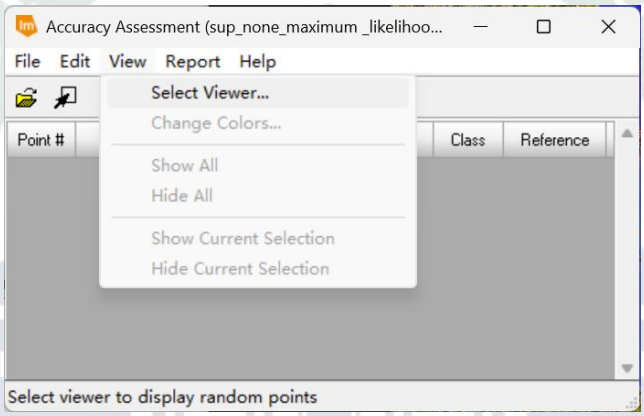


图2. 3. 10-3 选择显示随机产生的检查点的视图

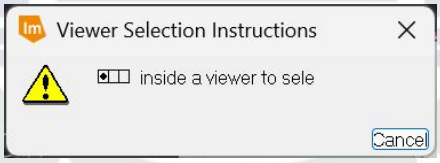


图2. 3. 10-4 选择显示随机产生的检查点的视图的提示弹窗

选择完视图之后图 2.3.10-4 所示的弹窗会自动关闭，此时关闭图 3.2.30-3 所示的 Select Viewer 的弹窗，回到 Accuracy Assessment 窗口，点击 View 可以发现 View 下方的选项由原本的禁用状态（图 2.3.10-3）变为启用状态了（图 2.3.10-5）：

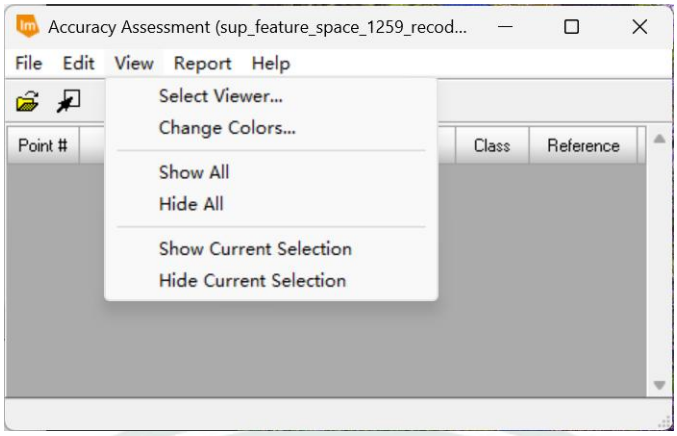


图2.3.10-5 选择完视图之后，View下方的选项由原本的禁用状态变为启用状态了

然后使用 Edit-Create/Add Random Points 来产生随机的检查点（图 3.2.30-6）：

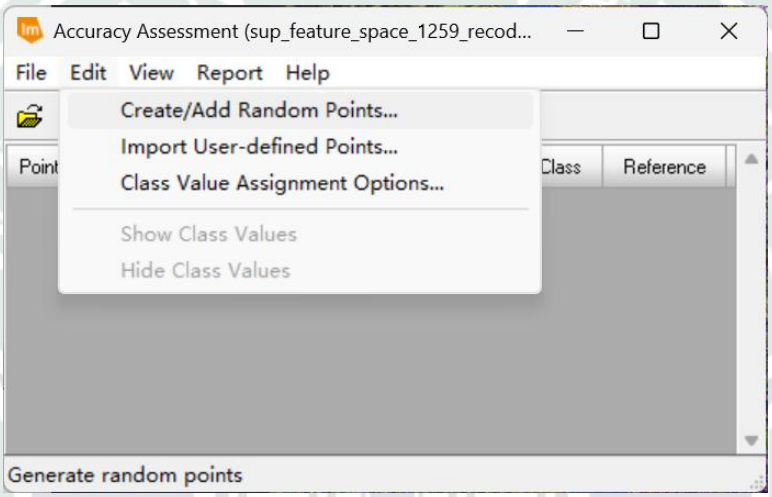


图2.3.10-6 使用Edit-Create/Add Random Points来产生随机的检查点

点击 Edit-Create/Add Random Points 之后，在弹出的 Add Random Points 窗口中进行配置。我一开始的配置如图 2.3.10-7 所示：

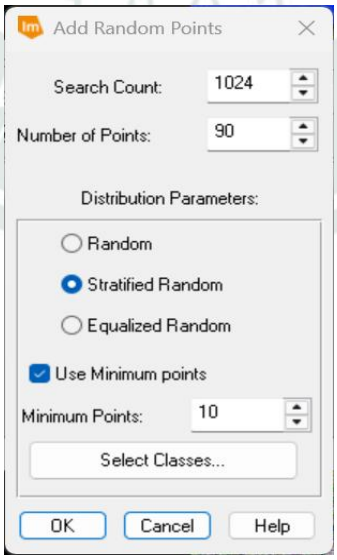
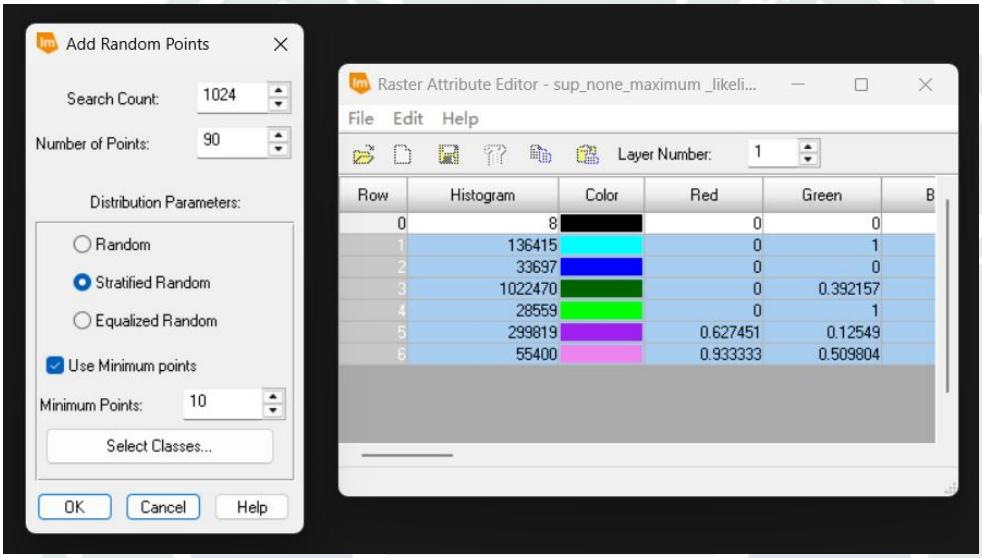


图2.3.10-7 我一开始的Add Random Points配置

其中，Search Point 是在图像中选取的用于搜索每一类地物的像素数量；Number of Points 是生成的随机检查点的总数（要求监督分类时每有一类地物，所对应的随机检查点的数量增加 15 个，我在监督分类时设置了 6 类地物，那么所对应的随机检查点的数量为 15×6=90 个）；中间的 Distribution Parameters 下方的选项是产生随机检查点的方法，建议选择 Stratified Random（按照像素数量加权平均）或者 Equalized Random（绝对平均），其中对于前者，Minimum Points 是每个类别的最少点数，可以保证对于那些像素数量占比比较小的地物类型也可以保证产生一定数量的随机检查点。

此外，还需要点击 Select Classes 按钮，设置需要生成随机检查点的类别（如图 2.3.10-8，一些其他的类别就不需要生成了）：按住 Shift 选中需要生成检查点的类别，直接点击 Add Random Points 窗口中的 OK 即可。



可能会遇到如图 2.3.10-9、2.3.10-10 所示的问题：

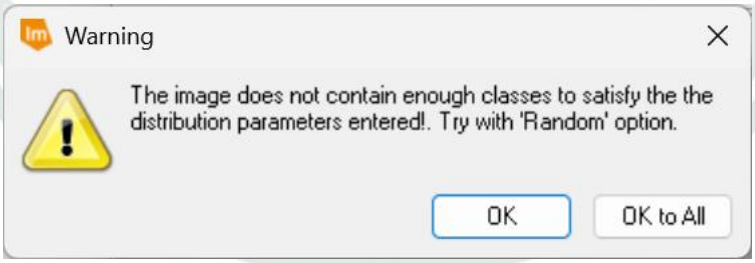


图2. 3. 10-9 提示：图像不包含足够的点

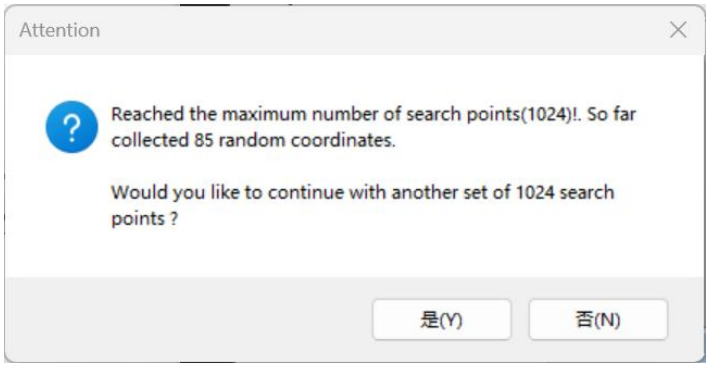


图2.3.10-10 提示：没有找到足够的满足条件的随机检查点

此时更改配置：首先增大 Search Point 的数量，这样使得模型能够在更大的范围内搜索符合要求的像元，如果 Search Point 太小，由于有的类别的地物占整幅影像的比例较小，可能会出现找不到足够的满足条件的随机检查点的情况；此外可以适当减小 Minimum Points，这样产生随机检查点的条件可以不那么严苛。

点击 OK，等待运行完毕（图 2.3.10-11）：

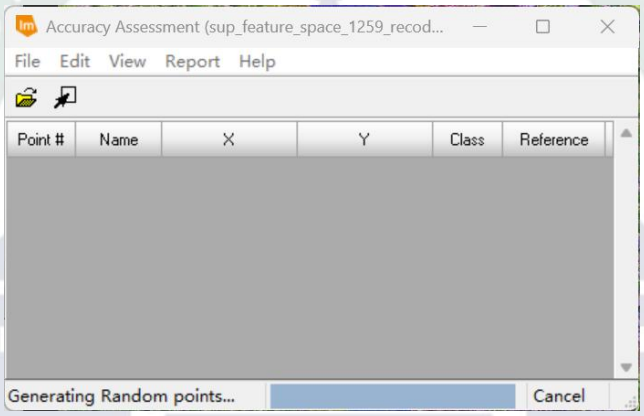


图2.3.10-11 由于Search Point的数量变得比较大，因此可能需要加载一段时间

运行完毕后，得到的满足条件的随机检查点的数据表格如图 2.3.10-12 所示：

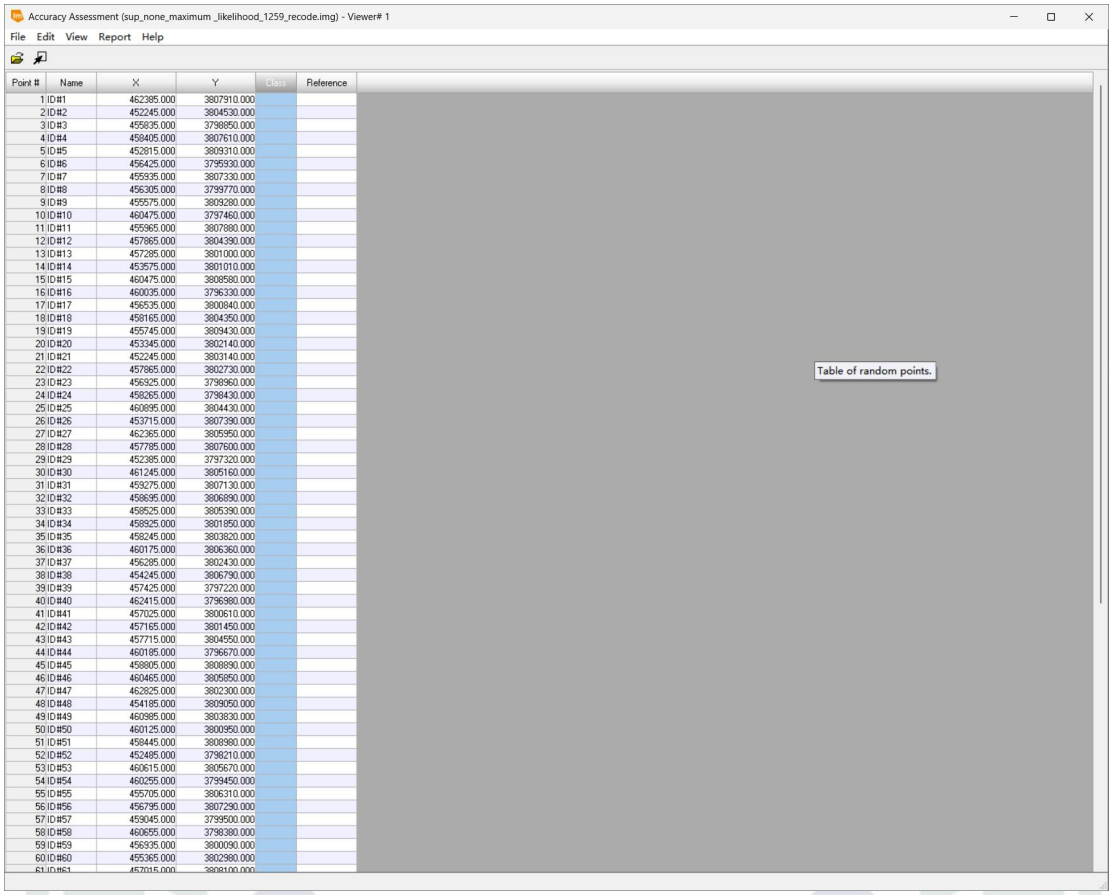


图2.3.10-12 得到的满足条件的随机检查点的数据表格（注意到此时Class那一列还是空的）

可以选择 File-Save as Annotation（图 2.3.10-13），将产生的检查点保存为可以在图层中显示的要素（图 2.3.10-14）：

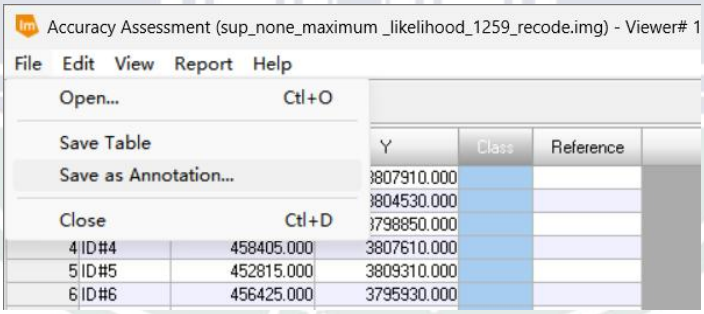


图2.3.10-13 Save as Annotation操作

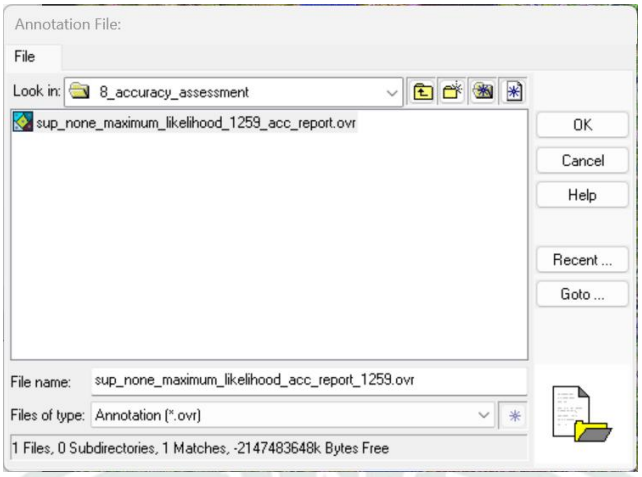


图2.3.10-14 将产生的检查点保存为可以在图层中显示的要素

可以选择 View-Show Class Values（图 2.3.10-15），将表格中的 Class 一列（表示监督分类得到的结果）呈现出来：

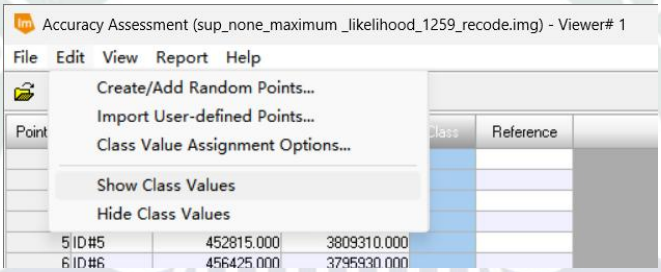


图2.3.10-15 将表格中的Class一列（表示监督分类得到的结果）呈现出来

接下来就需要进行目视判读，得到遥感影像像元类别的真值（Ground Truth）了。如图 2.3.10-16，可以选择 View-Show All 将表格中的所有点展示在上文所述的 Select Viewer 所选择的视图中，或者也可以长按 Shift 选中几列数据，然后使用 View-Show Current Selection 将选中的这几个点的数据展示在上文所述的 Select Viewer 所选择的视图中。

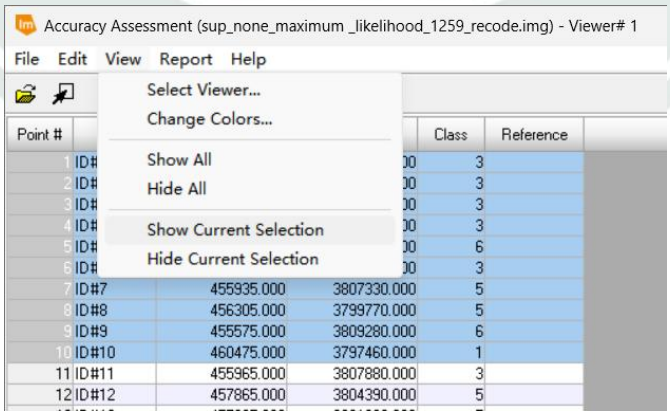


图2.3.10-16 将表格中的点展示在上文所述的Select Viewer所选择的视图中

为了目视判读的时候能够更加清晰地看到需要判读的点，可以在 View-Change Colors 里面设置已经进行目视判读并填写标注（Reference）和未进行目视判读、还没有

填写标注的点的颜色，加以区分，并使其在图上更加醒目（图 2.3.10-17）：

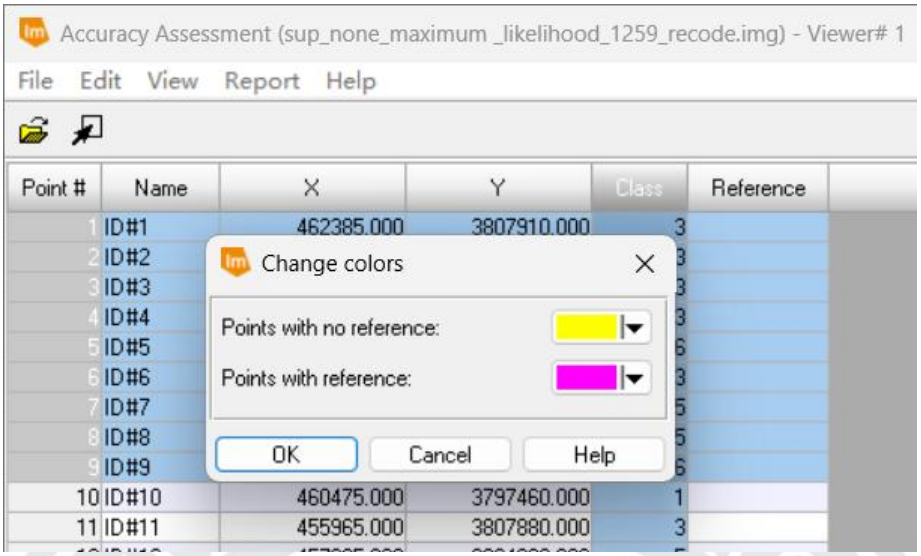


图2.3.10-17 在View-Change Colors里面设置已经进行目视判读并填写标注（Reference）和未进行目视判读、还没有填写标注的点的颜色

然后就要进行目视判读了：需要看着程序主界面上展出的点，根据视图中的 subset_res_1259.img（也可以参考下发的高分辨率的宜昌实验区.img）来判断到底是哪一类地物，将其对应的地物类别编号填写在 Accuracy Assessment 窗口中数据表格的 Reference 那一列中（图 2.3.10-18）。对应的类别编号是在 2.3.9 中对监督分类的结果的重编码时决定的（不同的监督分类方法的地物类别编号应该统一），我使用的类别编号为：1-长江，2-支出，3-林地，4-农田，5-城市（市区），6-裸地（如道路）。

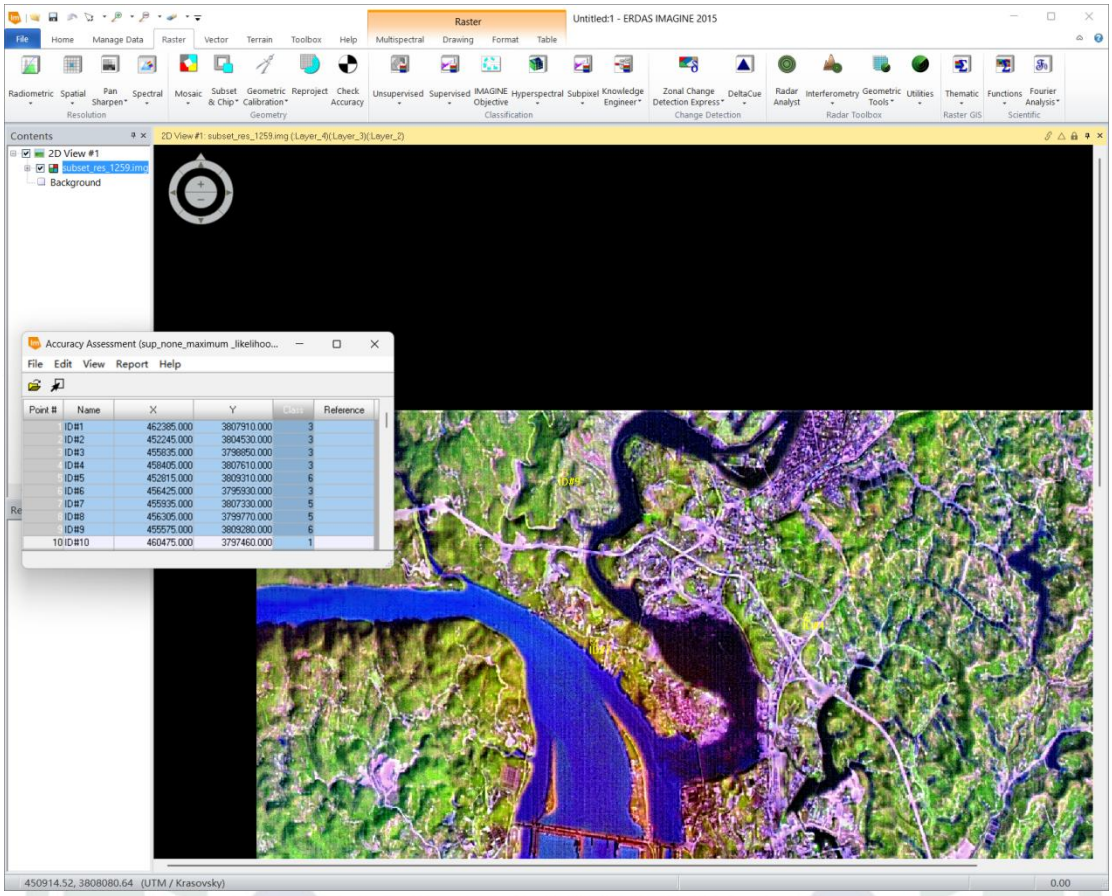


图2.3.10-18 目视判读：看着程序主界面上展出的点，根据视图中的影像来判断到底是哪一类地物，将其对应的地物类别编号填写在Accuracy Assessment窗口中数据表格的Reference那一列中

完成目视判读之后，请**一定要点击 File-Save Table 将自己辛辛苦苦判读的结果保存（关联）到监督分类的结果.img 中**（图 2.3.10-19）！这样下次通过 File-Open 打开这张.img 的时候，下方的数据表格就会自动呈现出来。

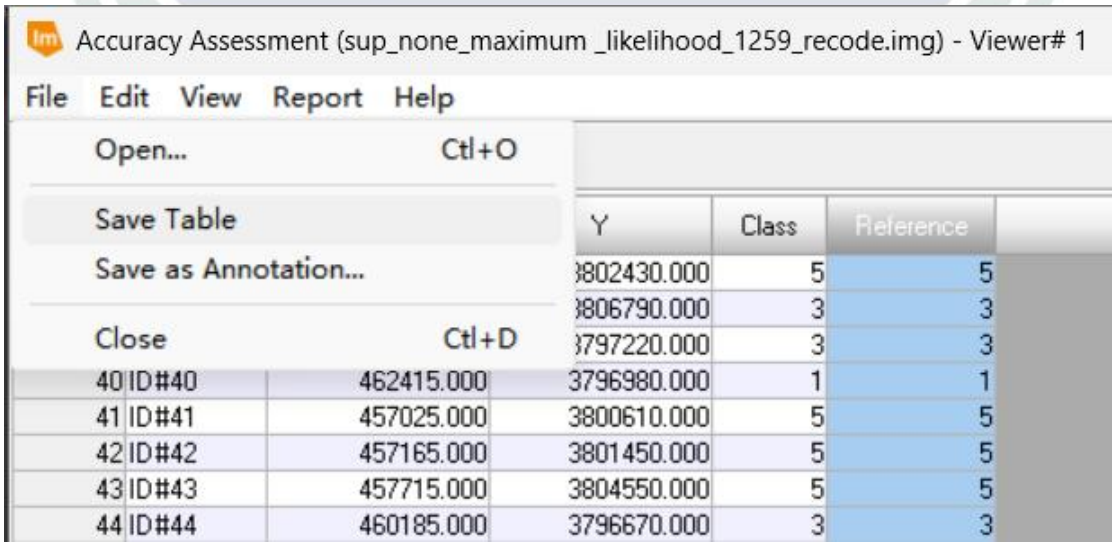
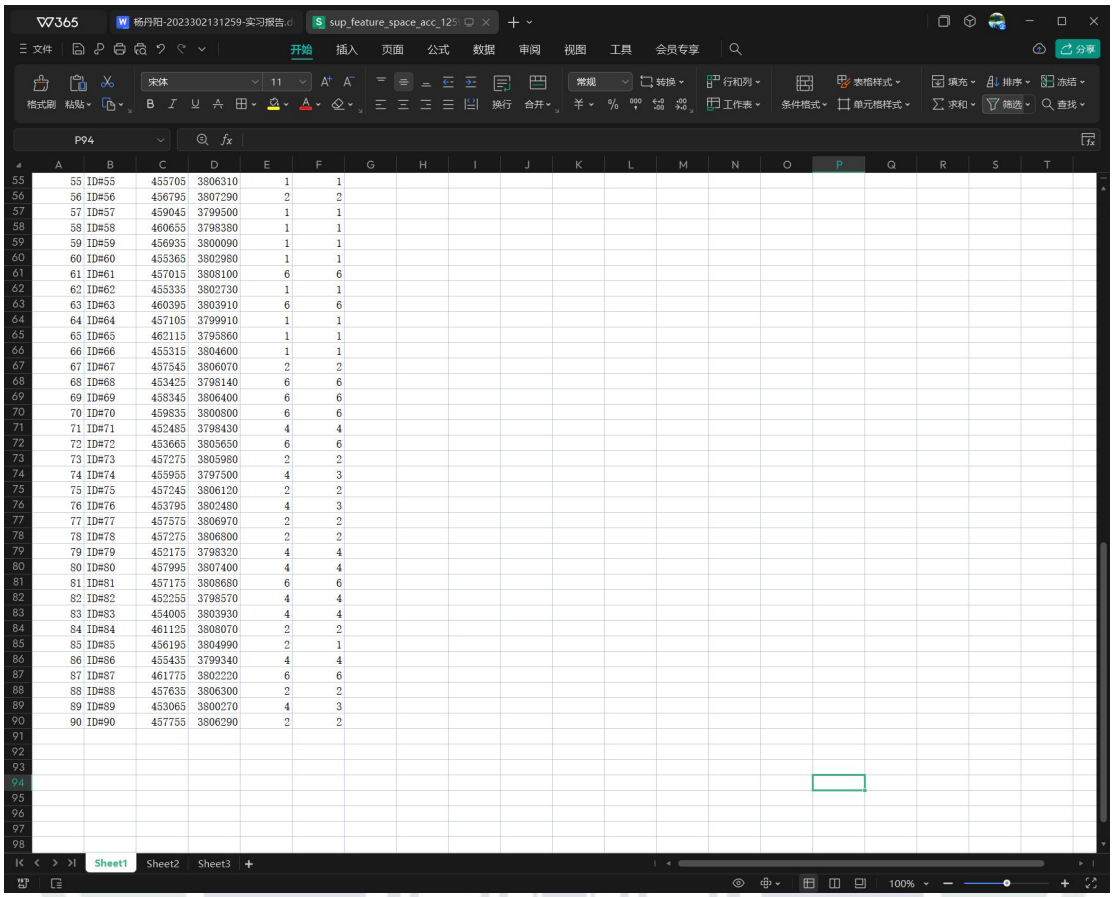


图2.3.10-19 完成目视判读之后点击File-Save Table将判读的结果表格保存（关联）到监督分类的结果.img中

此外，还需要**右键点击每一列的标题，选择 Copy，将所有数据拷贝到自己创建的**

一个 Excel 表格中（图 2.3.10-20）。这样一是为了保留备用、以防万一；二是对不同的监督分类的结果进行精度评定需要使用同一组随机检查点数据（包括坐标和目视判读的参考值）。



55	ID#55	455705	3806310	1	1	
56	ID#56	456795	3807290	2	2	
57	ID#57	459045	3799500	1	1	
58	ID#58	460655	3798380	1	1	
59	ID#59	456935	3800090	1	1	
60	ID#60	455365	3802980	1	1	
61	ID#61	457015	3808100	6	6	
62	ID#62	455335	3802730	1	1	
63	ID#63	460395	3803910	6	6	
64	ID#64	457105	3799910	1	1	
65	ID#65	462115	3795860	1	1	
66	ID#66	455315	3804600	1	1	
67	ID#67	457545	3806070	2	2	
68	ID#68	453425	3798140	6	6	
69	ID#69	458345	3806400	6	6	
70	ID#70	459835	3800800	6	6	
71	ID#71	452485	3798430	4	4	
72	ID#72	453665	3805650	6	6	
73	ID#73	457275	3805980	2	2	
74	ID#74	455955	3797500	4	3	
75	ID#75	457245	3806120	2	2	
76	ID#76	453795	3802480	4	3	
77	ID#77	457575	3806970	2	2	
78	ID#78	457275	3806800	2	2	
79	ID#79	452175	3798320	4	4	
80	ID#80	457995	3807400	4	4	
81	ID#81	457175	3808680	6	6	
82	ID#82	452255	3798570	4	4	
83	ID#83	454005	3803930	4	4	
84	ID#84	461125	3808070	2	2	
85	ID#85	456195	3804990	2	1	
86	ID#86	455435	3799340	4	4	
87	ID#87	461775	3802220	6	6	
88	ID#88	457635	3806300	2	2	
89	ID#89	453065	3800270	4	3	
90	ID#90	457755	3806290	2	2	

图2.3.10-20 右键点击每一列的标题，选择Copy，将所有数据拷贝到自己创建的一个Excel表格中

然后先关闭 Accuracy Assessment 表格**（关闭之前一定要完成图 2.3.10-19、图 2.3.10-20 所示的保存操作！）**，然后再次打开 Raster-Classification-Supervised-Accuracy Assessment，点击 Report-Accuracy Report（图 2.3.10-21）

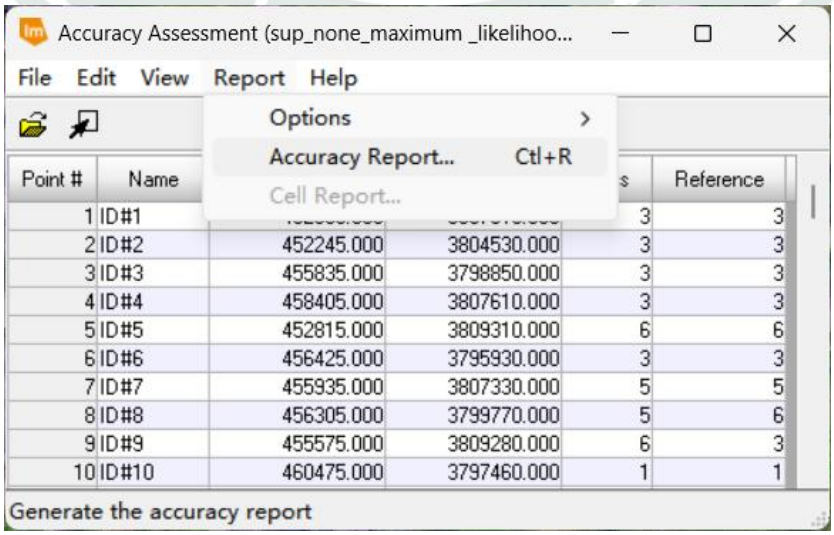
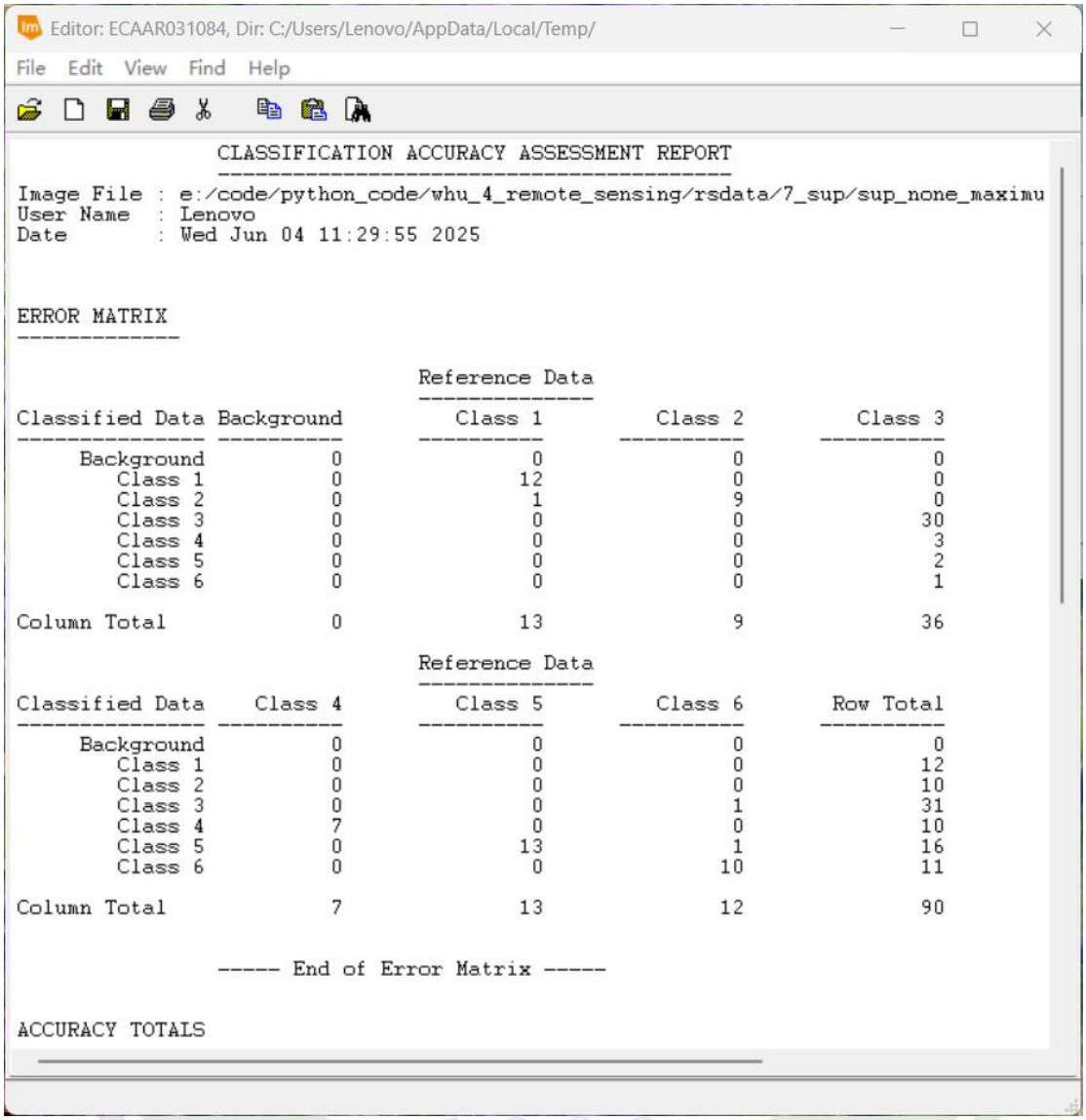


图2.3.10-21 使用Report-Accuracy Report生成精度评定报告

即可得到精度评定的报告（图 2.3.10-22）：



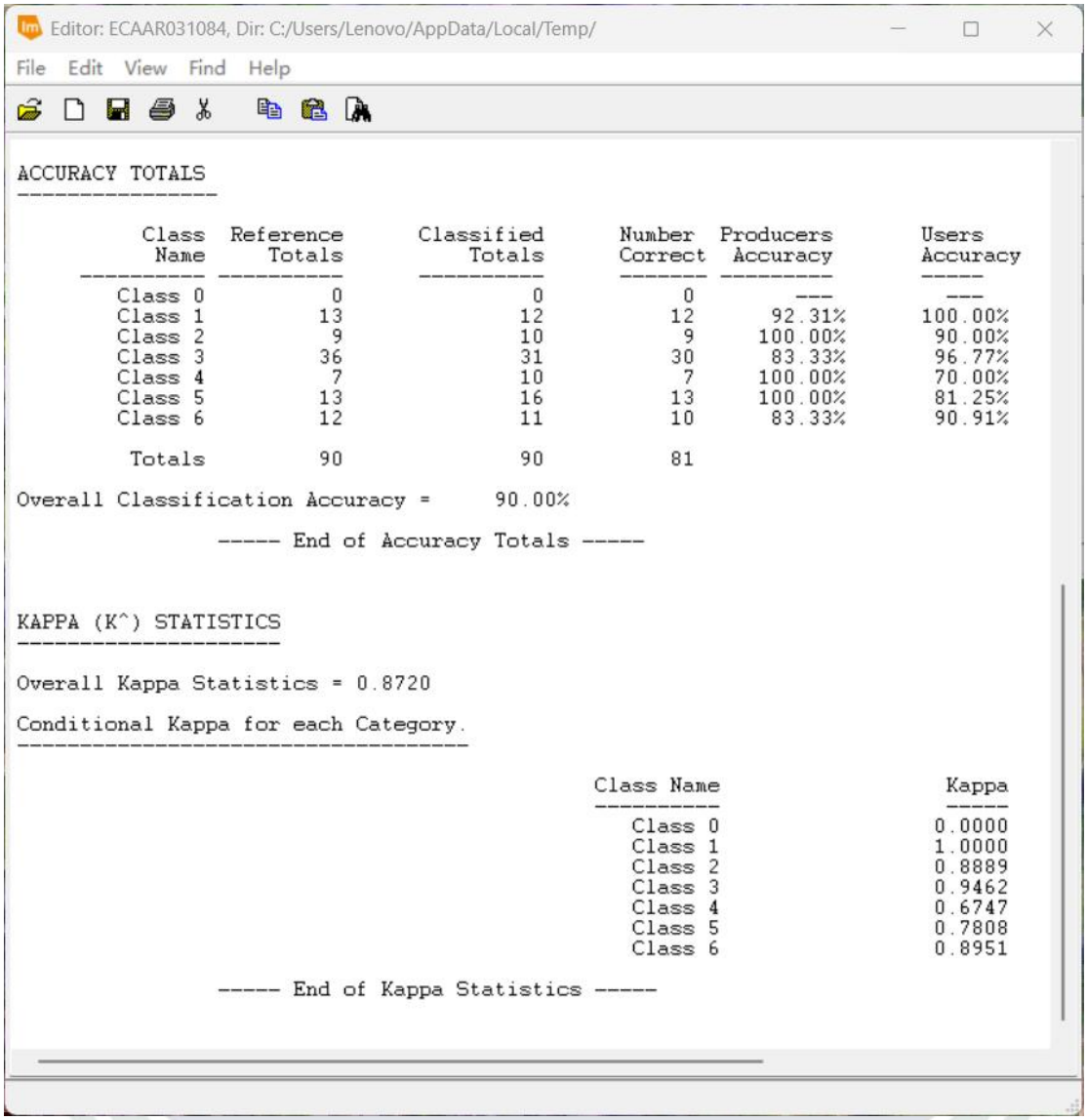


图2.3.10-22 得到的最大似然分类法（Maximum Likelihood Classifier, MLC）的精度评定的报告

点击 File-Save As 还可以将这份报告保存为文本文件等格式（图 2.3.10-23）：

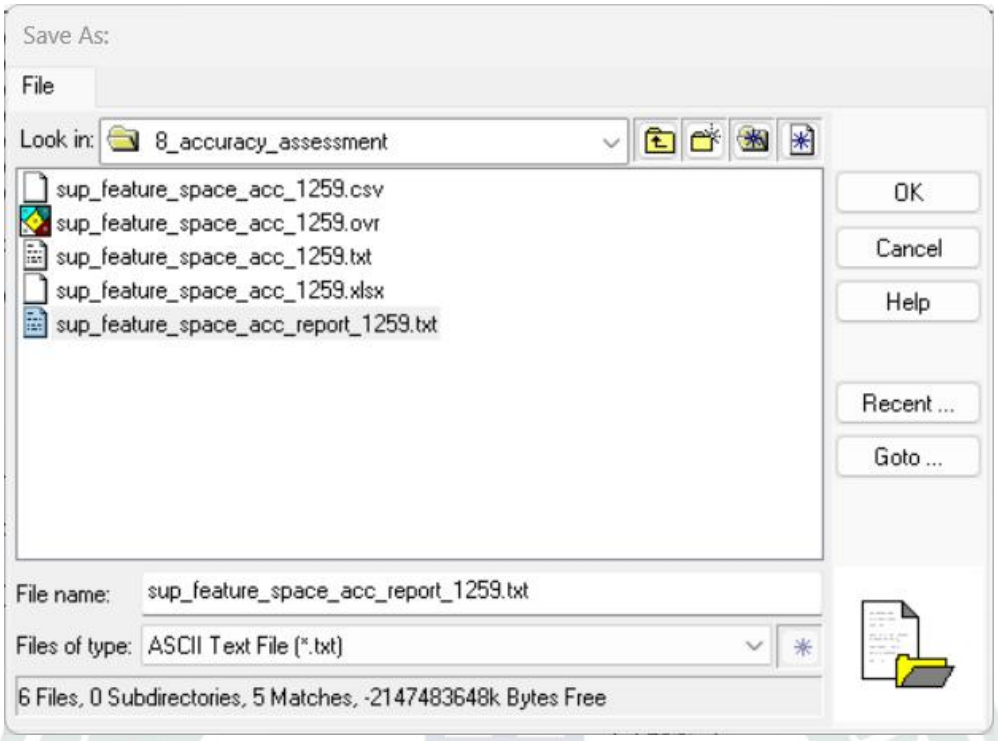


图2.3.10-23 点击File-Save As将精度评定报告保存为文本文件等格式

接下来关闭精度评定报告的窗口，点击 File-Open，会弹出提示（图 2.3.10-24），选择“是”打开使用其他监督分类的方法得到的.img 文件：

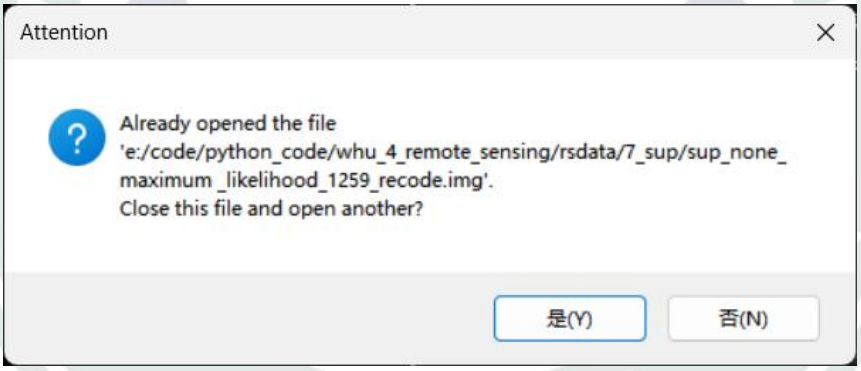


图2.3.10-24 提示是否需要打开另一份监督分类结果的.img文件

然后就要打开刚刚说要保存的那张 Excel 表格（图 2.3.10-20）了，打开之后 Ctrl A,Ctrl C，新建一个.txt 文件，Ctrl V，记事本中的内容如图 2.3.10-25 所示：

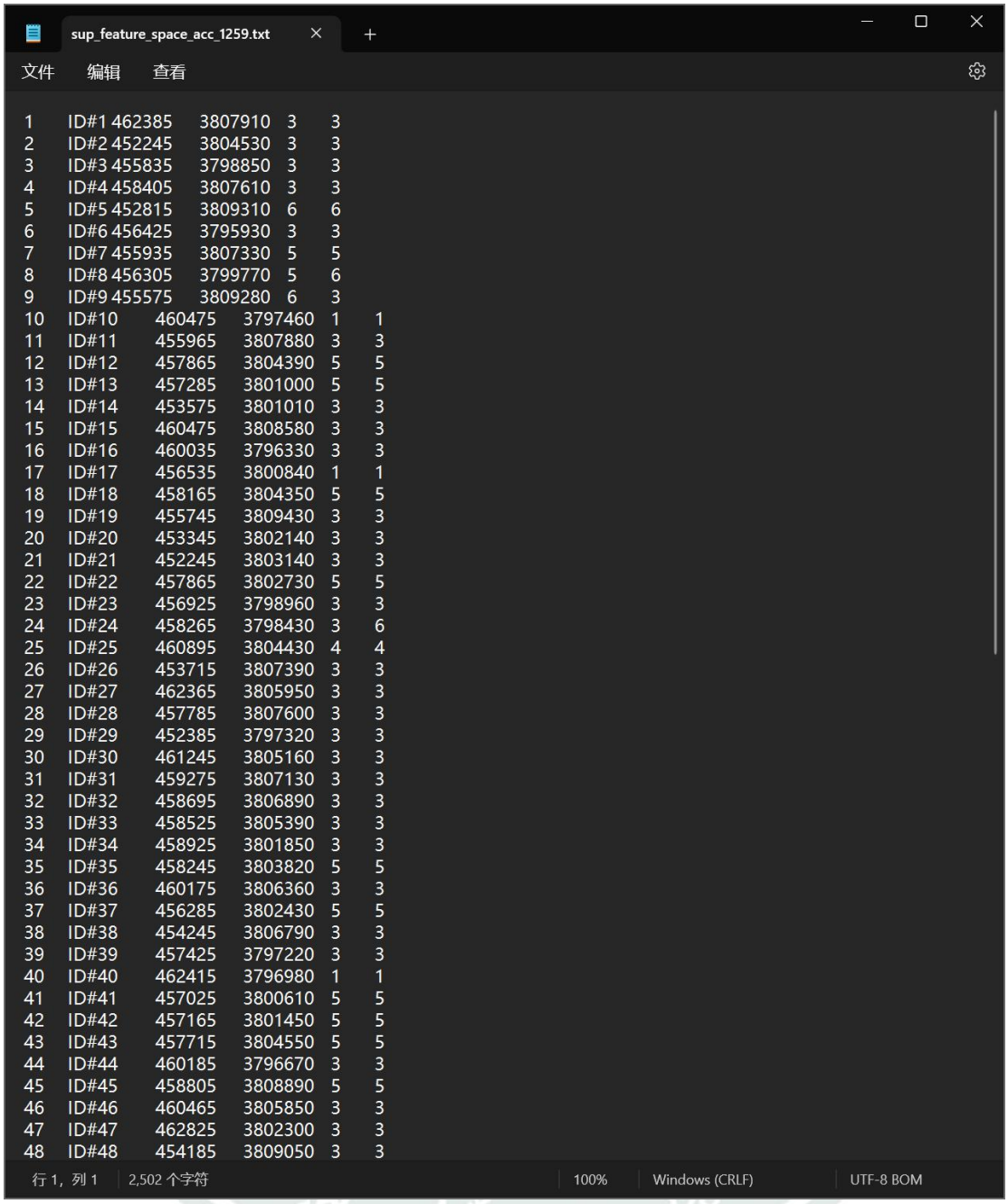


图2.3.10-25 记事本中随机检查点的信息

回到刚刚的 Accuracy Assessment 窗口，选择 Import User-defined Points（图 2.3.10-26）：

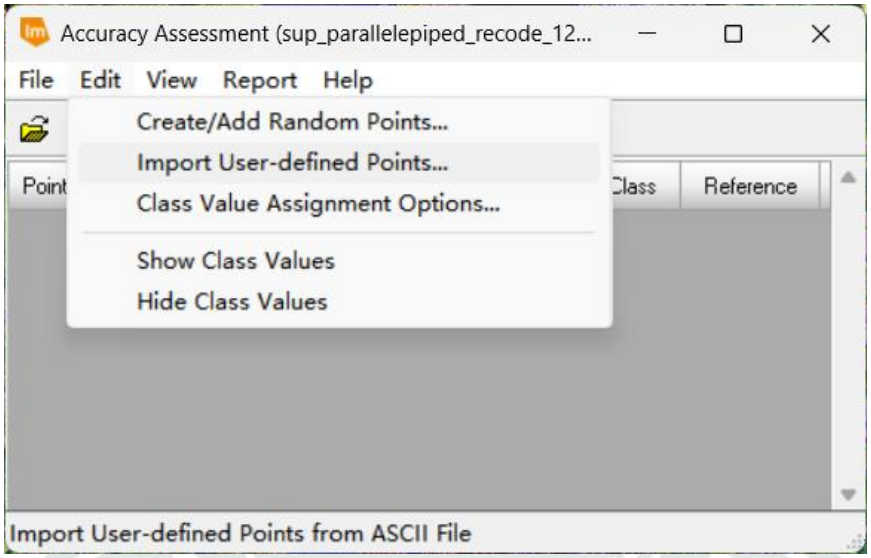


图2.3.10-26 Import User-defined Points导入之前定义的检查点

在弹出的窗口中（图 2.3.10-27），设置分隔符（Separater Character）为 WhiteSpace（就是在记事中看见的\t 制表符），设置下方的 Column Mapping 的 X、Y 分别为 3、4（就是记事本中是 X、Y 的列）。

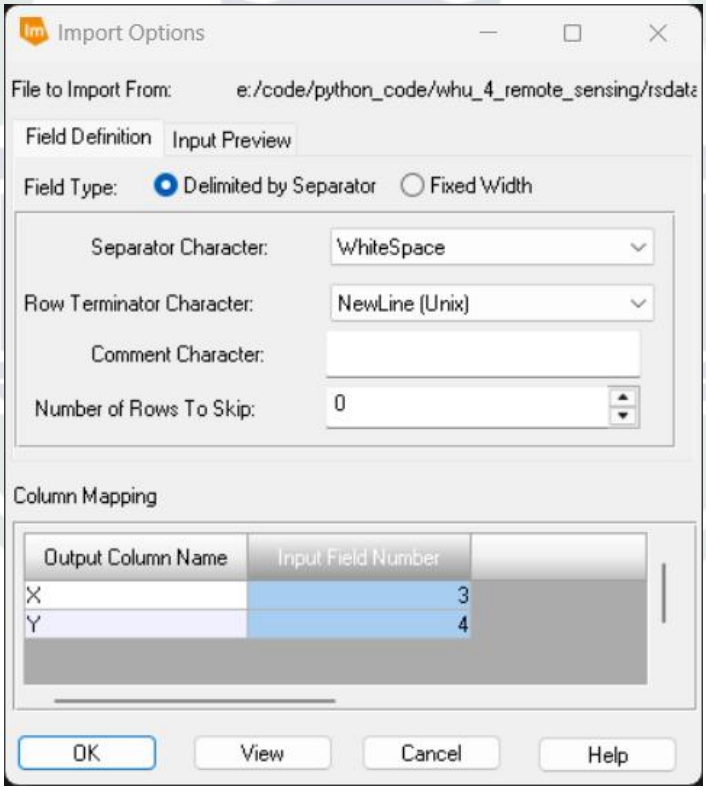
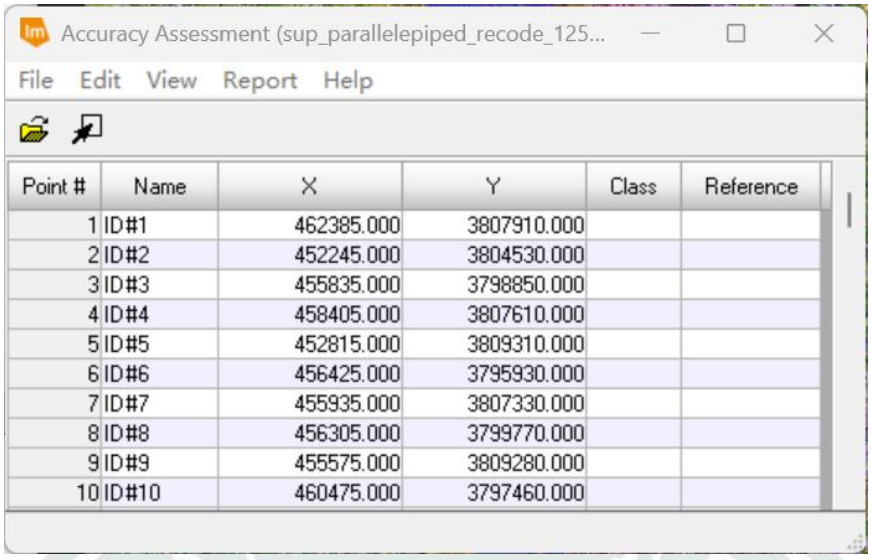


图2.3.10-27 设置导入随机检查点的配置

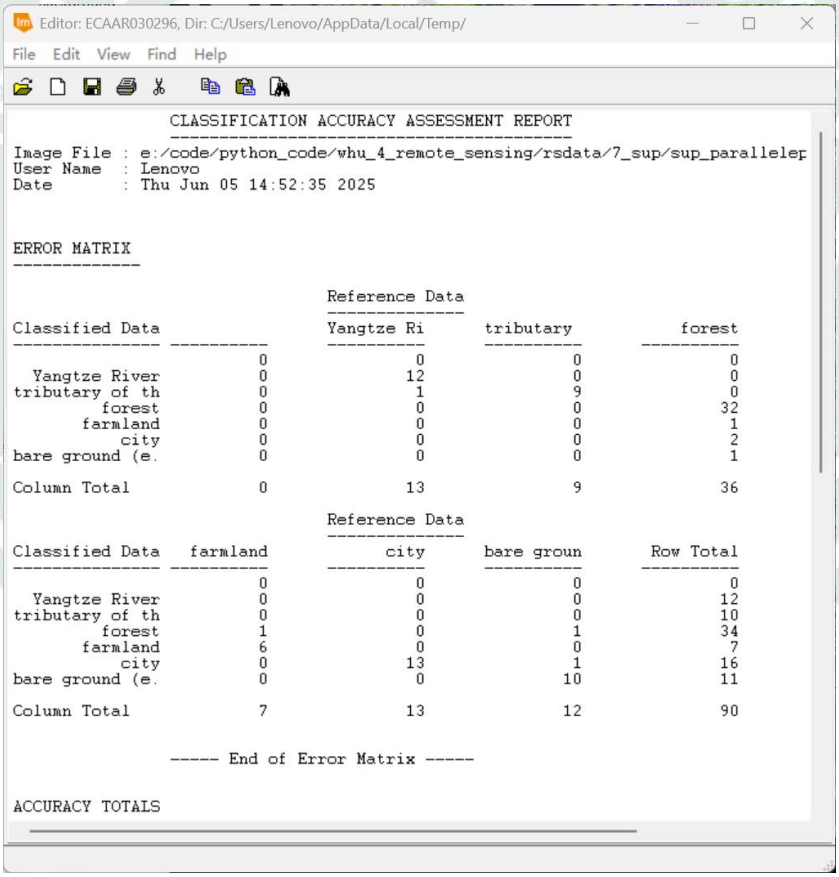
导入成功后，会发现 X、Y 坐标信息已经成功导入（图 2.3.10-28），但是 Reference 信息还需要自己从 Excel 表格中抄过来（图 2.3.10-29）：



Point #	Name	X	Y	Class	Reference
1	ID#1	462385.000	3807910.000		
2	ID#2	452245.000	3804530.000		
3	ID#3	455835.000	3798850.000		
4	ID#4	458405.000	3807610.000		
5	ID#5	452815.000	3809310.000		
6	ID#6	456425.000	3795930.000		
7	ID#7	455935.000	3807330.000		
8	ID#8	456305.000	3799770.000		
9	ID#9	455575.000	3809280.000		
10	ID#10	460475.000	3797460.000		

图2.3.10-28 X、Y坐标信息已经成功导入，但是Reference信息还需要自己从Excel表格中抄过来

然后抄完之后，同样的，先关掉再打开，查看精度评定报告（图 2.3.10-29、2.3.10-30）：




```
                CLASSIFICATION ACCURACY ASSESSMENT REPORT
                -----------------------------------------
Image File  : e:/code/python_code/whu_4_remote_sensing/rsdata/7_sup/sup_parallelep
User Name   : Lenovo
Date        : Thu Jun 05 14:52:35 2025


ERROR MATRIX
------------

                              Reference Data
                              --------------
Classified Data               Yangtze Ri      tributary          forest
---------------- ----------   ----------      ----------      ----------
                           0            0               0               0
  Yangtze River            0           12               0               0
tributary of th            0            1               9               0
         forest            0            0               0              32
       farmland            0            0               0               1
           city            0            0               0               2
bare ground (e.            0            0               0               1

Column Total               0           13               9              36

                              Reference Data
                              --------------
Classified Data    farmland           city       bare groun       Row Total
---------------- ----------   ----------      ----------      ----------
                           0            0               0               0
  Yangtze River            0            0               0              12
tributary of th            0            0               0              10
         forest            1            0               1              34
       farmland            6            0               0               7
           city            0           13               1              16
bare ground (e.            0            0              10              11

Column Total               7           13              12              90


                  ----- End of Error Matrix -----

ACCURACY TOTALS
```

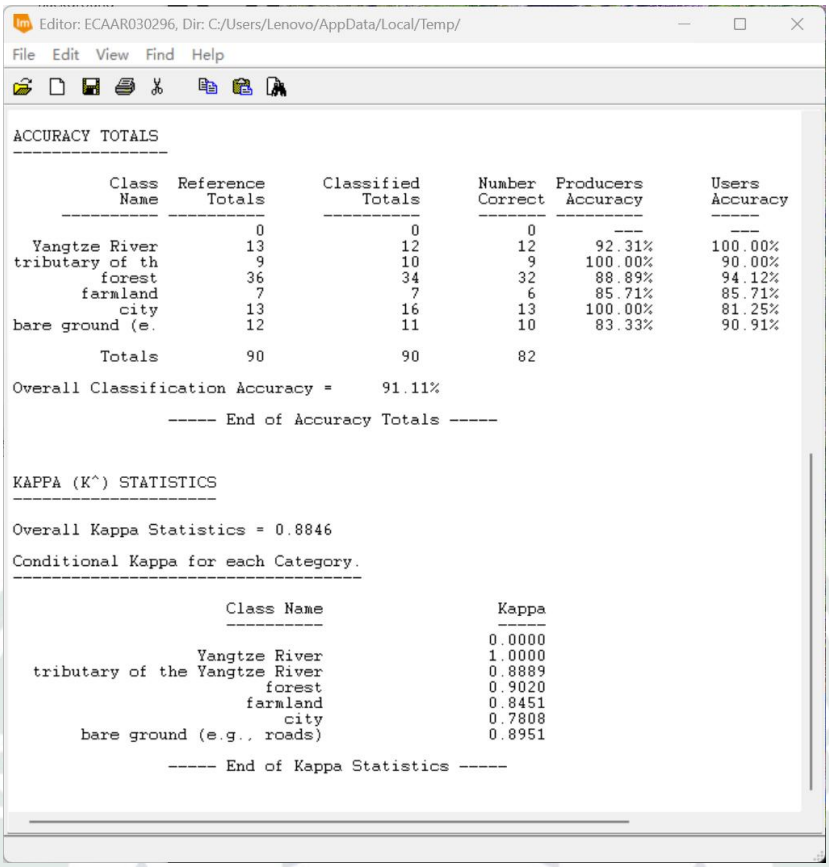
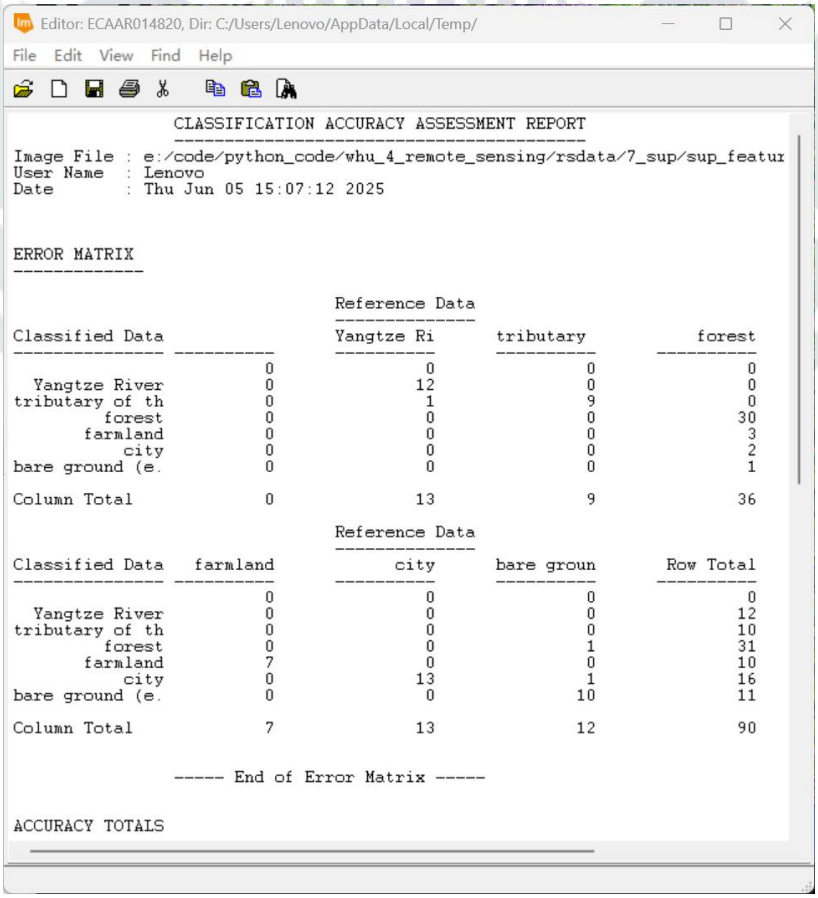


图2.3.10-29 Parallelepiped（平行六面体法）的精度评定报告



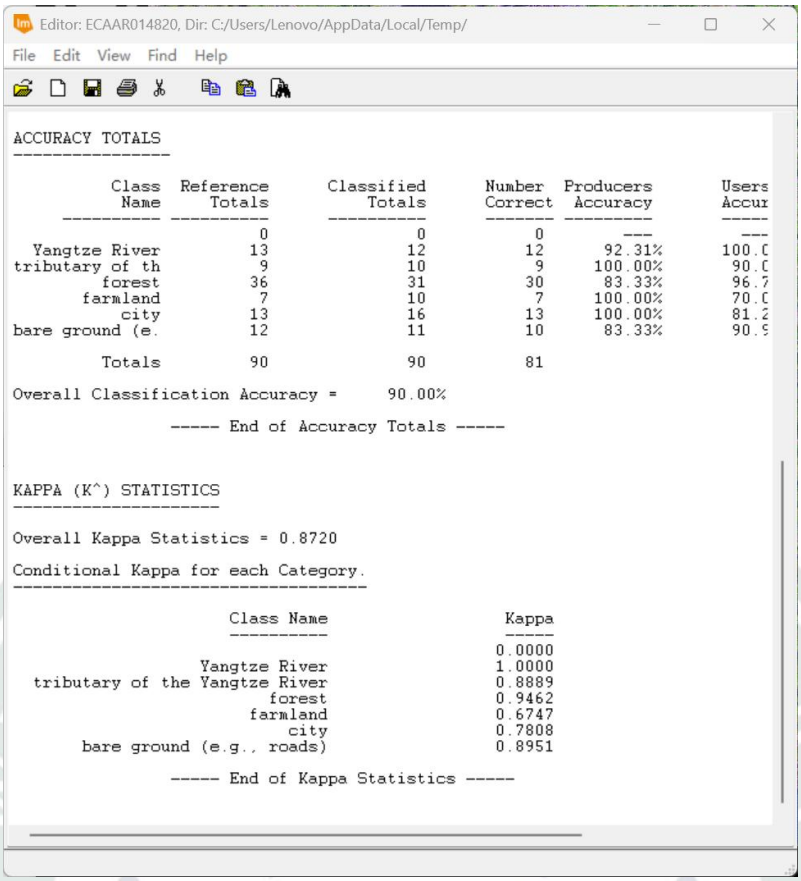


图2.3.10-30 Feature Space（特征空间法）的精度评定报告

综合以上三种监督分类方法的精度，Parallelepiped（平行六面体法）的精度最高（综合精度 91.11%，Kappa 系数 0.8846），后续专题制图将基于该方法的结果 sup_parallelepiped_recode_1259.img 进行。

2.3.11 后处理

首先选择 Raster-Thematic-Clump 工具，打开界面进行如下配置（图 2.3.11-1）：

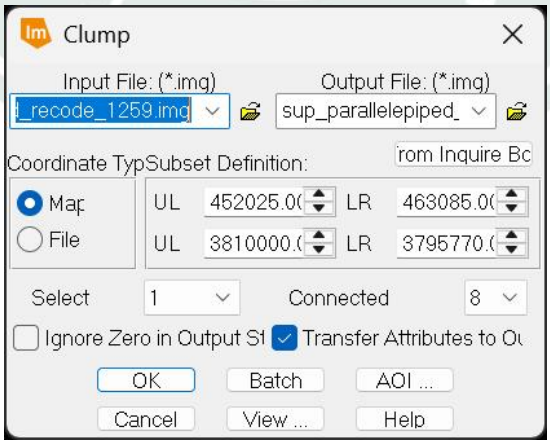


图2.3.11-1 Raster-Thematic-Clump工具的配置

其中，Connected 选项是选择连通域的连通规则是选择 4-邻域还是 8-邻域。

Clump 处理的结果是连通域分析的结果，处理之后在图上看上去没有什么区别，但是连通域分析的结果已经关联到经过 Clump 处理的图上了。

接下来就要让小区消失了。有两种工具可供选择：选择 Raster-Thematic-Sieve 工具会删除小区，但是删除之后并不给它赋予任何类别；选择 Raster-Thematic-Eliminate 工具会删除小区，并且会将删除后的区域赋予周围区域的类别。这里更加推荐使用 Eliminate 工具（图 2.3.11-2）。

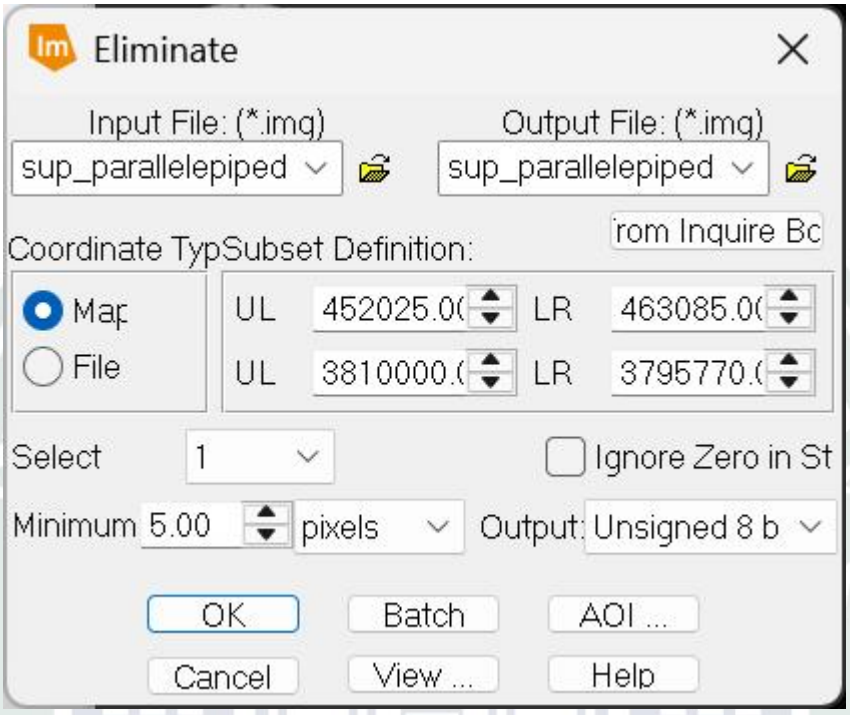


图2.3.11-2 Eliminate工具的配置

处理完成后，导入处理得到的图片，会发现是全黑的（图 2.3.11-3）：

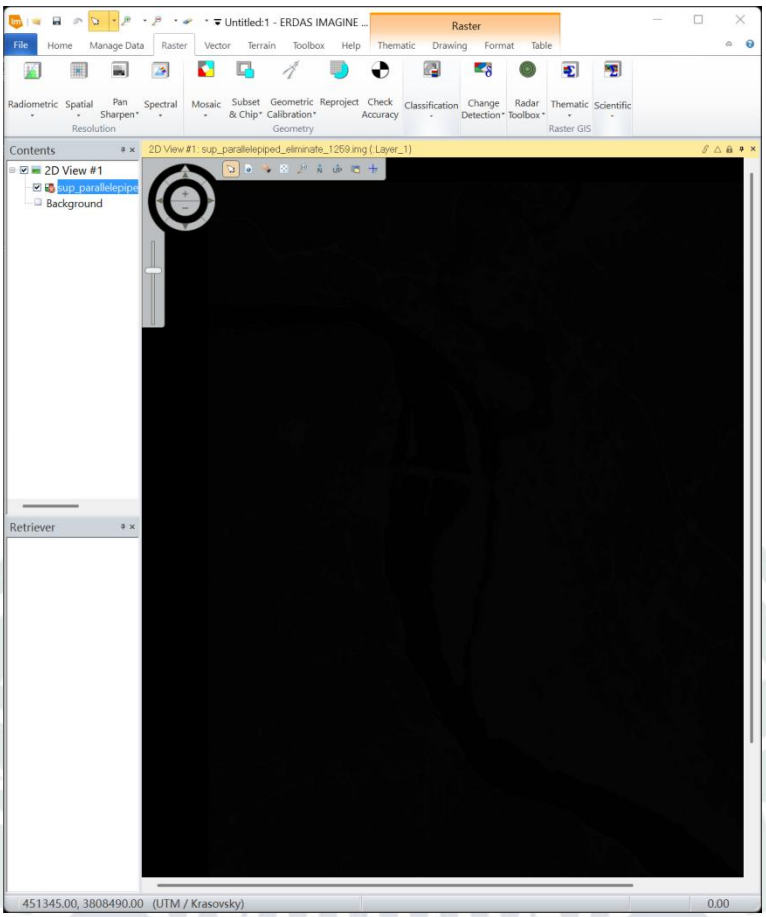


图2.3.11-4 打开刚刚处理完小区的影像，会发现全部是黑的

这个时候不要慌，打开其属性表，会发现这是颜色配置导致的（图 2.3.11-5）：

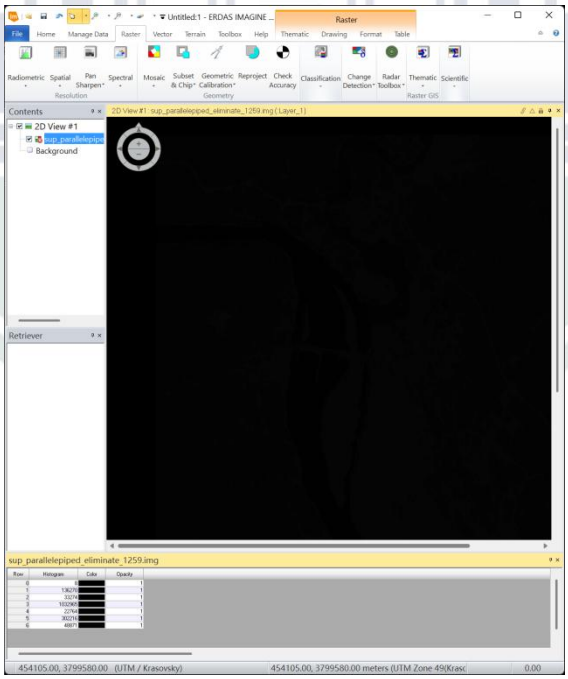


图2.3.11-5 上图中全部是黑的是因为颜色配置导致的

重新配置颜色之后，可以得到正常显示的效果（图 2.3.11-6）：

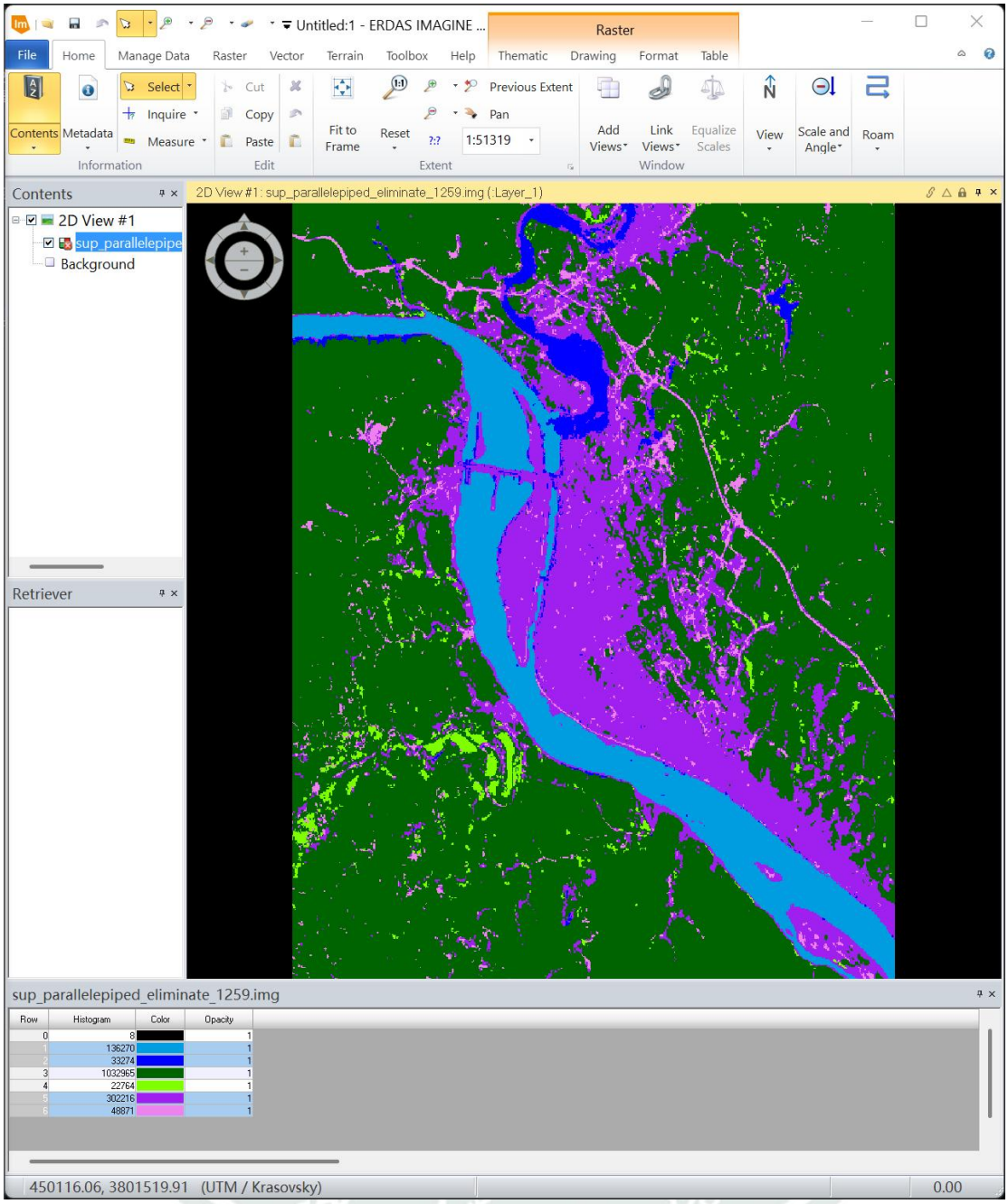


图2. 3. 11-6 正常显示颜色的进行消失小区之后的效果

此时观察数据表，会发现其中缺少类别名称和后面进行专题制图的类别名称信息和面积信息，需要手动添加分类名称信息。打开上方的 Raster-Table，可以找到添加分类名称的功能 Add Class Name 和添加面积信息的功能 Add Class Area，这两个在 2.3.12 专题制图的时候都会用到，所以都添加一下（图 2.3.11-7）。

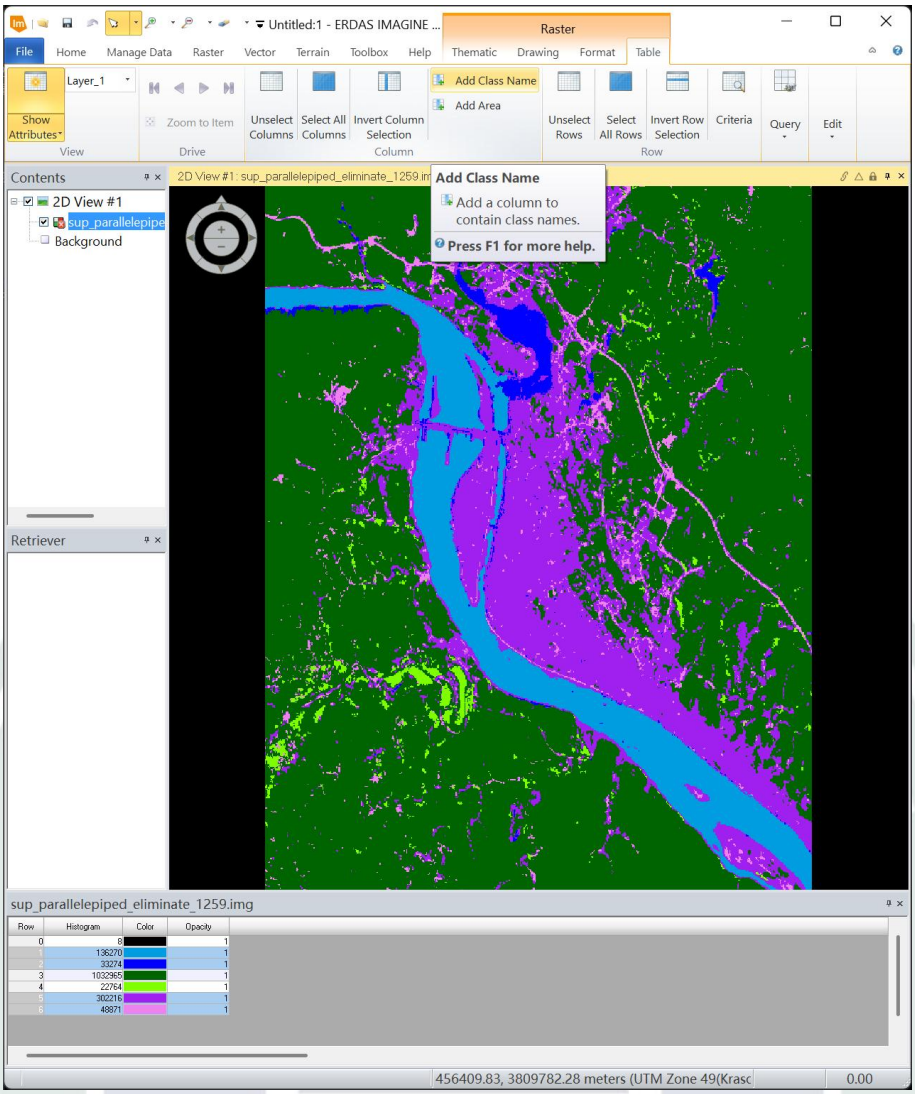


图2.3.11-7 添加分类名称的功能Add Class Name和添加面积信息的功能Add Class Area

添加面积信息的时候，选择平方千米作为面积单位（图 2.3.11-8）：

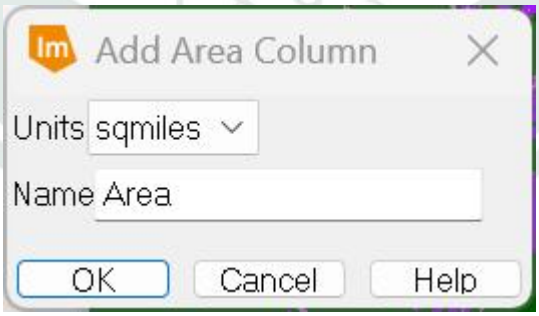


图2.3.11-8 添加面积信息Add Class Area选择平方千米作为面积单位

这里添加类型名称的时候，考虑到制图图例的显示，使用更加规范的命名（图 2.3.11-9）：1-Yangtze River，2-tributary of the Yangtze River，3-forest，4-farmland，5-city，6-bare ground (e.g., roads)。

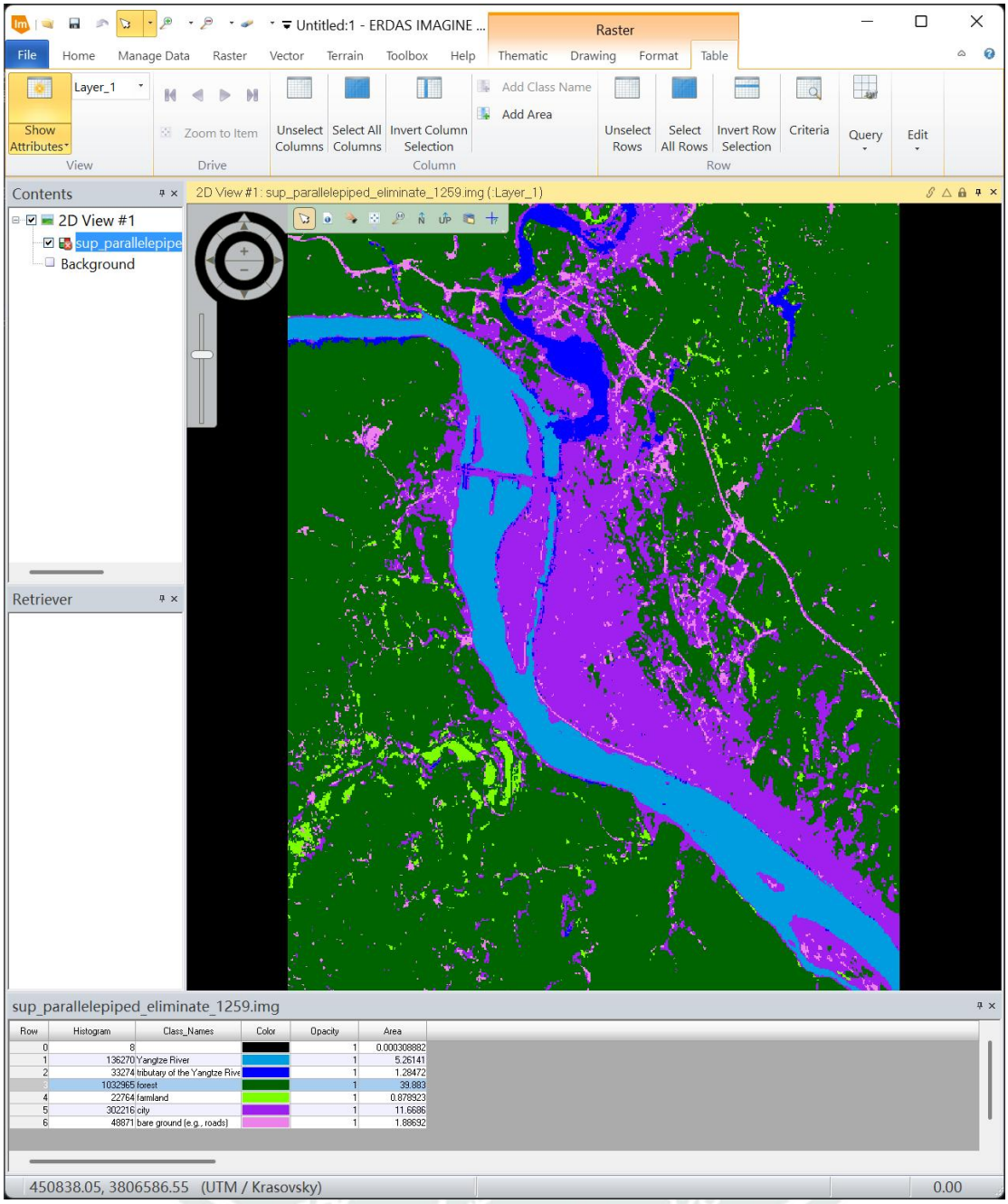


图2. 3. 11-9 添加类型名称时使用更加规范的命名

2.3.12 专题制图

首先点击上方的 Home-Add Views-Create New Map View（图 2.3.12-1）：

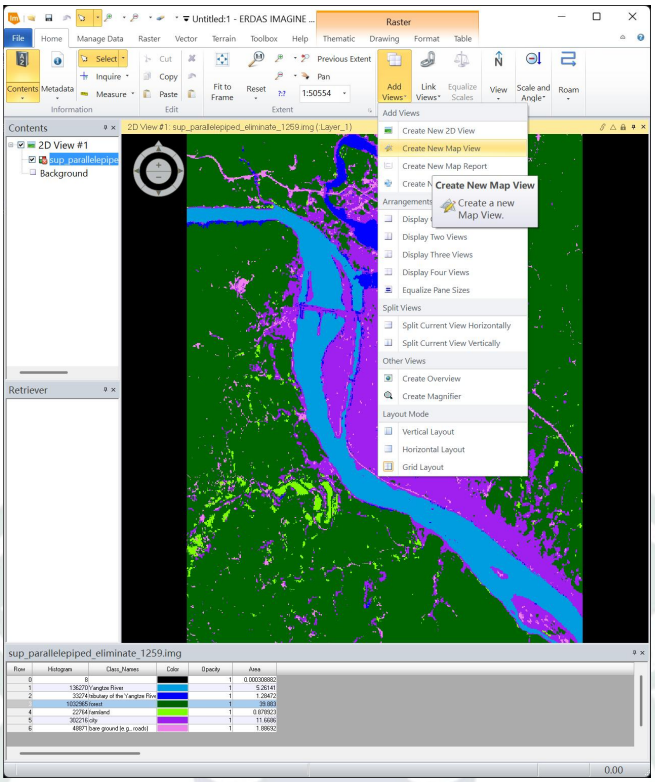


图2.3.12-1 点击上方的Home-Add Views-Create New Map View

这样可以得到一个地图布局（图 2.3.12-2）：

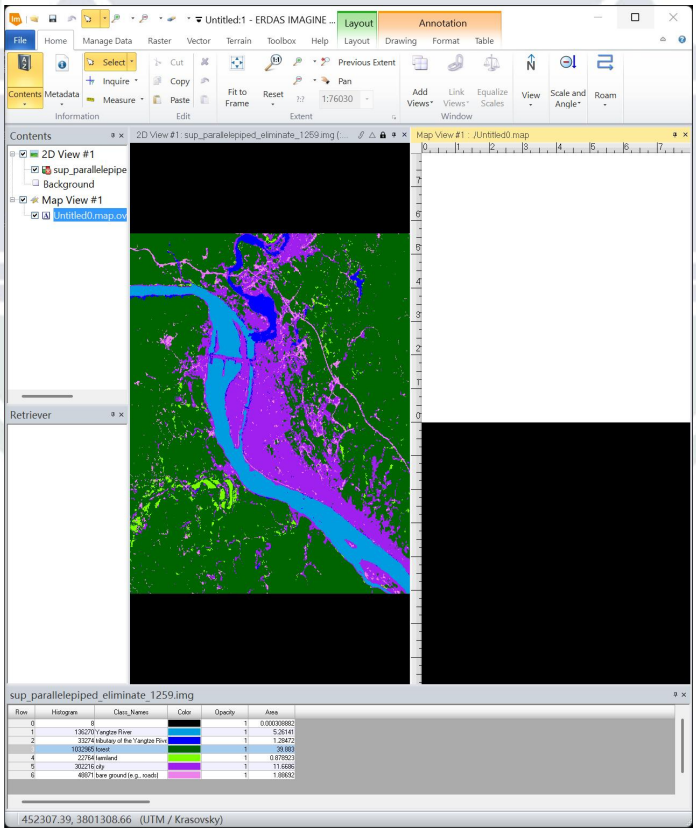


图2.3.12-2 地图布局

如果需要的话，可以在 Layout-Page Size 中设置页面大小规格（图 2.3.12-3）。但是

经过作者试验，其实就用默认的正方形是最合适的。

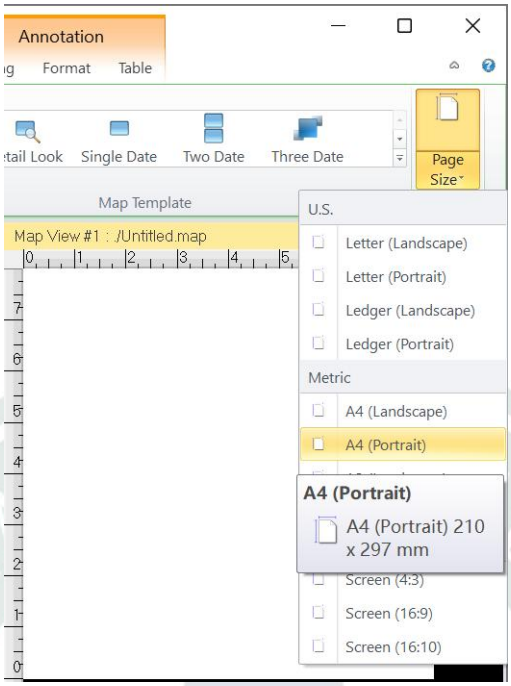


图2.3.12-3 可以在Layout-Page Size中设置页面大小规格

点击上方的 Layout-Map Frame，然后先在右边的布局中长按拖出一个地图框，然后在弹出的窗口中选择 Viewer（图 2.3.12-4）：

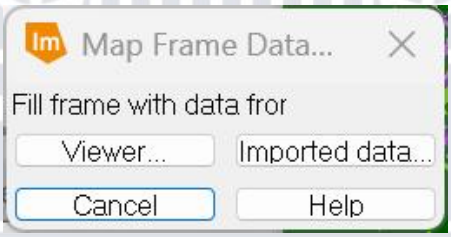


图2.3.12-4 拖出一个地图框后在弹出的窗口中选择Viewer

然后就会弹出图 2.3.12-5 这样的一个窗口：

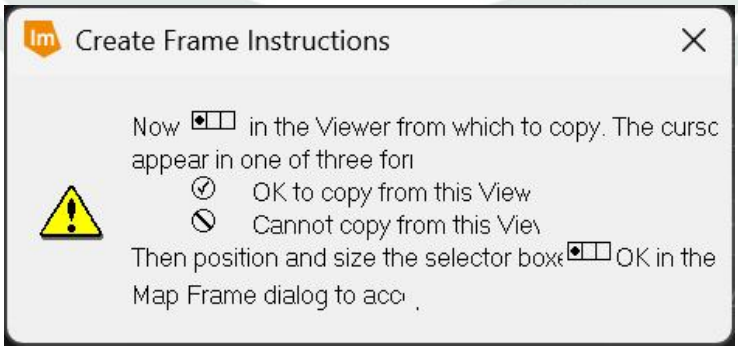


图2.3.12-5 提示需要选择一个视图的窗口

然后点击一下左边的地图视图，这个时候弹出一个窗口要求选择视图框（图 2.3.12-6），这个时候点击一下 Use Entire Coordinates，即可选上整个视图图幅。

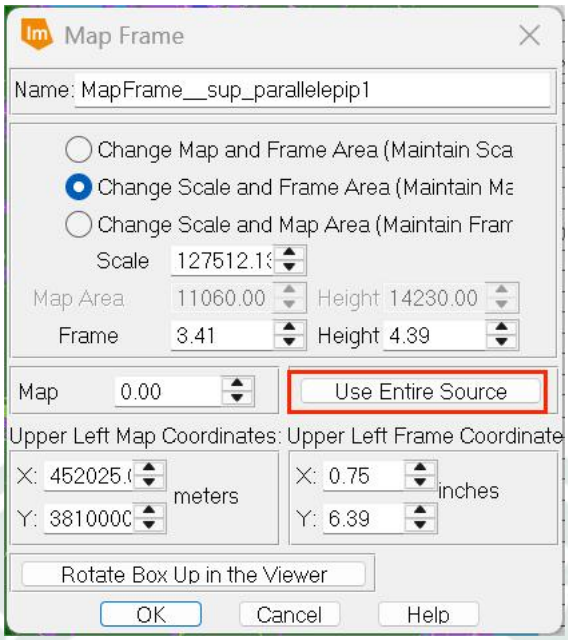


图2.3.12-7 选择视图框（点击一下Use Entire Coordinates即可选上整个视图图幅）

点击 OK，即可在布局上呈现出地图框（图 2.3.12-8）：

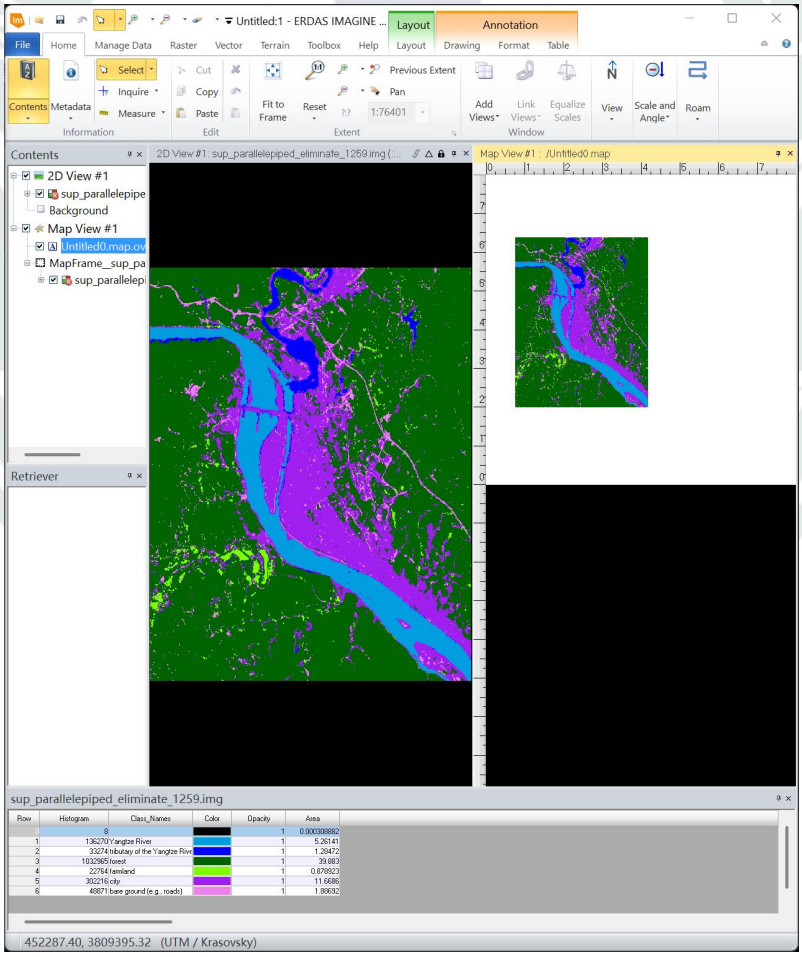


图2.3.12-8 布局上呈现出地图框

点击 Layout-Map Grid，然后点击一下布局汇总的视图框，弹窗中的 Horizontal Axis

和 Vertical Axis（可以点击 Copy to Vertical）如图 2.3.12-9 设置：

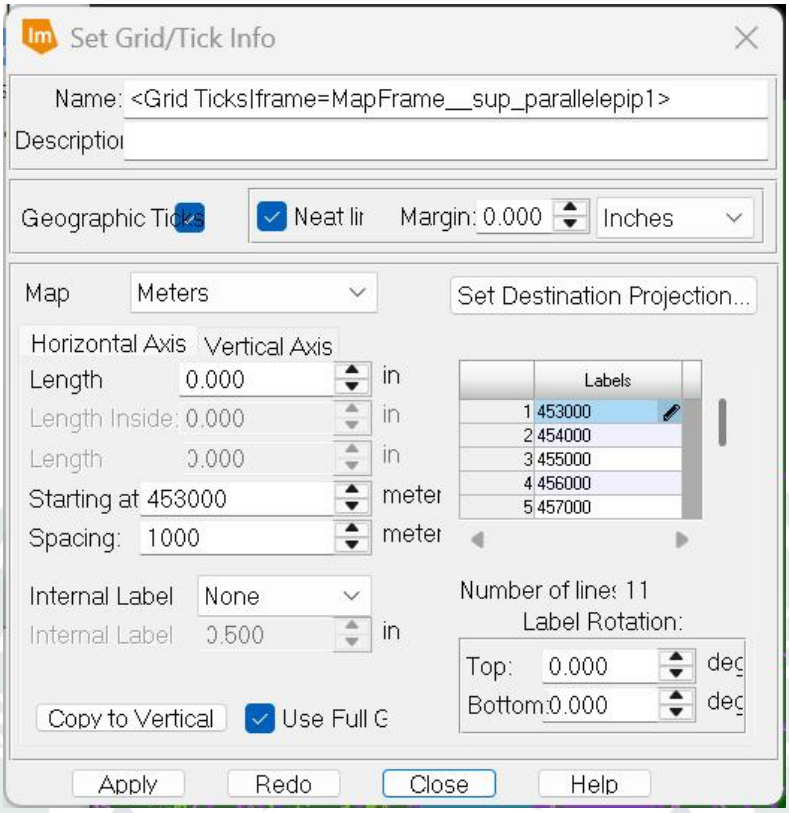


图2.3.12-9 设置Horizontal Axis和Vertical Axis的属性

这样即可得到地图网格（图 2.3.1.2-10）：

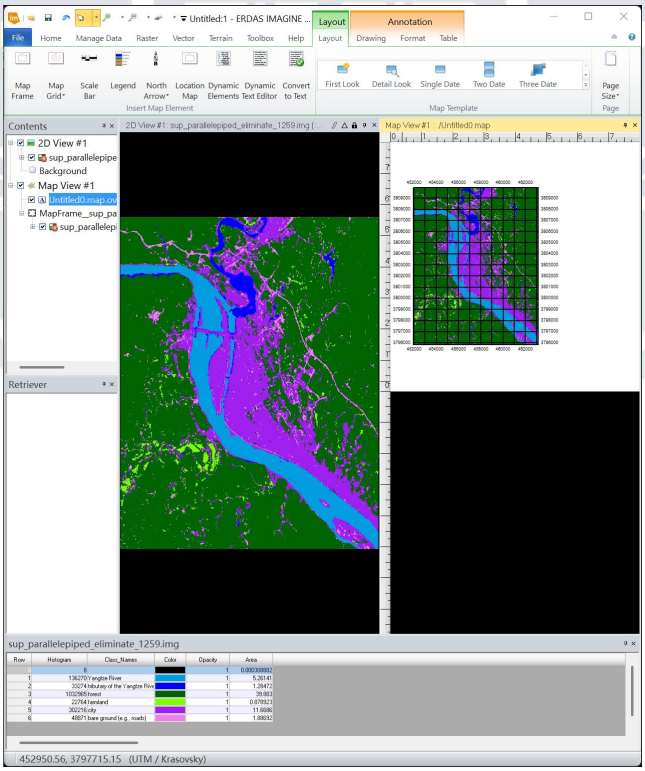


图2.3.12-10 得到地图网格

然后点击上方的 Layout-Scale Bar，首先在地图框的下方拖出一个区域来显示比例尺，拖动完成后会弹出在图 2.3.12-11 所示的提示，此时在右边布局的地图框（Mapframe）中点击一下，弹出图 2.3.12-12 所示配置窗口。

图2. 3. 12-11 提示选择地图框

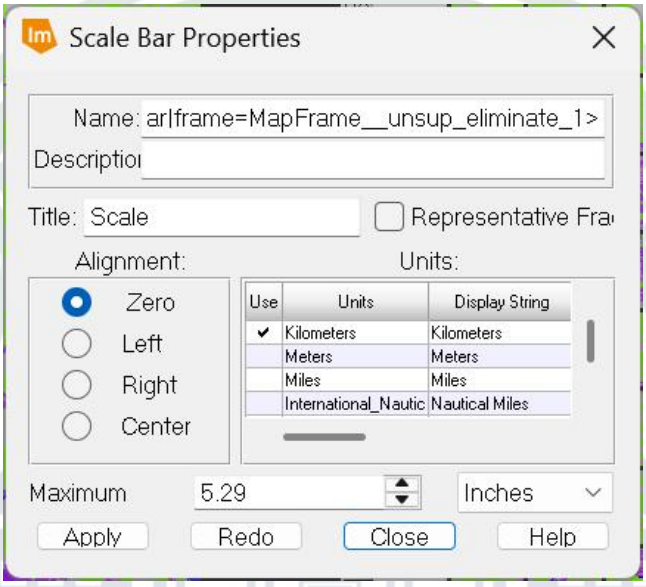


图3. 2. 12-12 比例尺的配置窗口

点击 OK，可以得到比例尺（图 2.3.12-13）：

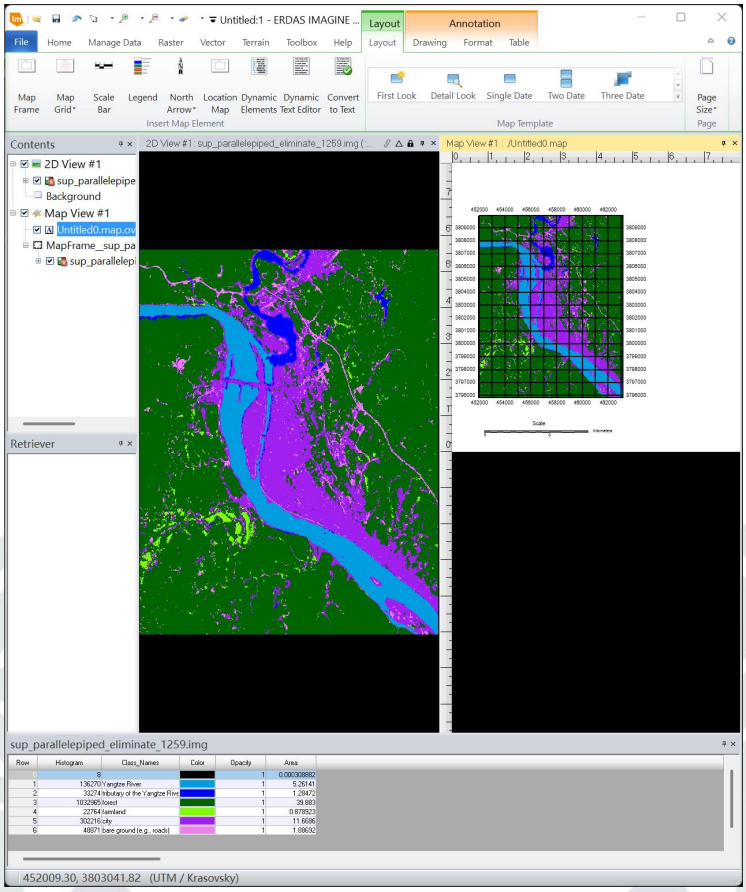


图2. 3. 12-13 得到比例尺

然后点击 Layout-Legent，先在右边的布局中拖动出一个区域来显示图例，然后点击一下右边的地图框，此时会打开配置窗口，选中有实际意义的类别，并在右边的 Add Discriptors 中点击添加 Area 这一栏（图 2.3.12-14）。

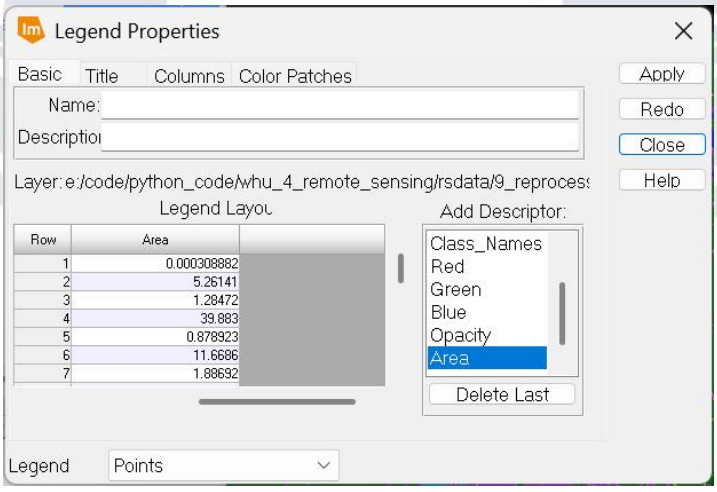


图2. 3. 12-14 图例的配置窗口

点击 OK，可以生成图例（图 2.3.12-15）：

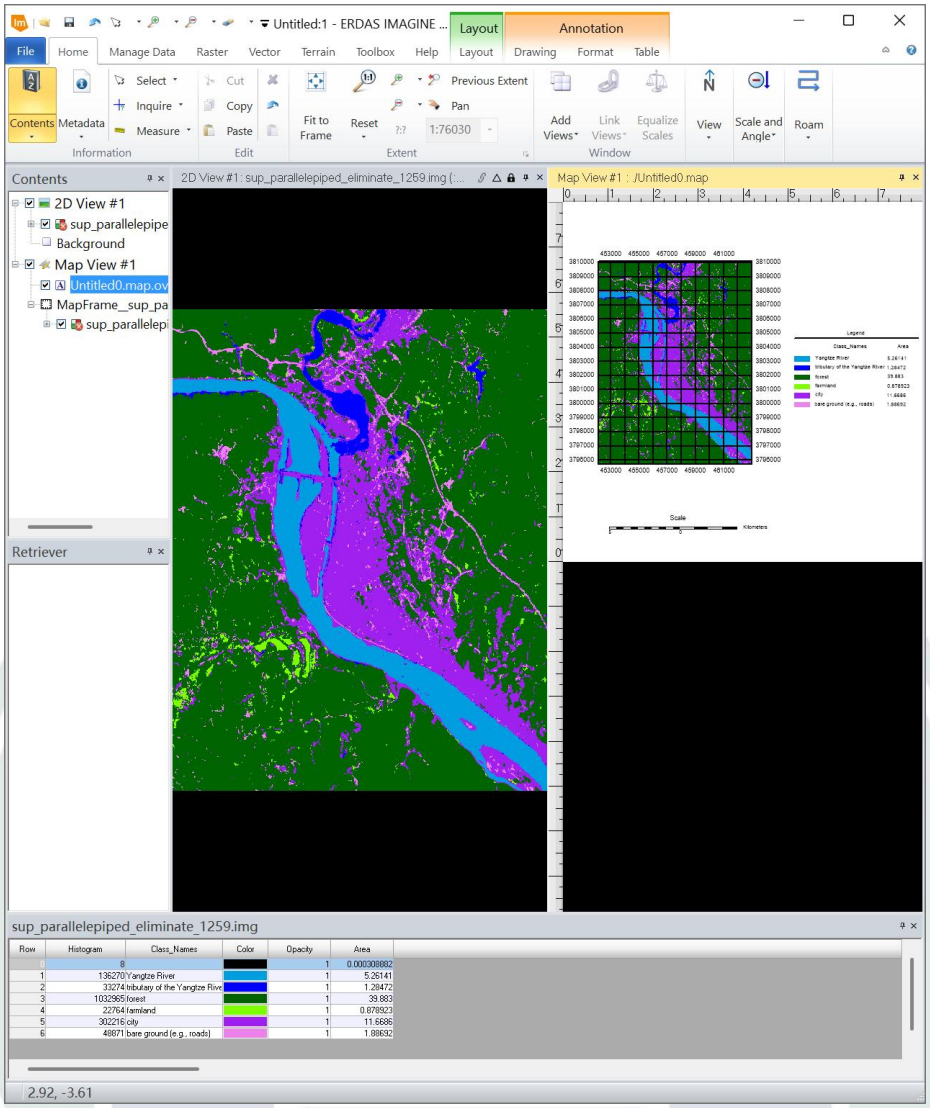


图2. 3. 12-15 生成的图例

然后点击 Layout-North Arrow-Default North Arrow Style，打开指北针样式的设置窗口（图 3.2.12-16）：

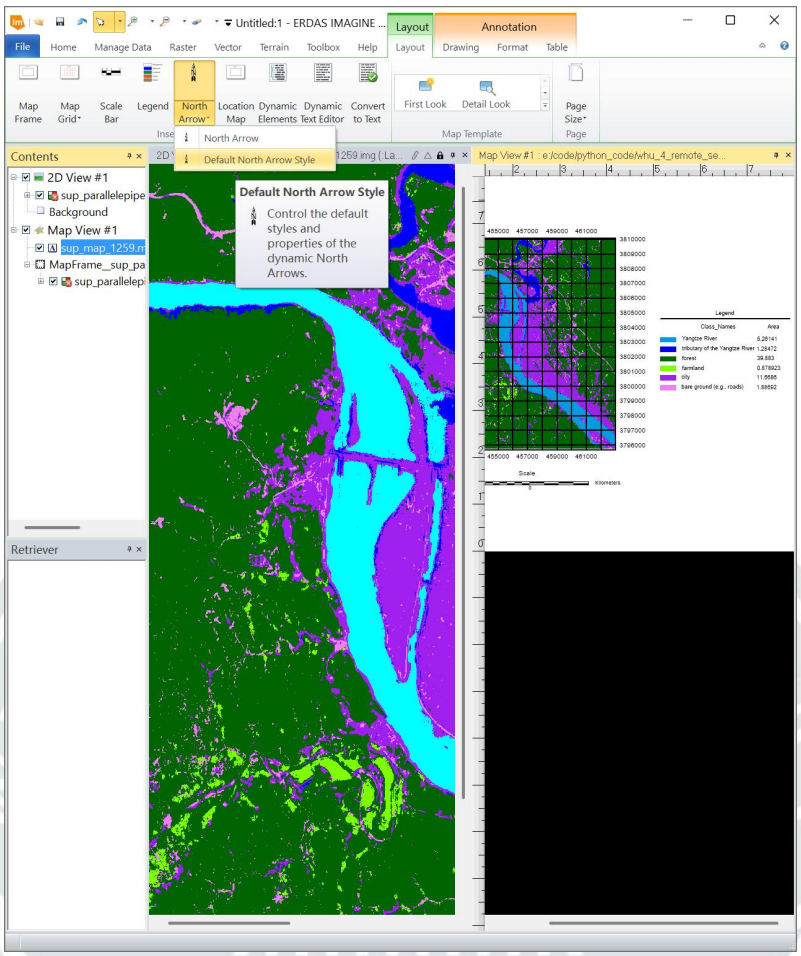
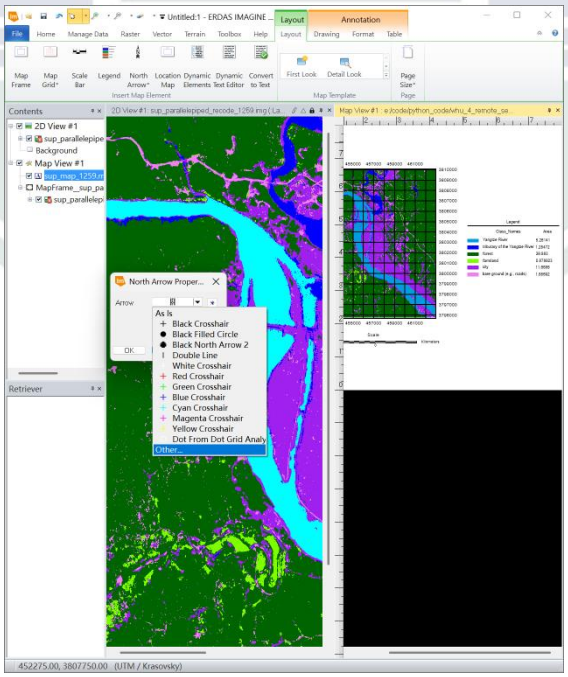


图2. 3. 12-16 指北针样式的设置窗口

接下来的设置如图 2.3.12-17 所示，在下拉菜单中点击 Others，设置 Size 为 40.00，在左侧选择样式为 North Arrows，然后再向下翻到 North Arrows 23。



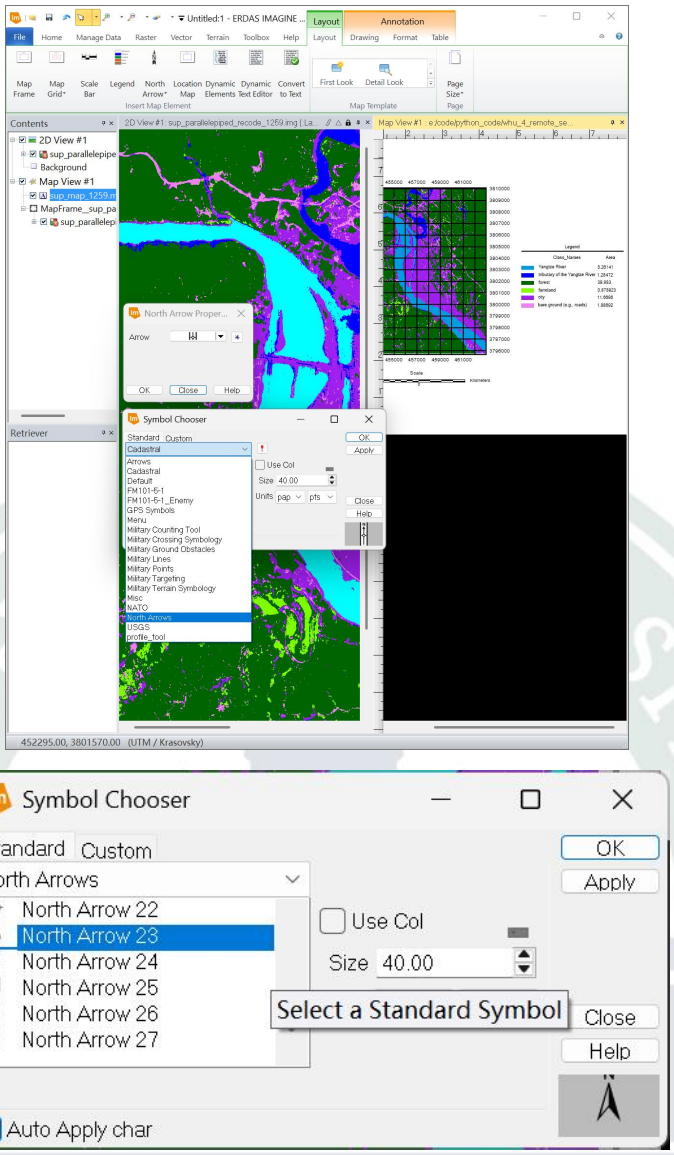


图2. 3. 12-17 指北针样式设置的步骤

点击两次 OK 关掉这两个设置指北针样式的窗口，然后点击 Layout-North Arrow-Default North Arrow，在图上拖出一个框，即可在地图上显示出指北针（图 2.3.12-18）：

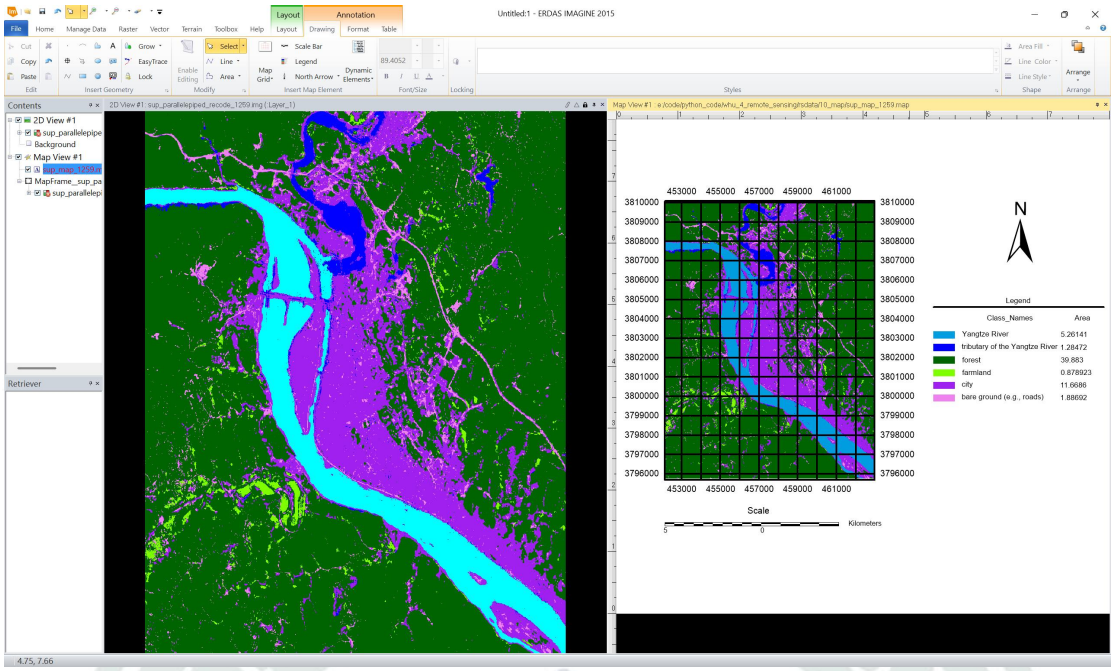


图2.3.12-18 得到指北针

然后点击上方的 Annotation-Drawing-Insert Geometry-“A”，可以在地图上设置文字信息，如地图标题、制图人信息等（图 2.3.12-19）：

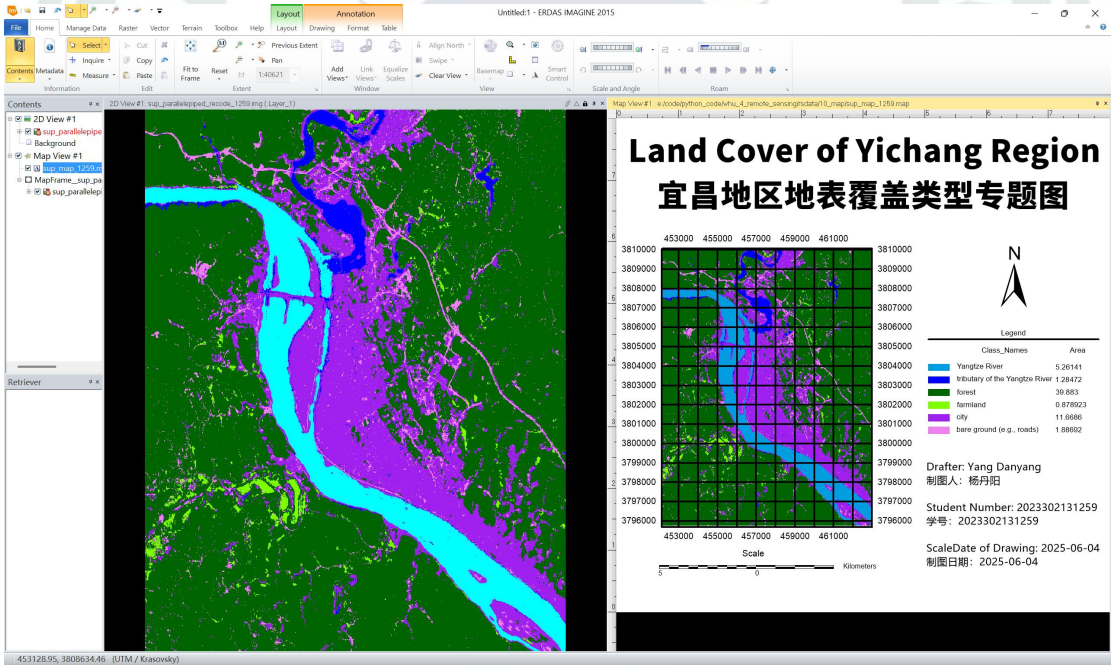


图2.3.12-19 给地图添加文字信息

然后可以在上方的 File-Save As 保存绘制的地图布局文件（.map），这样下次打开.map 到 Map View 的时候就可以继续绘制或修改地图（图 2.3.12-20）：

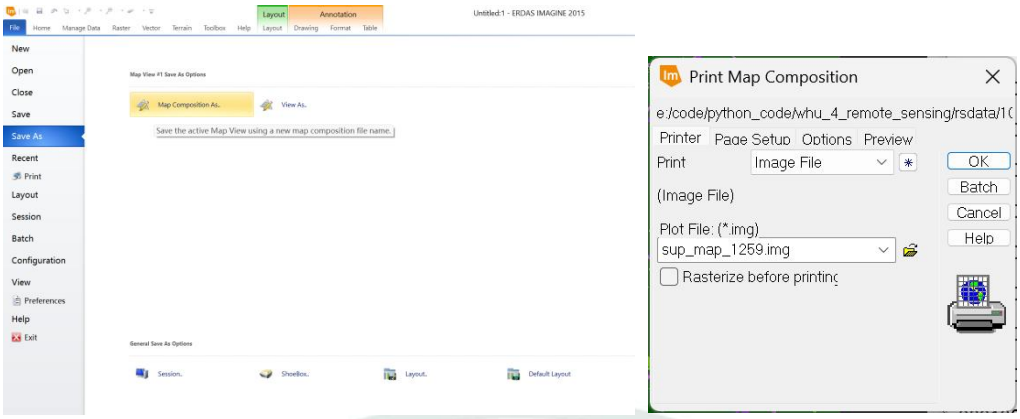


图2.3.12-20 保存绘制的地图布局文件（.map）

可以使用 File-Print 来将绘制完成的地图导出为 PDF（图 2.3.12-21）：

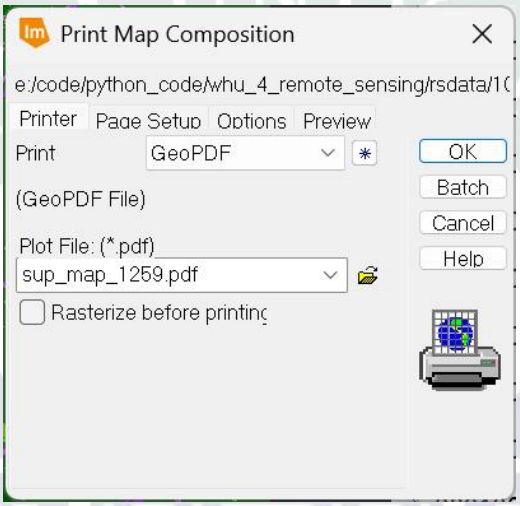


图2.3.12-21 导出绘制的地图为PDF

最终作者绘制完成的宜昌地区地表覆盖类型专题图如图 2.3.12-22 所示：

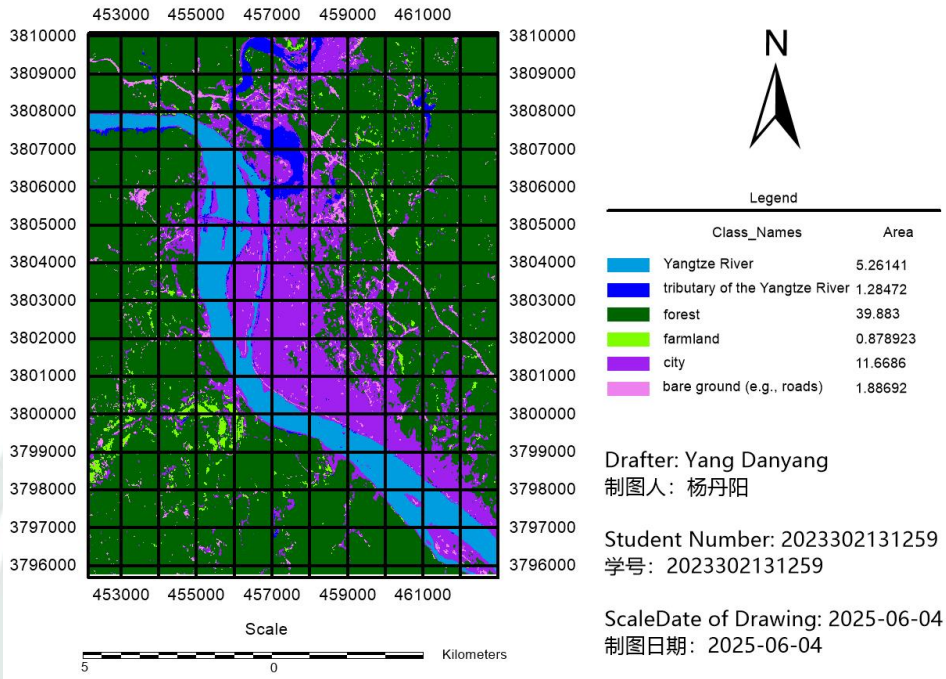


图2.3.12-22 宜昌地区地表覆盖类型专题图

2.4 实习结果与分析

2.4.1 应用 ERDAS 软件的几何纠正误差图

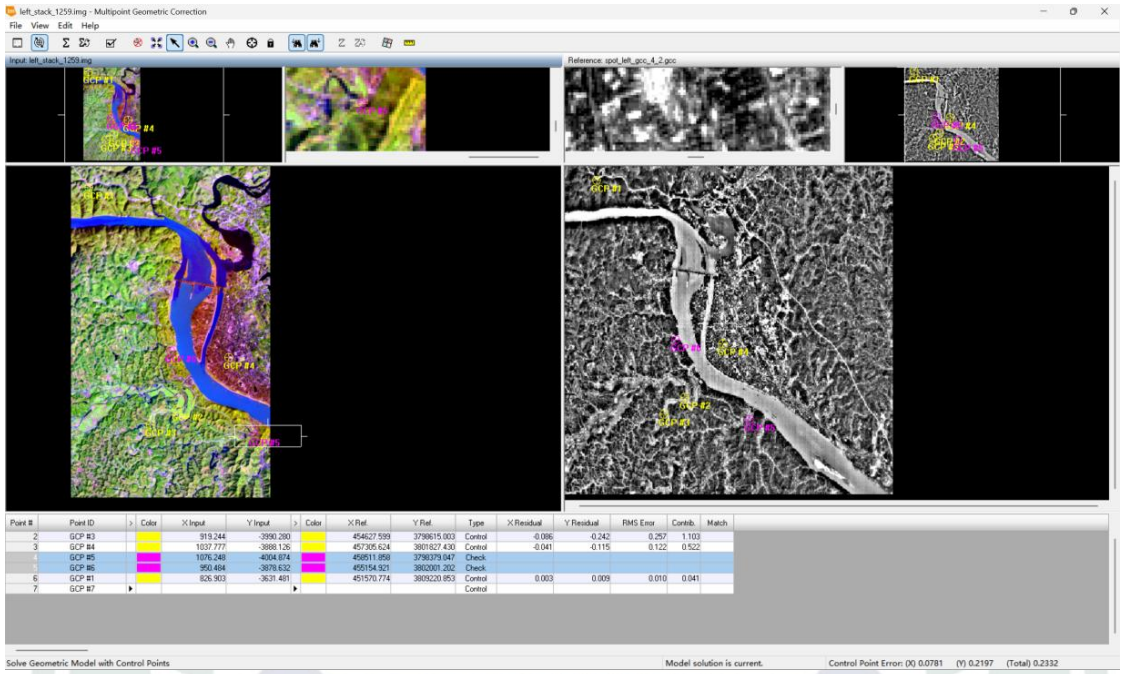


图2.4.1-1 对left_stack_1259.img进行几何精度校正的控制点精度（0.2332<1）

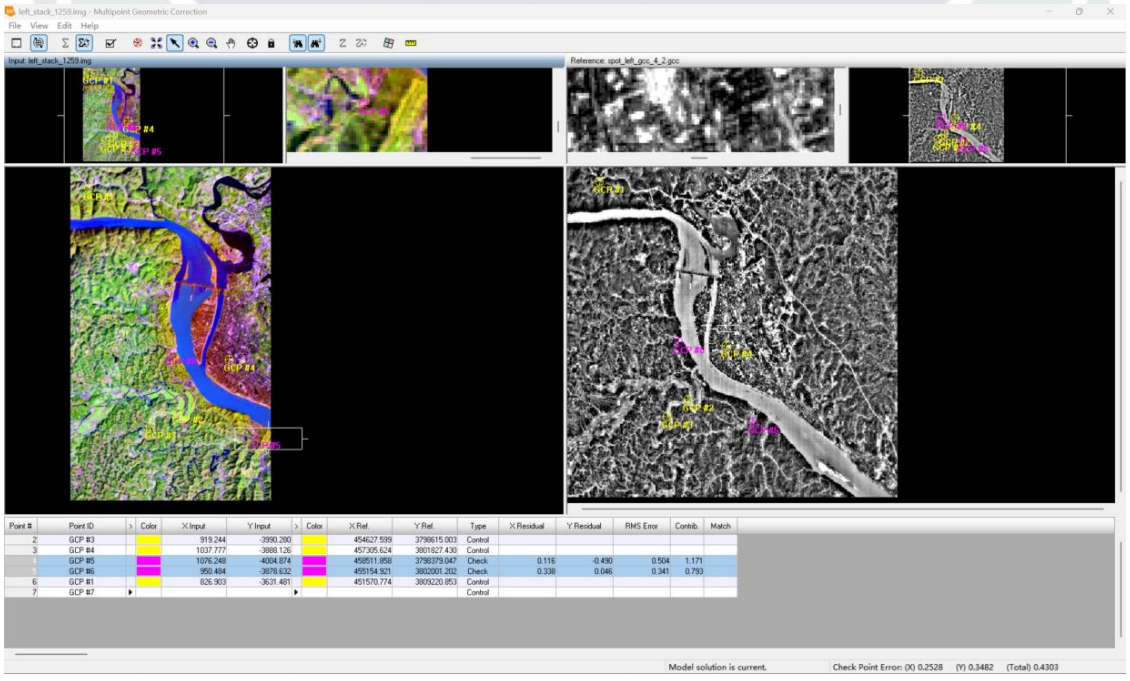


图2.4.1-2 对left_stack_1259.img进行几何精度校正的检查点精度（0.4303<1）

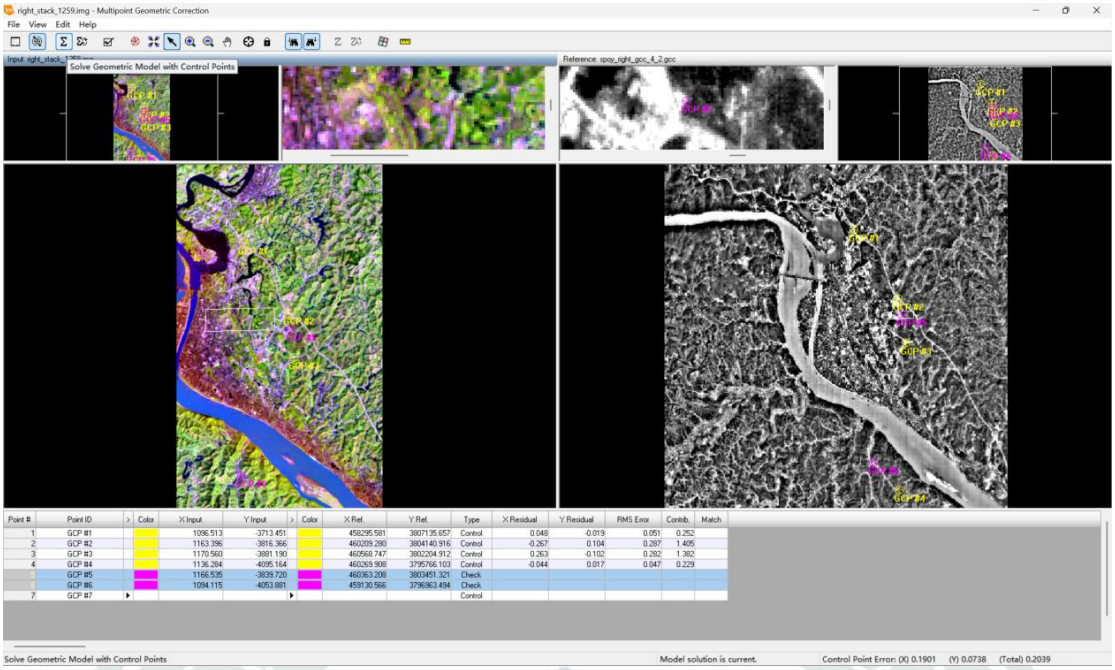


图2.4.1-3 对right_stack_1259.img进行几何精度校正的控制点精度（0.2039<1）

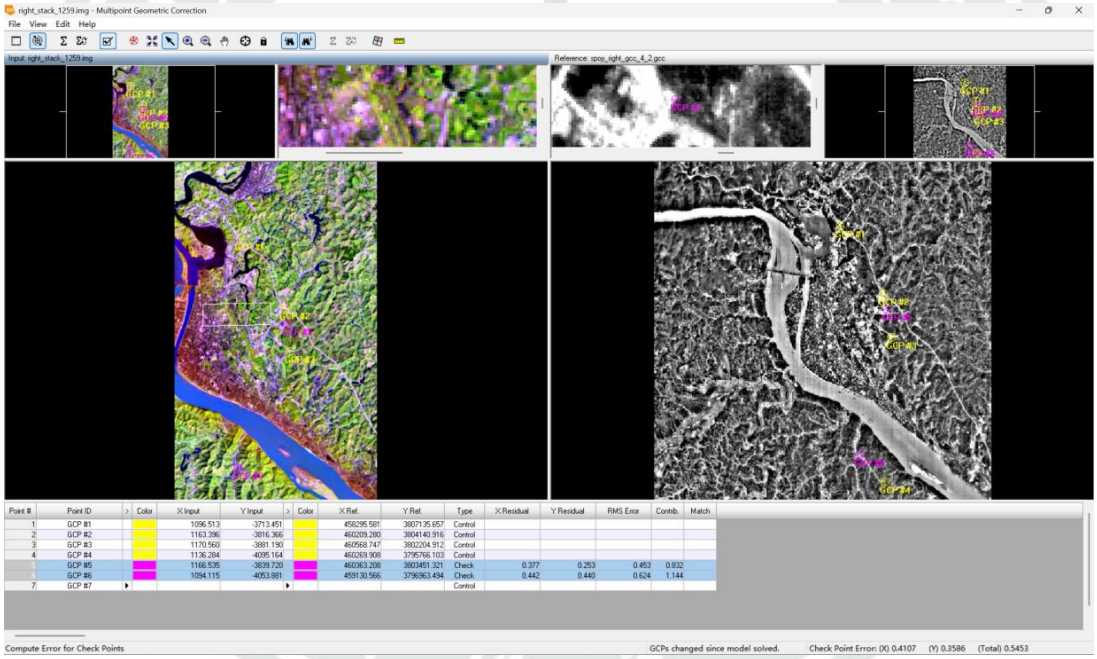


图2.4.1-4 对right_stack_1259.img进行几何精度校正的检查点精度（0.5453<1）

结果分析：在对 left_stack_1259.img 进行几何精度校正时，控制点精度为 0.2332，检查点精度为 0.4303；对 right_stack_1259.img 进行几何精度校正时，控制点精度为 0.2039，检查点精度为 0.5453。这些点位误差（同时包含 X 和 Y 坐标的误差）均小于 1，说明几何纠正效果较好，能够满足本次实习的技术指标要求，所得的几何校正模型在一定程度上能够准确地反映影像的几何变形情况，为后续的影像处理提供了可靠的基础。

从图 2.4.1-1~2.4.1-4 上控制点和检查点的分布来看，我所选取的控制点和检查点都尽可能在全图范围均匀分布，选取在待校正影像和目标影像上都易于辨识的点，以保证在整幅影像中的几何校正精度。

通过几何纠正，我将 left_stack_1259.img 和 right_stack_1259.img 的几何位置成功转换到 sp_yc.tif 的几何位置，实现了影像之间的几何配准。纠正后的影像在校正效果上没有出现明显的错位，看起来是一个整体的平行四边形，这说明几何纠正操作达到了预期目标，为后续的影像镶嵌、融合等操作提供了准确的几何基础。

2.4.2 应用 ERDAS 软件的 ERDAS 影像镶嵌图

应用 ERDAS 软件的 ERDAS 影像镶嵌图的**最佳结果为图 2.4.2-1 所示的使用 Weighted Seamline 镶嵌边，并使用平滑（Smoothing）和羽化（Feathering）的方法，被用于后续处理步骤**，其余结果仅作为对照参考。

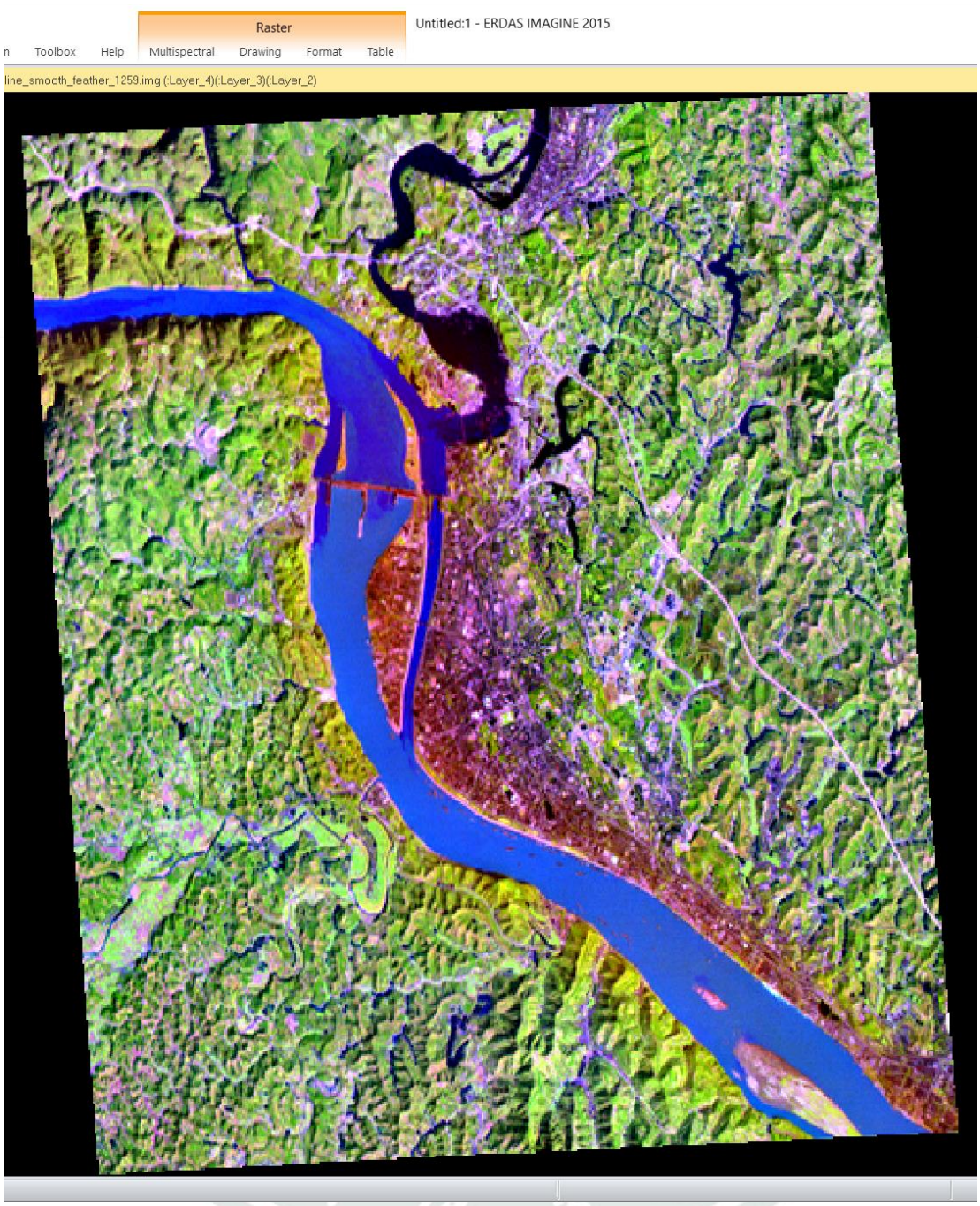


图2.4.2-1 使用Weighted Seamline镶嵌边，并使用平滑（Smoothing）和羽化（Feathering）**【最佳结果】**

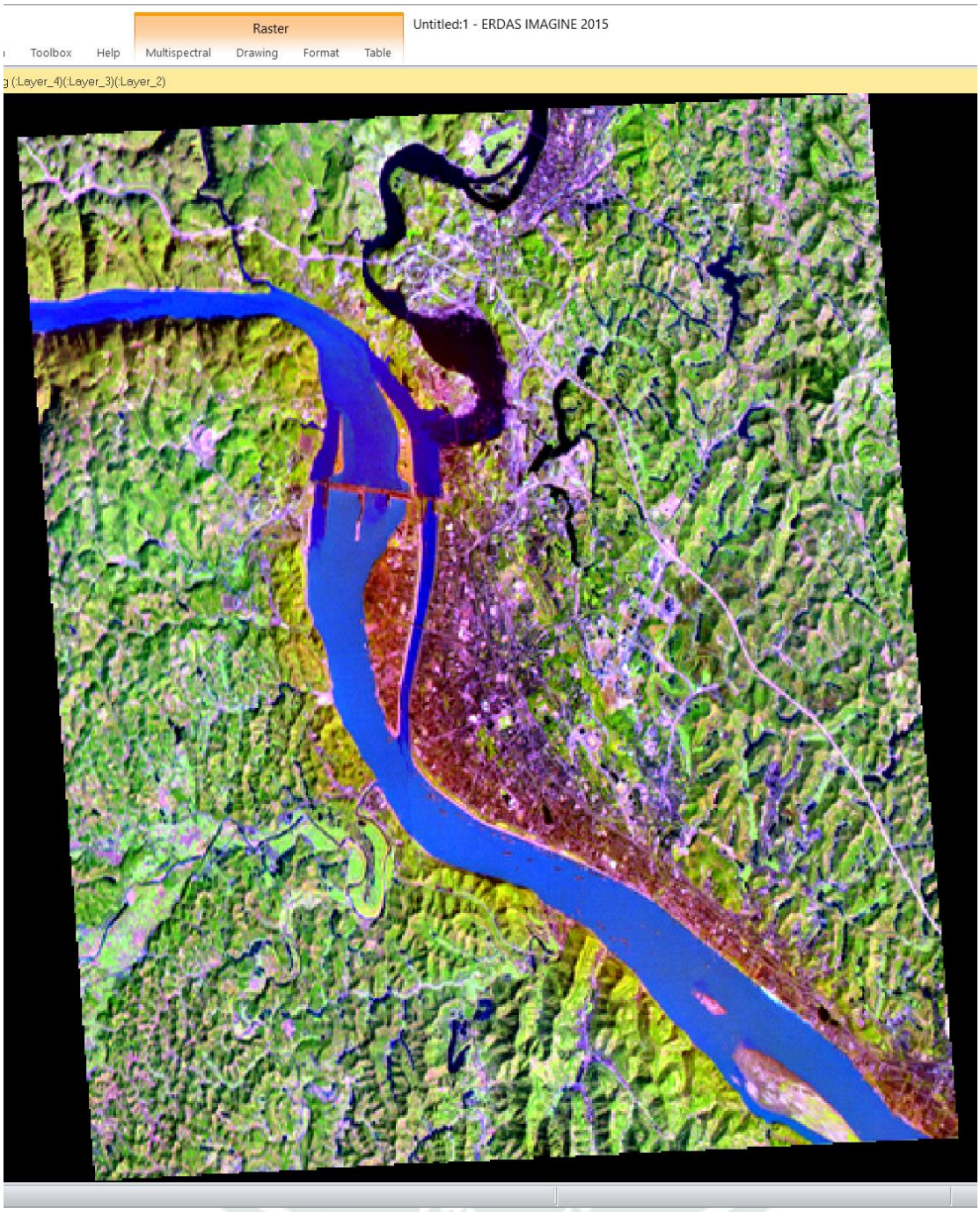


图2.4.2-2 未使用镶嵌边，使用覆盖（Overlay）（可以看出拼接痕迹）

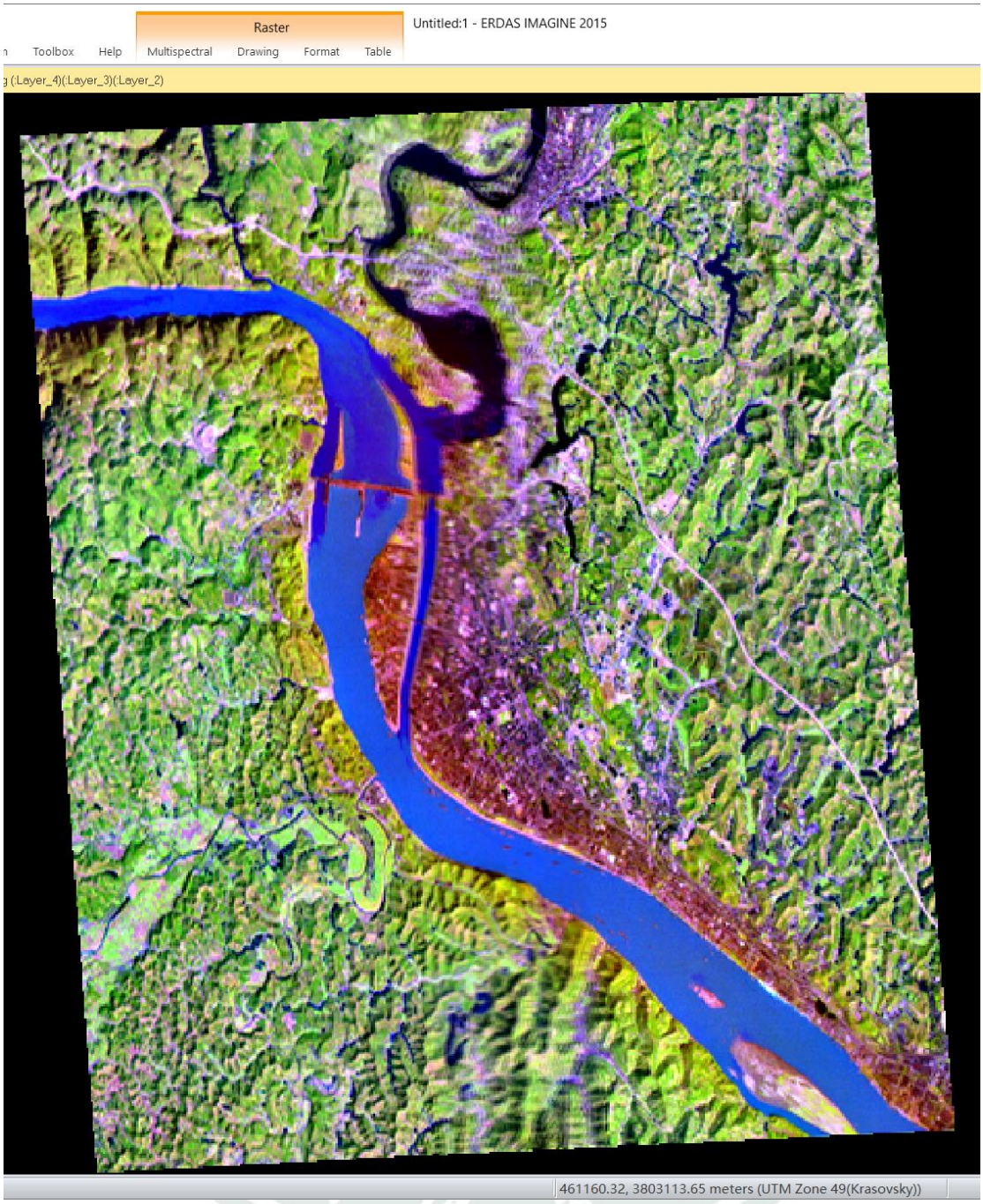


图2.4.2-3 未使用镶嵌边，使用羽化（Feather）（有明显模糊）

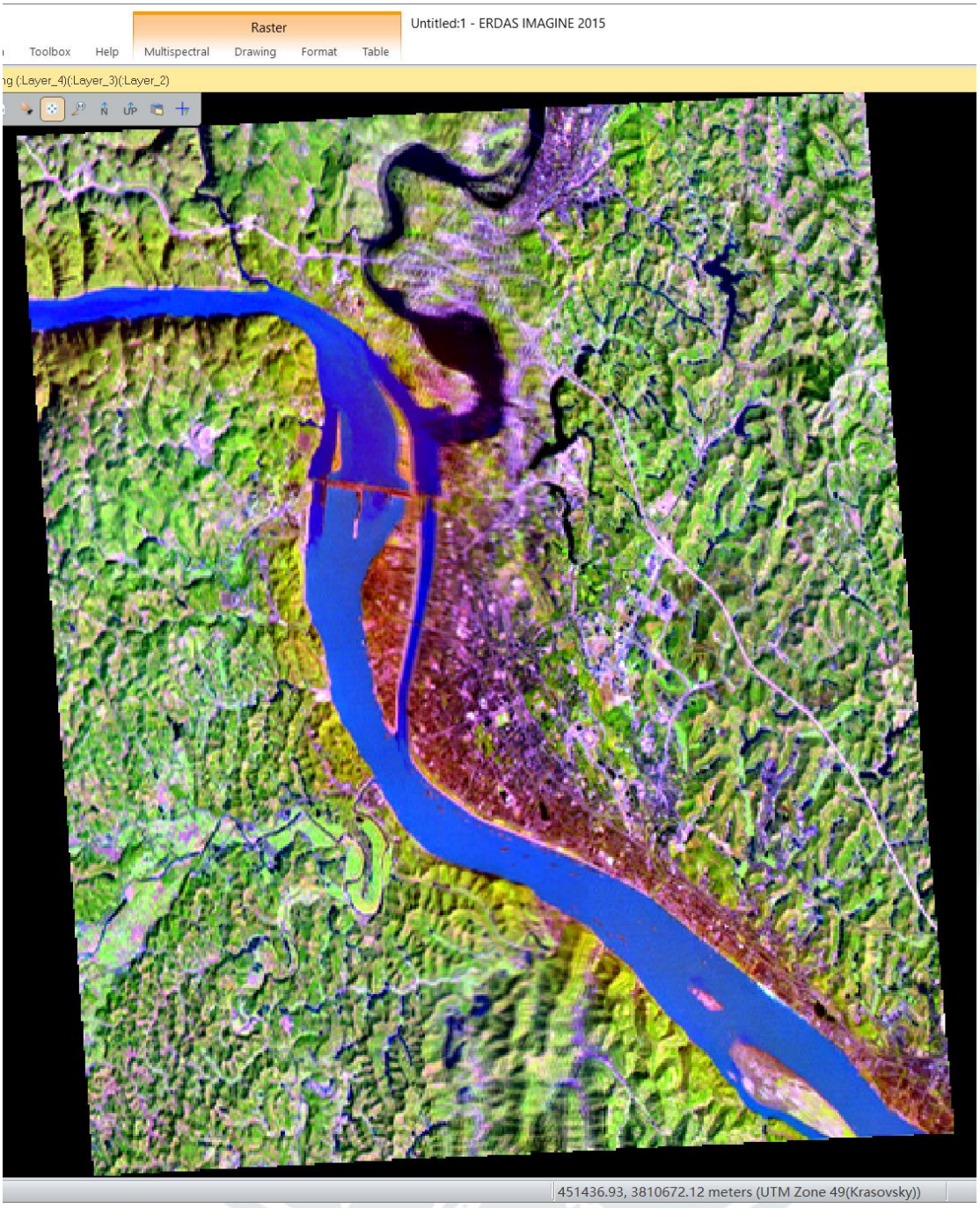


图 2.4.2-4 未使用镶嵌边，使用平均（Average）（有明显模糊）

结果分析：在尝试不同的镶嵌边设置和融合算法后，发现使用 Weighted Seamline 镶嵌边，并结合平滑（Smoothing）和羽化（Feathering）的方法效果最佳。由于镶嵌边搜索的位置是两景影响差距较小的路径，并且在镶嵌线周围进行了平滑和羽化操作，能够有效地减少重叠区域的拼接痕迹和模糊现象，使镶嵌后的影像在视觉上更加自然、连续。

对于其他方法，采用覆盖（Overlay）方法由于直接对左右两张影像采取简单的覆盖操作，而两景影像不可避免地在几何上会有一点误差，会导致所得影像在覆盖的直线边缘处**拼接痕迹明显；**而羽化（Feather）和平均（Average）方法则由于在几何配准时存在细小局部误差，导致这两景影像在重叠区域本应在同一位置的像素处于稍微错开的位

置，又被进行加权平均等像素值运算，使得重叠区域**出现明显的模糊**，影响影像的质量和可读性。这表明在进行影像镶嵌时，选择合适的镶嵌边和融合算法对于提高镶嵌效果至关重要。

最终选用的图 2.4.2-1 所示的使用 Weighted Seamline 镶嵌边，并使用平滑（Smoothing）和羽化（Feathering）得到的融合影像在重叠区域的过渡自然，没有明显的拼接痕迹和模糊现象，影像的整体性和一致性得到了较好的保持。这说明采用的镶嵌方法和参数配置是合理的，能够满足实际应用中对影像镶嵌质量的要求。这也给我在第 4 章使用 Python GDAL 库自己动手编程实现影像融合提供了一些重要的启示。

2.4.3 应用 ERDAS 软件的影像融合图

应用 ERDAS 软件的影像融合图的**最佳结果为图 2.4.3-1 所示的 HPF Resolution Merge 得到的结果，被用于后续处理步骤**，其余结果仅作为对比参考。

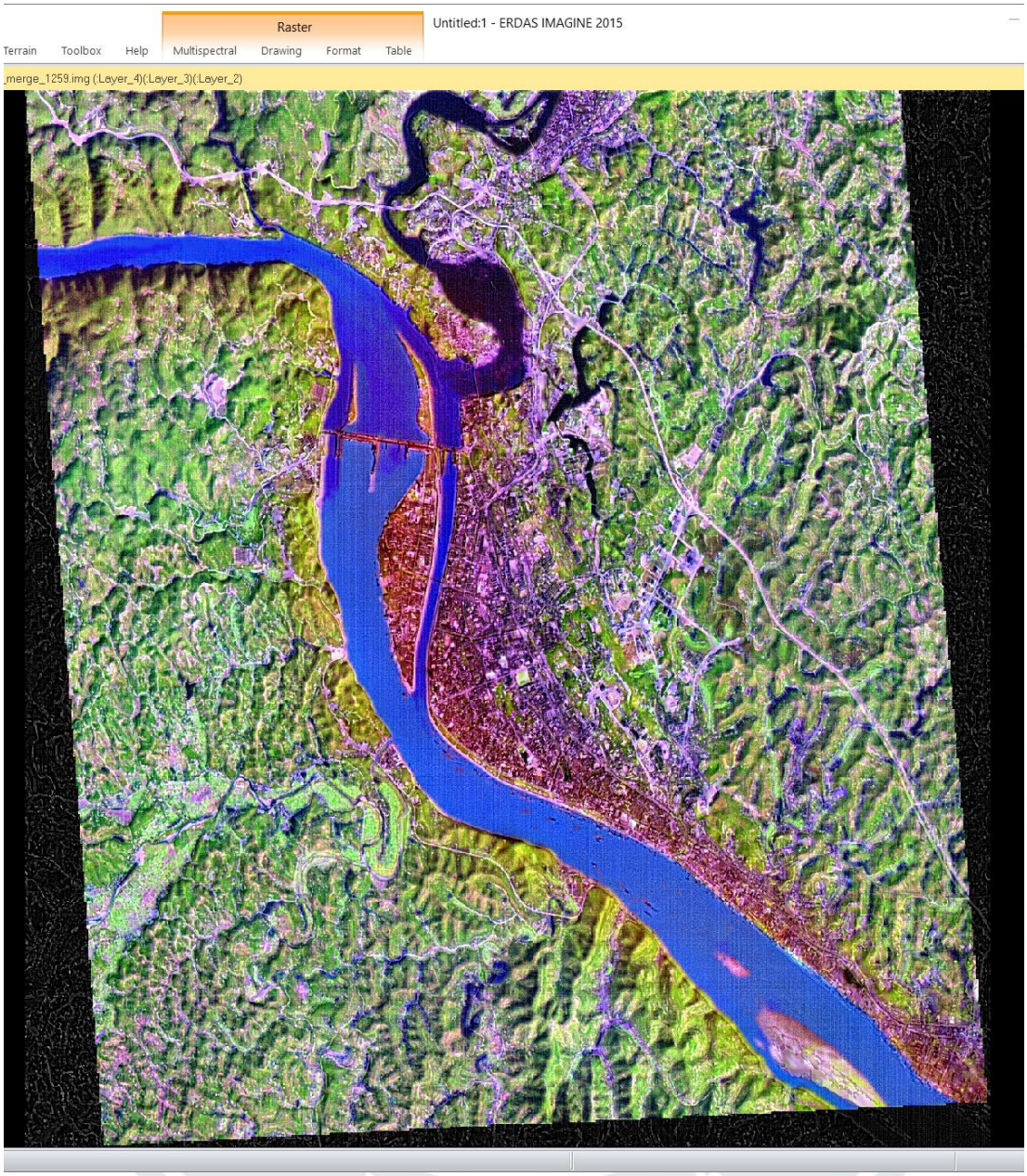


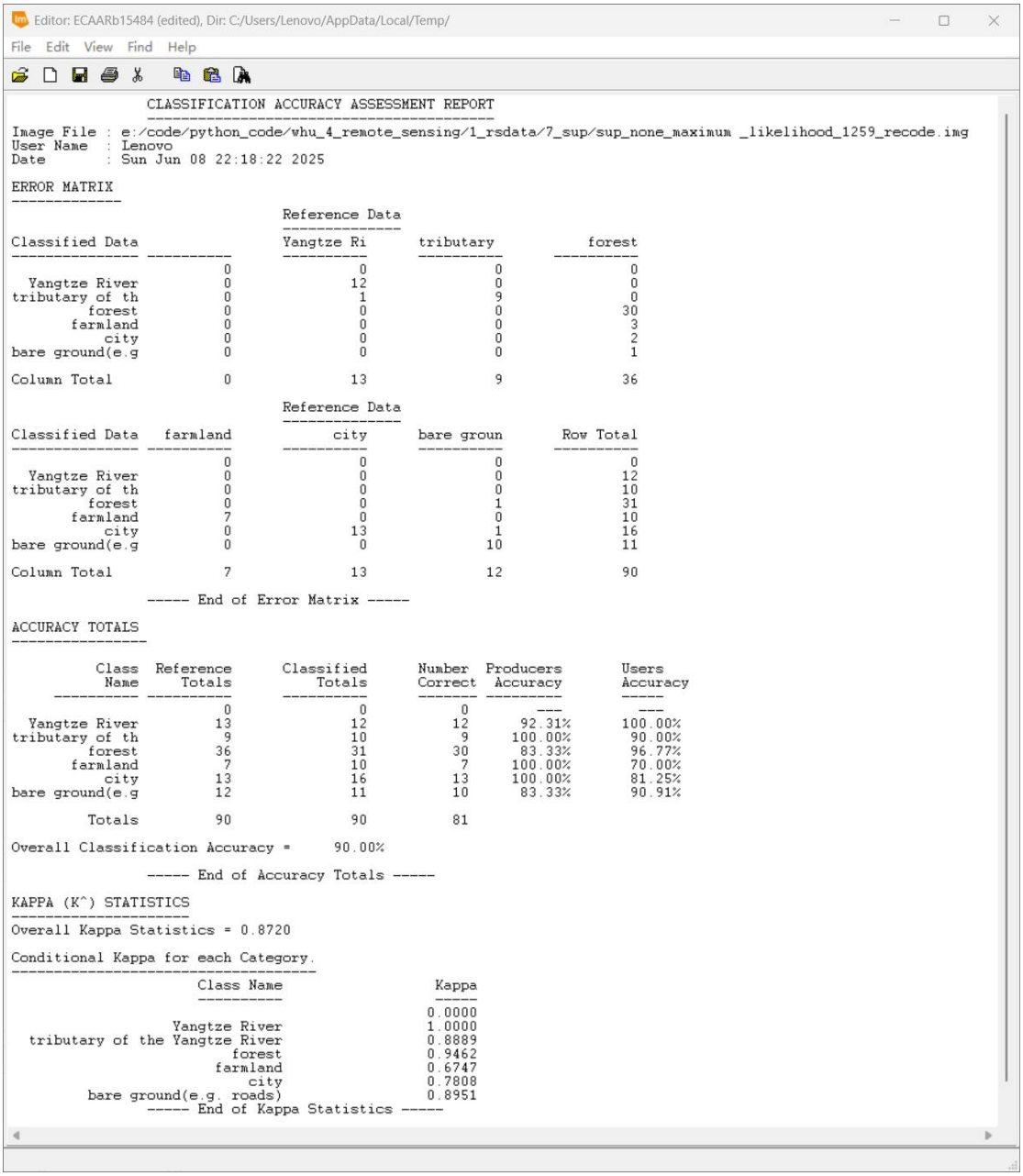
图2.4.3-1 HPF Resolution Merge的效果【最佳结果】

图2.4.3-2 Resolution Merge的效果（色彩存在失真）

结果分析：在这一步中作者使用了 HPF Resolution Merge 和普通的 Resolution Merge 两种影像融合方法。Resolution Merge 的融合结果存在一定的色彩失真（例如，请注意 Resolution Merge 融合结果中长江支流处的色彩异常等），而 HPF Resolution Merge 由于使用了高通滤波（High Pass Filtering），能够将高分辨率的 sp_yc.tif 影像的空间信息更加清晰地呈现在 TM 影像的光谱信息中，融合后的影像在细节表现和光谱信息的结合上更为出色，因此后续操作基于 HPF Resolution Merge 的结果进行。这表明 HPF Resolution Merge 方法在处理本次实习所使用的影像数据时，能够更好地平衡影像的空间分辨率和光谱信息，提高了影像的综合质量，为后续的分类等操作提供了更优质的数据基础。

2.4.4 应用 ERDAS 软件的分类精度评价图

应用 ERDAS 软件的分类精度评价图的**最佳结果为图 2.4.4-2 所示的平行六面体法（Parallelepiped）的精度评定报告中呈现的结果，被用于后续处理步骤**，其余结果仅作为对比参考。



Editor: ECAARb15484 (edited), Dir: C:/Users/Lenovo/AppData/Local/Temp/

File Edit View Find Help

```
               CLASSIFICATION ACCURACY ASSESSMENT REPORT
               -----------------------------------------
Image File  : e:/code/python_code/whu_4_remote_sensing/1_rsdata/7_sup/sup_none_maximum _likelihood_1259_recode.img
User Name   : Lenovo
Date        : Sun Jun 08 22:18:22 2025

ERROR MATRIX
-------------
                                 Reference Data
                                 --------------
Classified Data                  Yangtze Ri      tributary          forest
---------------- -----------     -----------     -----------     -----------
                           0               0               0               0
  Yangtze River            0              12               0               0
tributary of th            0               1               9               0
         forest            0               0               0              30
       farmland            0               0               0               3
           city            0               0               0               2
bare ground(e.g            0               0               0               1

Column Total               0              13               9              36

                                 Reference Data
                                 --------------
Classified Data   farmland            city      bare groun       Row Total
---------------- -----------     -----------     -----------     -----------
                           0               0               0               0
  Yangtze River            0               0               0              12
tributary of th            0               0               0              10
         forest            0               0               1              31
       farmland            7               0               0              10
           city            0              13               1              16
bare ground(e.g            0               0              10              11

Column Total               7              13              12              90

                ----- End of Error Matrix -----

ACCURACY TOTALS
----------------

         Class  Reference       Classified      Number  Producers       Users
          Name     Totals          Totals      Correct  Accuracy        Accuracy
     ---------- ----------      ----------      ------- ---------       -----
                           0               0           0       ---         ---
  Yangtze River           13              12          12     92.31%      100.00%
tributary of th            9              10           9    100.00%       90.00%
         forest           36              31          30     83.33%       96.77%
       farmland            7              10           7    100.00%       70.00%
           city           13              16          13    100.00%       81.25%
bare ground(e.g           12              11          10     83.33%       90.91%

         Totals           90              90          81

Overall Classification Accuracy =     90.00%

                ----- End of Accuracy Totals -----

KAPPA (K^) STATISTICS
---------------------
Overall Kappa Statistics = 0.8720

Conditional Kappa for each Category.
------------------------------------
                     Class Name                  Kappa
                     ----------                  -----
                                                 0.0000
                  Yangtze River                  1.0000
  tributary of the Yangtze River                 0.8889
                         forest                  0.9462
                       farmland                  0.6747
                           city                  0.7808
          bare ground(e.g. roads)                0.8951
                ----- End of Kappa Statistics -----
```

图2.4.4-1 最大似然分类法（Maximum Likelihood Classifier, MLC）的精度评定的报告

图 2.4.4-1 是报告的截图，可能看不清，最大似然分类法（Maximum Likelihood Classifier, MLC）的精度评定的报告的文字内容全文如下：

```
CLASSIFICATION ACCURACY ASSESSMENT REPORT
-----------------------------------------
```

```
Image File :
e:/code/python_code/whu_4_remote_sensing/rsdata/7_sup/sup_none_maximum
_likelihood_1259_recode.img
User Name  : Lenovo
Date       : Wed Jun 04 11:29:55 2025
ERROR MATRIX
-------------
    Reference Data
    --------------
Classified Data Background    Class 1    Class 2    Class 3
--------------- ---------- ---------- ---------- ----------
     Background          0          0          0          0
        Class 1          0         12          0          0
        Class 2          0          1          9          0
        Class 3          0          0          0         30
        Class 4          0          0          0          3
        Class 5          0          0          0          2
        Class 6          0          0          0          1
Column Total          0         13          9         36
    Reference Data
    --------------
Classified Data    Class 4    Class 5    Class 6  Row Total
--------------- ---------- ---------- ---------- ----------
     Background          0          0          0          0
        Class 1          0          0          0         12
        Class 2          0          0          0         10
        Class 3          0          0          1         31
        Class 4          7          0          0         10
        Class 5          0         13          1         16
        Class 6          0          0         10         11
Column Total          7         13         12         90
  ----- End of Error Matrix -----
ACCURACY TOTALS
----------------
         Class  Reference Classified Number Producers Users
          Name     Totals     Totals Correct  Accuracy Accuracy
     ---------- ---------- ---------- ------- --------- -----
        Class 0          0          0       0       ---   ---
        Class 1         13         12      12    92.31% 100.00%
        Class 2          9         10       9   100.00%  90.00%
        Class 3         36         31      30    83.33%  96.77%
        Class 4          7         10       7   100.00%  70.00%
        Class 5         13         16      13   100.00%  81.25%
        Class 6         12         11      10    83.33%  90.91%
```

```
        Totals        90        90     81
Overall Classification Accuracy =     90.00%
  ----- End of Accuracy Totals -----
KAPPA (K^) STATISTICS
---------------------
Overall Kappa Statistics = 0.8720
Conditional Kappa for each Category.
-----------------------------------
                                     Class Name          Kappa
                                     ----------          -----
                                        Class 0         0.0000
                                        Class 1         1.0000
                                        Class 2         0.8889
                                        Class 3         0.9462
                                        Class 4         0.6747
                                        Class 5         0.7808
                                        Class 6         0.8951
  ----- End of Kappa Statistics -----
```

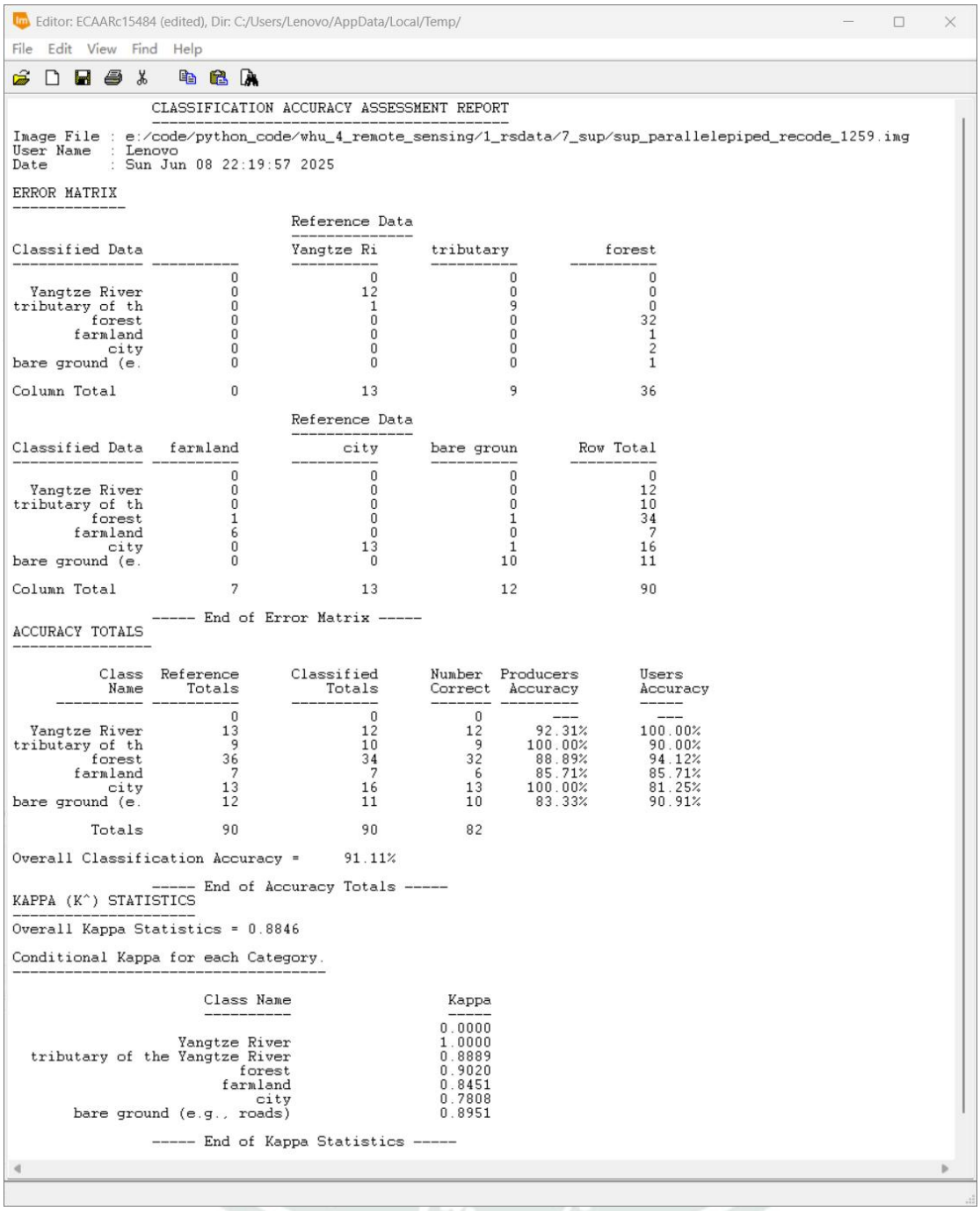


图2.4.4-2 平行六面体法（Parallelepiped）的精度评定报告【最佳结果】

图 2.4.4-2 是报告的截图，可能看不清，平行六面体法（Parallelepiped）的精度评定的报告的文字内容全文如下：

```
  CLASSIFICATION ACCURACY ASSESSMENT REPORT
  ----------------------------------------
Image File :
e:/code/python_code/whu_4_remote_sensing/rsdata/7_sup/sup_parallelepiped_recod
e_1259.img
User Name  : Lenovo
Date       : Thu Jun 05 14:52:35 2025
ERROR MATRIX
```

```
-------------
   Reference Data
   --------------
Classified Data           Yangtze Ri tributary      forest
--------------- ---------- ---------- ---------- ----------
                        0          0          0          0
  Yangtze River         0         12          0          0
tributary of th         0          1          9          0
        forest          0          0          0         32
      farmland          0          0          0          1
          city          0          0          0          2
bare ground (e.         0          0          0          1
Column Total          0         13          9         36
   Reference Data
   --------------
Classified Data   farmland       city bare groun  Row Total
--------------- ---------- ---------- ---------- ----------
                        0          0          0          0
  Yangtze River         0          0          0         12
tributary of th         0          0          0         10
        forest          1          0          1         34
      farmland          6          0          0          7
          city          0         13          1         16
bare ground (e.         0          0         10         11
Column Total          7         13         12         90
  ----- End of Error Matrix -----
ACCURACY TOTALS
----------------
         Class  Reference Classified Number Producers Users
          Name     Totals     Totals Correct  Accuracy Accuracy
     ---------- ---------- ---------- ------- --------- -----
                        0          0      0       ---   ---
  Yangtze River        13         12     12     92.31% 100.00%
tributary of th         9         10      9    100.00%  90.00%
        forest         36         34     32     88.89%  94.12%
      farmland          7          7      6     85.71%  85.71%
          city         13         16     13    100.00%  81.25%
bare ground (e.         12         11     10     83.33%  90.91%
        Totals         90         90     82
Overall Classification Accuracy =     91.11%
  ----- End of Accuracy Totals -----
KAPPA (K^) STATISTICS
---------------------
Overall Kappa Statistics = 0.8846
```

Conditional Kappa for each Category.

Class Name	Kappa
	0.0000
Yangtze River	1.0000
tributary of the Yangtze River	0.8889
forest	0.9020
farmland	0.8451
city	0.7808
bare ground (e.g., roads)	0.8951

----- End of Kappa Statistics -----

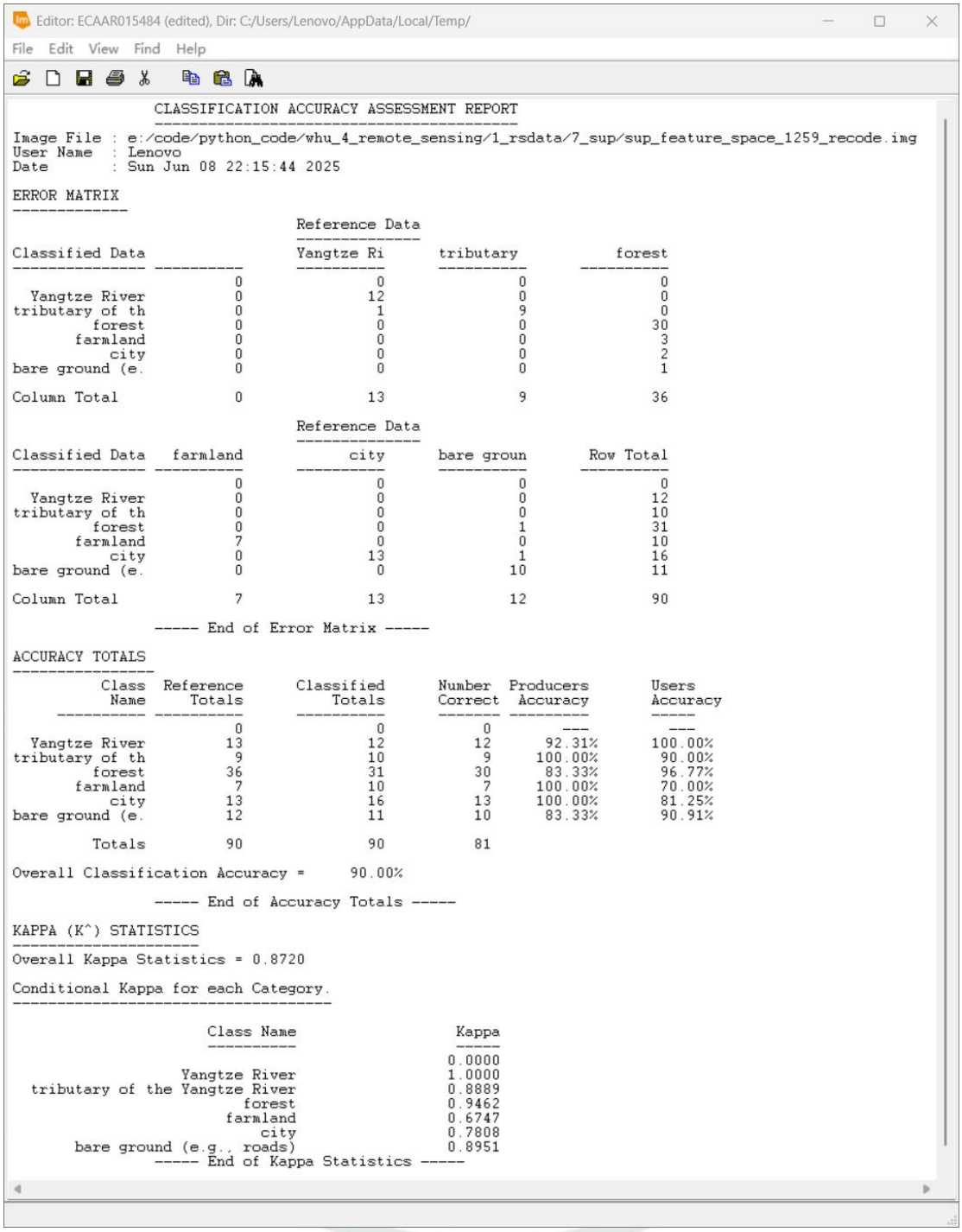


图2.4.4-3 特征空间法（Feature Space）的精度评定报告

图 2.4.4-3 是报告的截图，可能看不清，特征空间法（Feature Space）的精度评定的报告的文字内容全文如下：

```
  CLASSIFICATION ACCURACY ASSESSMENT REPORT
  ----------------------------------------
Image File :
e:/code/python_code/whu_4_remote_sensing/rsdata/7_sup/sup_feature_space_1259_r
ecode.img
User Name  : Lenovo
```

```
Date      : Thu Jun 05 15:07:12 2025
ERROR MATRIX
-------------
   Reference Data
   --------------
Classified Data           Yangtze Ri tributary     forest
--------------- ---------- ---------- ---------- ----------
                        0          0          0          0
  Yangtze River         0         12          0          0
tributary of th         0          1          9          0
        forest          0          0          0         30
      farmland          0          0          0          3
          city          0          0          0          2
bare ground (e.         0          0          0          1
Column Total          0         13          9         36
   Reference Data
   --------------
Classified Data   farmland      city bare groun  Row Total
--------------- ---------- ---------- ---------- ----------
                        0          0          0          0
  Yangtze River         0          0          0         12
tributary of th         0          0          0         10
        forest          0          0          1         31
      farmland          7          0          0         10
          city          0         13          1         16
bare ground (e.         0          0         10         11
Column Total          7         13         12         90
  ----- End of Error Matrix -----
ACCURACY TOTALS
----------------
         Class  Reference Classified Number Producers Users
          Name     Totals     Totals Correct  Accuracy Accuracy
    ---------- ---------- ---------- ------- --------- -----
                        0          0      0       ---   ---
  Yangtze River        13         12     12     92.31% 100.00%
tributary of th         9         10      9    100.00%  90.00%
        forest         36         31     30     83.33%  96.77%
      farmland          7         10      7    100.00%  70.00%
          city         13         16     13    100.00%  81.25%
bare ground (e.         12         11     10     83.33%  90.91%
        Totals         90         90     81
Overall Classification Accuracy =     90.00%
  ----- End of Accuracy Totals -----
KAPPA (K^) STATISTICS
```

```
---------------------
Overall Kappa Statistics = 0.8720
Conditional Kappa for each Category.
-----------------------------------
                    Class Name          Kappa
                    ----------          -----
                                       0.0000
                  Yangtze River         1.0000
 tributary of the Yangtze River          0.8889
                        forest          0.9462
                      farmland          0.6747
                          city          0.7808
      bare ground (e.g., roads)          0.8951
 ----- End of Kappa Statistics -----
```

结果分析：在本次实习中，我通过 ERDAS 软件对遥感影像进行监督分类主要采用了三种方法：最大似然分类法（Maximum Likelihood Classifier, MLC）、平行六面体法（Parallelepiped）和特征空间法（Feature Space）。

最大似然分类法（MLC）是一种基于统计的参数化方法，假设每个类别的光谱数据服从多元正态分布。该方法在本次实习中表现出较高的总体分类精度（90.00%）和 Kappa 系数（0.8720），说明分类结果具有较高的可靠性。其中，长江和长江支流的分类精度较高，生产者精度和用户精度均接近或达到 100%，表明这些地物在影像中的光谱特征较为明显，易于区分。森林和裸地的分类精度也较高，但农田和城市的分类精度相对较低，特别是农田的生产者精度仅为 70.00%，这可能是因为农田的光谱特征在不同季节和生长阶段存在较大变化，导致分类时出现一定的误差。

平行六面体法（Parallelepiped）通过定义多维的“盒子”来分类，若像元的光谱值落在某类别的盒子内则被归为该类别。该方法在本次实习中表现出最高的总体分类精度（91.11%）和 Kappa 系数（0.8846），说明其在处理本次数据时具有较好的分类效果。与最大似然分类法相比，平行六面体法在森林和农田的分类精度上略有提高，特别是森林的生产者精度达到了 88.89%，用户精度达到了 94.12%，说明该方法在处理这些地物时能够更好地识别其光谱特征。农田的分类精度虽然有所提高，但仍然存在一定的误差，这可能是因为农田的光谱特征在不同条件下变化较大，导致分类时难以完全准确。

特征空间法（Feature Space）基于像元在特征空间中的位置进行分类，特征空间由选择的波段（或主成分）构成。该方法在本次实习中的总体分类精度和 Kappa 系数与最大似然分类法相当，均为 90.00%和 0.8720。与最大似然分类法相比，特征空间法在森林的分类精度上略有提高，但农田的分类精度仍然较低，说明该方法在处理农田时存在一定的局限性。

总的来说，平行六面体法（Parallelepiped）在本次实习中表现出了最高的总体分类

精度（91.11%）和 Kappa 系数（0.8846），因此被认为是本次实习中最适合的监督分类方法。最大似然分类法（MLC）和特征空间法（Feature Space）的分类精度相当，但平行六面体法在某些地物（如森林）的分类精度上略胜一筹。最大似然分类法适用于光谱特征较为明显且服从正态分布的地物，如长江和长江支流；平行六面体法适用于光谱特征较为集中且可以通过多维“盒子”定义的地物，如森林和城市；特征空间法：适用于需要通过特定波段或主成分进行分类的场景，但在处理光谱特征变化较大的地物（如农田）时可能效果不佳。

在后续研究中，可以尝试结合多种分类方法的优势，例如先使用特征空间法进行初步分类，再通过最大似然分类法或平行六面体法进行优化，以进一步提高分类精度。对于农田等光谱特征变化较大的地物，可以考虑引入更多的辅助数据（如地形、土壤类型等）来提高分类的准确性。

2.4.5 应用 ERDAS 软件的分类专题信息制图

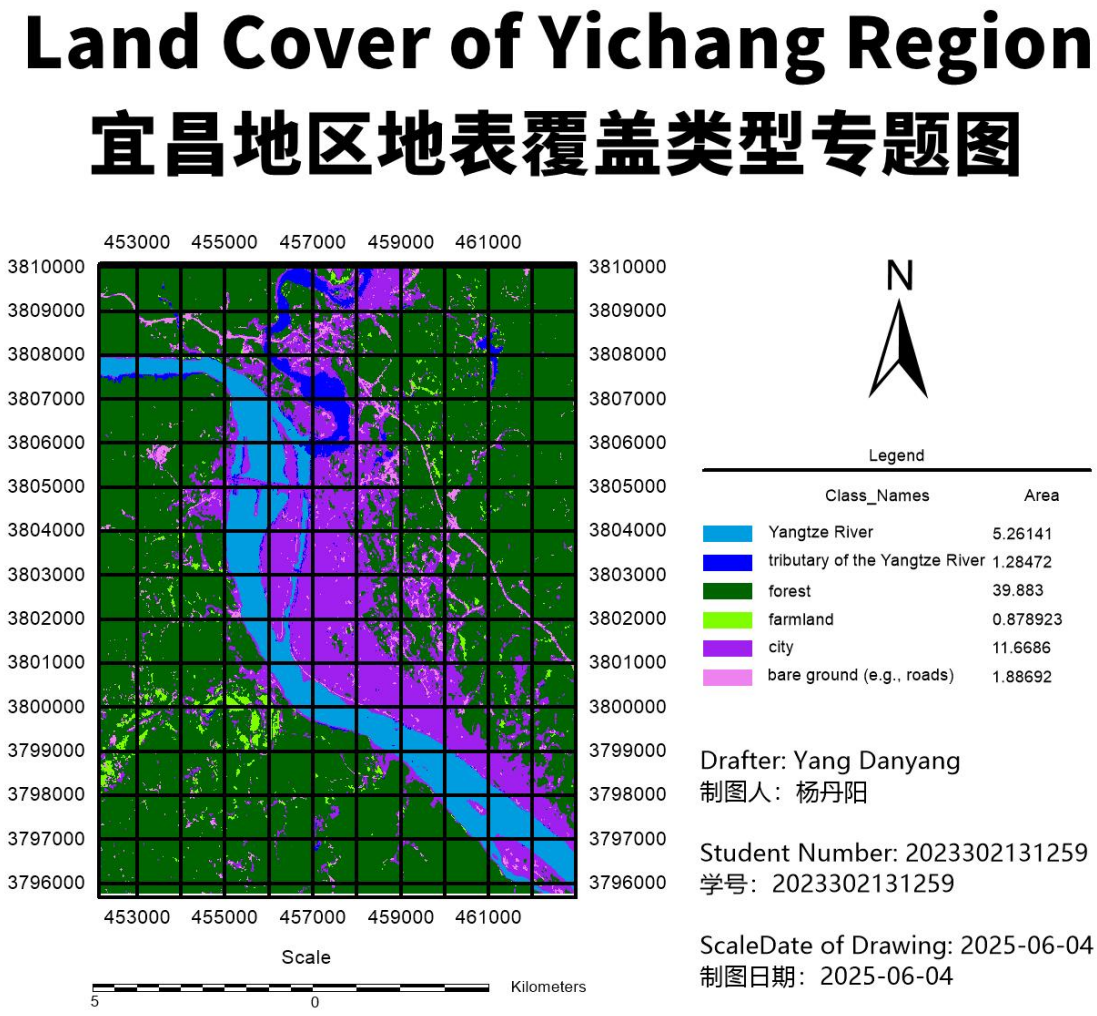


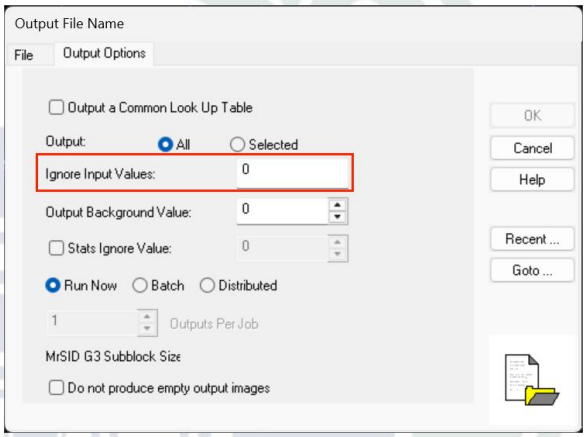
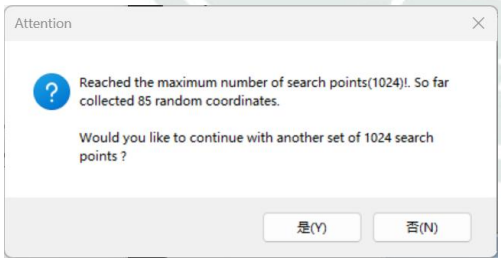
图2.4.5-1 宜昌地区地表覆盖类型专题图

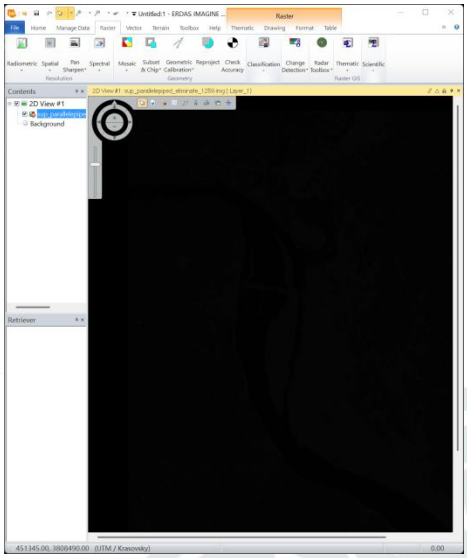
结果分析：最终制作完成的的宜昌地区地表覆盖类型专题图清晰地展示了该地区的地表覆盖类型分布情况，包括长江、长江支流、森林、农田、城市和裸地等不同地物类别。专题图通过不同的颜色和图例，直观地表达了各类地物的空间分布和范围，能够为相关领域的研究和决策提供直观的地理信息支持。专题图的布局合理，包含了地图标题、制图人信息、制图日期、比例尺、图例等必要的地图元素，使地图信息更加完整、易读。同时，通过添加指北针和网格等辅助元素，增强了地图的可读性和实用性。

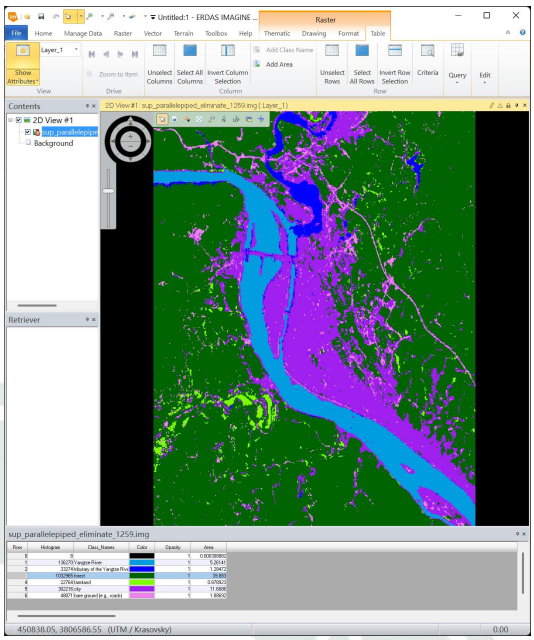
该专题图可以应用于多个领域，如城市规划、环境保护、土地利用研究等。例如，在城市规划中，可以依据专题图了解城市周边的土地利用情况，为城市的扩展和基础设施建设提供参考；在环境保护方面，可以分析森林、农田等生态系统的分布和变化，为生态保护和修复措施的制定提供依据。专题图还能够为相关研究提供基础数据支持，如

在进行区域生态评估、土地利用变化分析等研究时，可以作为重要的数据源，帮助研究人员更好地理解和分析研究区域的地表覆盖特征及其变化规律。

2.5 实习过程中遇到的问题及其解决方法

遇到的问题	解决方法
几何校正控制点选取困难，部分区域地物特征不明显，导致同名点匹配难度大	通过放大视图、结合影像纹理特征（如道路交叉口、水体边缘）提高选点精度
ERDAS软件操作不熟悉，部分工具的位置隐蔽，难以找到	通过Help搜索功能快速定位
不使用镶嵌边进行影像拼接得到的影像中存在阴影	通过点击导出影像拼接窗口上方的Output Options可以设置“Ignore Input Values”为0可以解决这一问题（如下图所示） 
ISODATA算法设置最大分类数为10时，部分类别实际属于同一地物，导致分类结果冗余	通过重编码（Recode）工具合并实际上属于同一类地物的冗余类别
生成随机检查点时，系统提示未找到足够多的满足要求的随机点（如下图所示） 	通过设置生成随机检查点的属性，增加搜索随机检查点的数量，以扩大随机点的搜索范围，从而确保某些像素占全图比例小的地物类型也可以找到足够多的随机点
在进行精度评定中的目视判读步骤时，一开始的时候忘记Save Tabel了，导致目视判读结果在关掉窗口之后找不回来了	养成好习惯：在目视判读结束后及时点击Save Tabel，这样下次打开那个img的时候还能恢复相关数据
在ERDAS软件中进行后处理时，一开始打开处理后的影像发现全是黑色（如下图所示）：	经过检查后我发现是颜色配置问题，重新配置后才得到正常显示的效果如下图所示：





这说明在操作过程中需要仔细检查每个步骤的设置，避免因疏忽导致错误结果

表2.5-1 在应用ERDAS软件进行遥感分类专题信息提取与制图的实习过程中遇到的问题及其解决方法

2.6 实习小结

在本部分的实习中，我通过应用 ERDAS 软件成功完成了遥感分类专题信息提取与制图的全过程，达到了预期的实习目的。在实习过程中，我深入学习并掌握了波段叠加、几何校正、影像融合、分类、精度评定和专题制图等关键技术，为今后从事遥感图像处理和地理信息分析工作奠定了坚实的基础。

在波段叠加与几何校正环节，我学会了如何将多波段影像集成到一个数据集中，并通过选择控制点和建立几何模型，成功地将影像与地理坐标系统对齐，确保了后续分析的准确性。在影像融合部分，通过对比不同融合算法的效果，选择了最适合的融合方法，有效地提升了影像的空间分辨率和光谱信息质量。在分类环节，我尝试了监督分类和非监督分类两种方法，通过对比不同分类算法的结果，掌握了如何根据具体需求选择合适的分类方法，并通过精度评定指标评估了分类结果的准确性。最后，在专题制图环节，我学会了如何设计地图布局、选择合适的符号和颜色，以及添加辅助信息，成功地将分类结果以直观的地图形式展示出来，为地理信息的可视化表达提供了有力支持。

通过应用 ERDAS 软件进行遥感分类专题信息提取与制图，我不仅提高了自己的实践操作能力，还加深了对遥感图像处理理论知识的理解，培养了解决实际问题的能力。同时，我在实习过程中也遇到了一些问题和挑战，如控制点的选择和精度评定的细节处理等，但通过不断尝试和探索，我逐渐克服了这些问题，积累了宝贵的经验。我相信，

这些经验和技能将在未来的学术研究和工作中发挥重要作用，为我打开更广阔的发展空间。

3 应用 OGE 平台的基于特征指数遥感专题信息提取

3.1 实习目的

开放地球引擎（Open Geospatial Engine，OGE）是武汉大学龚健雅院士团队研发的全栈自主的时空信息基础设施，于 2024 年 1 月正式发布上线。该平台瞄准“数字地球”时空信息服务需求，具备“算力-算法-数据”弹性耦合和开放共享能力。OGE 依托于云计算设施，动态汇聚与管理全球范围内的地球观测数据、数字高程模型、定量遥感产品、遥感 AI 样本、虚拟星座数据等多源异构时空数据。

目前，开放地球引擎服务平台已成为一个集海量地理空间数据、交互式编程分析、实时分布式计算和数据可视化为一体的在线时空数据分析云平台。在数据上，平台汇聚了中国高分、欧洲哨兵（Sentinel）、美国陆地资源（Landsat）等系列卫星遥感数据以及海量遥感样本数据，具备多源海量卫星影像、航空影像、高程产品、基础地理、普通文件、Web 标准服务等多类型数据的资源整合、集中配置和统筹管理能力；在系统设计与运算支持上，平台采用云原生技术，以 Kubernetes 集群容器化编排架构进行设计，支持用户从多终端随时访问平台功能与资源，提供云原生算子、基础算子、专业模型 200 余个，用户可通过 JavaScript 和 Python 脚本语言调用海量数据和计算资源进行大规模地理数据实时计算分析；在应用上，平台提供各类专业级的分析应用，开箱即用，包括定量遥感产品、虚拟星座、三维重建系统、时空知识图谱、水文模型、碳排放模型等，在“一带一路”、全球测绘、数字孪生地球建设中发挥着空间信息技术的独特优势，可为自然资源管理、城市治理、公共服务、智慧交通、国防安全、灾害救援、生态建设等提供可控的地球时空信息基座。

本次实习旨在通过应用开放地球引擎（Open Geospatial Engine，OGE）平台，深入掌握基于特征指数的遥感专题信息提取技术，提升对遥感数据处理与分析的能力，同时熟悉 OGE 平台的操作流程与功能，为后续的遥感应用研究和实践奠定坚实基础。具体来说，需要**理解遥感特征指数的原理与应用**，深入学习植被指数（如 RVI、NDVI）、水体指数（如 NDWI、MNDWI、AWEI）和建筑指数（如 NDBI、IBI）的构建原理及其在定量遥感分析中的作用，掌握不同指数的适用场景与局限性，能够根据实际需求选择合适的指数进行地物信息提取；**熟悉 OGE 平台的功能与操作**，通过实践操作，熟练掌握

OGE 平台的常用功能，包括查找和调用多源卫星影像数据、上传自定义数据、进行交互式编程分析、导出处理结果以及数据可视化等，体会 OGE 平台在遥感数据处理与分析中的便捷与高效；**提升遥感数据处理与编程能力**，运用 Python 等编程语言调用 OGE 平台提供的相关库函数，编写多种特征指数计算代码，将理论公式转化为可执行的代码，并通过调试优化代码以获得准确的结果，从而巩固编程基础，提升对遥感数据处理流程的理解；**培养地理信息分析与应用能力**，对提取结果展开分析，根据指数值的范围和分布特征来判断地物类型，并结合实际地理背景对结果进行解读，探讨不同地物类型之间的关系及其对环境的影响，培养运用遥感技术解决实际地理问题的能力，并了解遥感技术在生态环境保护、水资源管理、城市规划、土地利用监测等领域的应用前景，体会其在解决实际问题中的巨大潜力，激发对遥感技术深入学习和研究的兴趣，为未来的职业发展与学术研究打下基础、提供启示。

3.2 实习基本原理

遥感特征指数是基于地物光谱反射特性差异而构建的数学表达式，通过不同波段反射率的组合运算来增强特定地物信息，抑制其他地物干扰。这些指数充分利用了地表覆盖类型在电磁波谱不同波段上的差异化响应特征，是定量遥感分析的重要工具。

3.2.1 植被指数

植被在可见光和近红外波段具有显著的光谱特征（图 3.2.1-1）：**在红光波段（0.6-0.7μm）和蓝光波段（0.4-0.5μm）**由于叶绿素（叶绿素 a 形成以 0.45μm 为中心的吸收带，叶绿素 b 形成以 0.66μm 为中心的吸收带）和类胡萝卜素（胡萝卜素、叶黄素在 0.43μm-0.48μm 形成吸收带）进行光合作用而的产生强烈吸收，**反射率较低**，仅在**绿光波段**（0.5-0.6μm）有一个**小的反射峰**；在**红边区域**（Red Edge Position，700-750nm）是**反射率急剧上升**的过渡区域；在**近红外波段**由于叶片内部细胞壁的散射作用，产生**反射率高**（40-60%）的近红外平台（750-1300nm）；在短波红外区域（1300-2500nm）受叶片含水量影响显著，在 1450nm 和 1950nm 附近出现反射率相对较低（10-40%）的水分吸收谷。其中，**“红边效应”**为植被指数构建提供了理论依据[16]。

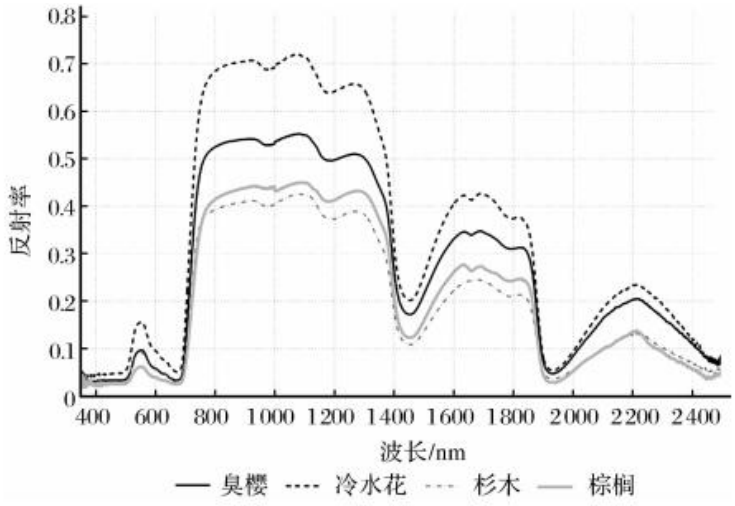


图3.2.2.1-1 4种植被的光谱曲线

植物波谱具有上述基本特征，但仍有细部差别，这种差别与植物种类（图 3.2.2.1-1）、季节、病虫害影响、含水量多少等有关系。

3.2.2.1 比值植被指数（RVI）

比值植被指数 RVI（Ratio Vegetation Index）是最简单的植被指数，其计算公式为：

$$RVI = \frac{NIR}{Red}$$[17]

其中，NIR为近红外波段反射率，Red为红光波段反射率。

RVI 的值越大，表示该处植被越茂盛。植被的 RVI 通常大于 2，绿色健康植被覆盖地区的 RVI 远大于 1，无植被覆盖的地面（裸土、人工建筑、水体、植被枯死或严重虫害）的 RVI 在 1 附近。

RVI 通过简单的比值运算放大了植被在两个波段间的反射差异，但它也具有明显的缺陷：当植被覆盖度较高时，RVI 对植被十分敏感，但当植被覆盖度<50%时，这种敏感性显著降低；易受大气和土壤背景影响，大气效应大大降低对植被检测的灵敏度，所以在计算前需要进行大气校正，或用反射率计算 RVI。

3.2.2.2 归一化植被指数（NDVI）

归一化植被指数 NDVI（Normalized Difference Vegetation Index）通过归一化处理克服了 RVI 的不足，其计算公式为：

$$NDVI = \frac{NIR-Red}{NIR+Red}$$[18]

NDVI 的值域为[-1, 1]，当 NDVI > 0.2 时表示有植被覆盖，NDVI > 0.4 是表示植被

覆盖度较高，NDVI 接近 0 时表示裸土或建筑用地，NDVI < 0 时通常表示水体。

NDVI 具有使用广泛、不依赖绝对反射率等优点。然而，NDVI 也存在一些局限性，如在植被稀疏的区域，裸土的反射也会干扰 NDVI 的数值，因此可以考虑结合 EVI 或 SAVI 等其他植被指数指数，以降低大气、土壤等其他因素的影响。

3.2.2 水体指数

水体的**整体反射率均较低**，在可见光波段（400-760nm）清洁水体整体呈现蓝色或蓝绿色，在蓝绿波段（450-550nm）反射率相对较高，在红光波段（650-700nm）水体吸收较强，反射率低；而水体在近红外（700-1300nm）和短波红外波段（1300-2500nm）由于强烈吸收而**反射率极低（接近于 0）**，这是区分水体与其他地物的重要特征[19][20][21]。这种光谱特性为水体指数的构建提供了物理基础。

水体指数在识别河流湖泊、洪泛区等需要基于遥感影像进行地表水体提取的应用场景中具有重要作用。

水体的光谱特性还受到诸多因素的影响。水体中的**叶绿素含量**会使得水体整体反射率提高，出现绿光波段的反射峰和红边效应等植被的波谱特性（图 3.2.2-1），可应用于监测赤潮和水华现象；水体中**泥沙等悬浮物**的增加也会增加水体整体的反射率，使光谱曲线上移，但光谱曲线仍保持相对平缓，峰值出现在黄红区，近红外区域仍保持低反射特征；此外，**水深**也会影响水体的响应特性，对于清洁水体，当水较浅时底质反射影响明显，只有当水体达到一定深度后起光谱特性才主要表现为水体本身的光谱特征。

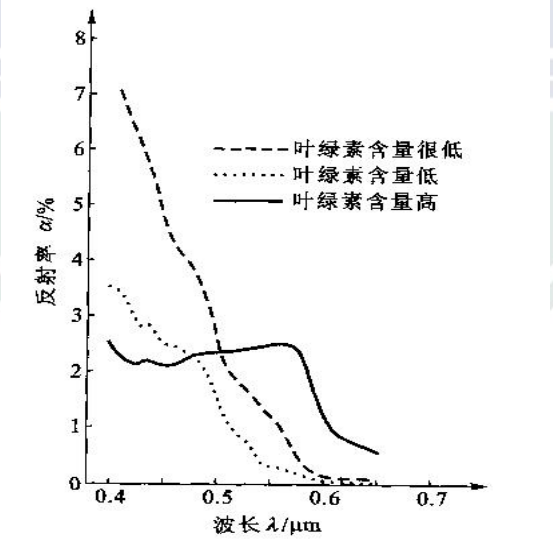


图3.2.2-1 叶绿素浓度增加时，蓝光反射率显著下降，绿光反射率显著上升

3.2.2.1 归一化水体指数（NDWI）

NDWI（Normalized Difference Water Index）利用绿光和近红外波段的反射差异识别水体：

$$NDWI = \frac{Green-NIR}{Green+NIR}$$ [22]

NDWI 的值域为[-1,1]，当$NDWI < 0$ 时，通常被认为是非水体；当$NDWI \geq 0$ 时，则可能是水体。

NDWI 计算简便，但易受建筑与阴影的影响，常常将其误认为是水体[23]，因此在实际应用中需要使用能够更加精确地反应水体特征的水体指数。

3.2.2.2 修正归一化水体指数（MNDWI）

MNDWI（Modified Normalized Difference Water Index）用短波红外波段替代近红外波段，提高了对建筑阴影的区分能力：

$$MNDWI = \frac{Green-SWIR1}{Green+SWIR1}$$ [24]

3.2.2.3 自动水体提取指数（AWEI）

AWEI 包含两个版本，分别适用于不同场景：

$AWEI_{nsh}$（无阴影优化）：

$$AWEI_{nsh} = 4 \times (Green - SWIR1) - 0.25 \times NIR - 2.75 \times SWIR2$$

$AWEI_{sh}$（有阴影优化）：

$$AWEI_{sh} = Blue + 2.5 \times Green - 1.5 \times (NIR + SWIR1) - 0.25 \times SWIR2$$

AWEI 的取值为实数范围，正值代表大概率为水体。

AWEI 具有抑制阴影与建筑干扰的优点[25]，并且可以根据不同的地形条件，在两种 AWEI 的公式中进行选择，例如对于地形起伏较大的山区或者房屋密集的城区，由于阴影区域面积较大，则可以选用有阴影优化的 AWEI 公式。

3.2.3 建筑指数

建筑用地通常由混凝土、沥青等人工材料构成，**在短波红外波段具有较高反射率，而在近红外波段反射率相对较低**，这与植被光谱特性形成对比（植被由于细胞壁的散射作用而近红外反射较强、由于叶片中水的吸收作用而中红外反射较弱）。

对比不同时期遥感影像建筑指数的变化，可以识别城市的边界或其中建筑的变化，用于监测城市扩张或识别违章建筑；同时，建筑密集区域往往热辐射增强，可将建筑指数与热红外进行综合分析，研究城市的热岛效应等。

3.2.3.1 归一化建筑指数（NDBI）

NDBI 通过短波红外和近红外波段的归一化差值识别建筑用地：

$$NDBI = \frac{SWIR1-NIR}{SWIR1+NIR}$$[26]

NDBI 的值域为[-1,1]。当 NDBI 的值接近 1 时，为高反射率的建筑区域；当的 NDBI 值接近 0 时，为裸地或混合地物区域。NDBI 计算简单，但也存在将裸地误判为建筑的问题[27]。

3.2.3.2 指数型建筑指数（IBI）

IBI 是一个复合指数，综合考虑了 NDBI、SAVI 和 MNDWI[28]：

$$IBI = \frac{NDBI - \frac{SAVI + MNDWI}{2}}{NDBI + \frac{SAVI + MNDWI}{2}}$$

其中，SAVI（土壤调节植被指数）计算公式为：

$$SAVI = \frac{(NIR - Red) \times (1 + L)}{NIR + Red + L}$$

式中$L \in [0,1]$为土壤调节因子，在本次实习中我的代码里面取 0.1。IBI 融合了建筑、水体和植被的指数信息，对建筑区域的识别精度高，但需要计算三个指数，计算过程较为繁琐。

3.3 实习过程

3.3.1 OGE 平台中的常用功能的操作方法

在介绍本次实习过程中使用的基于特征指数遥感专题信息提取的代码和得到的结果之前，先统一介绍一下在 OGE 平台中的常用功能的操作方法，3.3.2 中所述的各种基于特征指数遥感专题信息提取的操作方法类似，不同只在于代码的设计上。**请注意：本节不重点阐述相关操作的原理（对相关原理的解释请参见 3.2 节），也不对结果展开具体分析（对所得结果的分析请参见 3.4 节）。**

3.3.1.1 查找 OGE 平台中的其他卫星影像数据

点击资源中心，可以在对地观测数据中找到相应的卫星影像信息（图 3.3.1.1-1）：

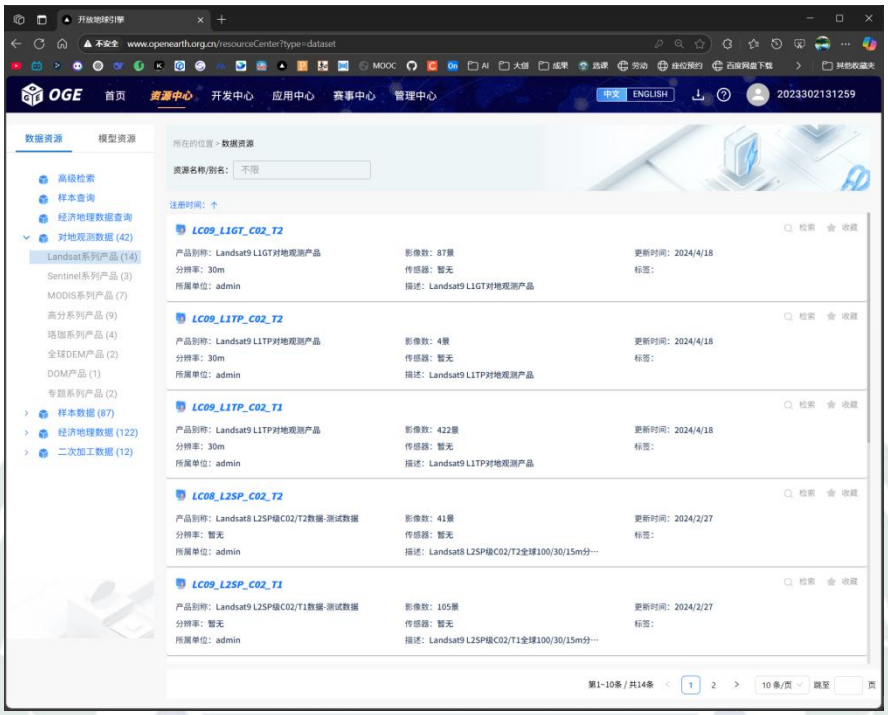


图3. 3. 1. 1-1 在资源中心-对地观测数据中找到相应的卫星影像信息

点击想要调用的卫星，即可看到该卫星的波段信息与调用代码示例（图 3.3.1.1-2）：

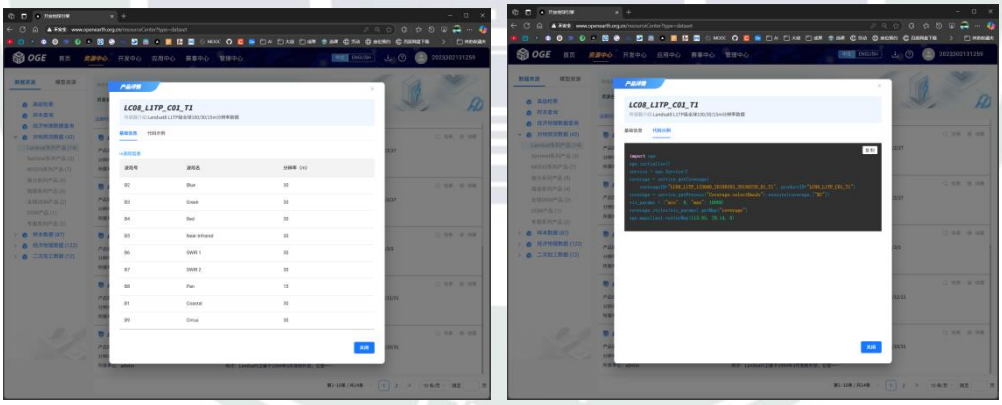


图3. 3. 1. 1-2 卫星的波段信息与调用代码示例

也可以点击资源中心页面上的高级检索，输入查询条件，输入完成点击提交查询，会得到若干个符合条件的影像，即图 3.3.1.1-3 上的白色方框（其中的红色方框是设置的经纬度范围）：

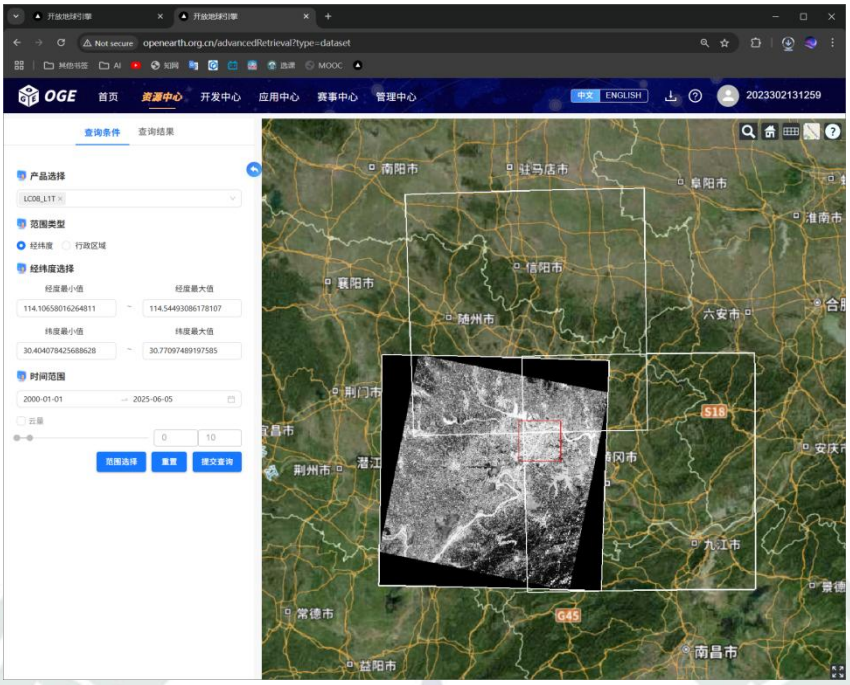


图3.3.1.1-3 在资源中心页面的高级检索中输入查询条件

此时页面也会跳转到图 3.3.1.1-4 所示的界面，可以勾选自己想要使用的遥感影像，选择下载。

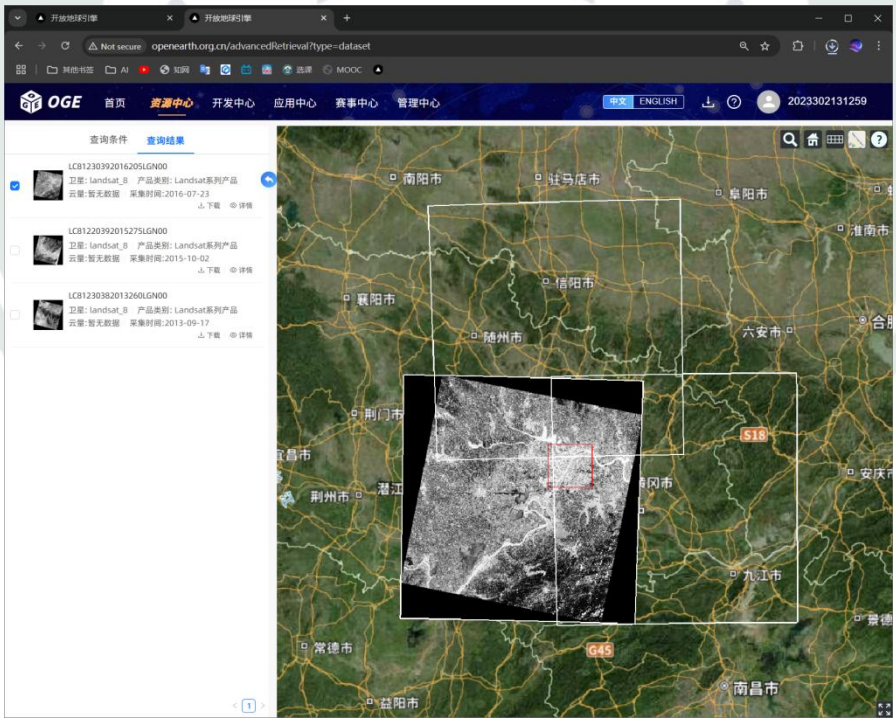


图3.3.1.1-4 卫星影像的查询结果

也可以点击影像右下角的“详情”查看其详细信息（图 3.3.1.1-5）：

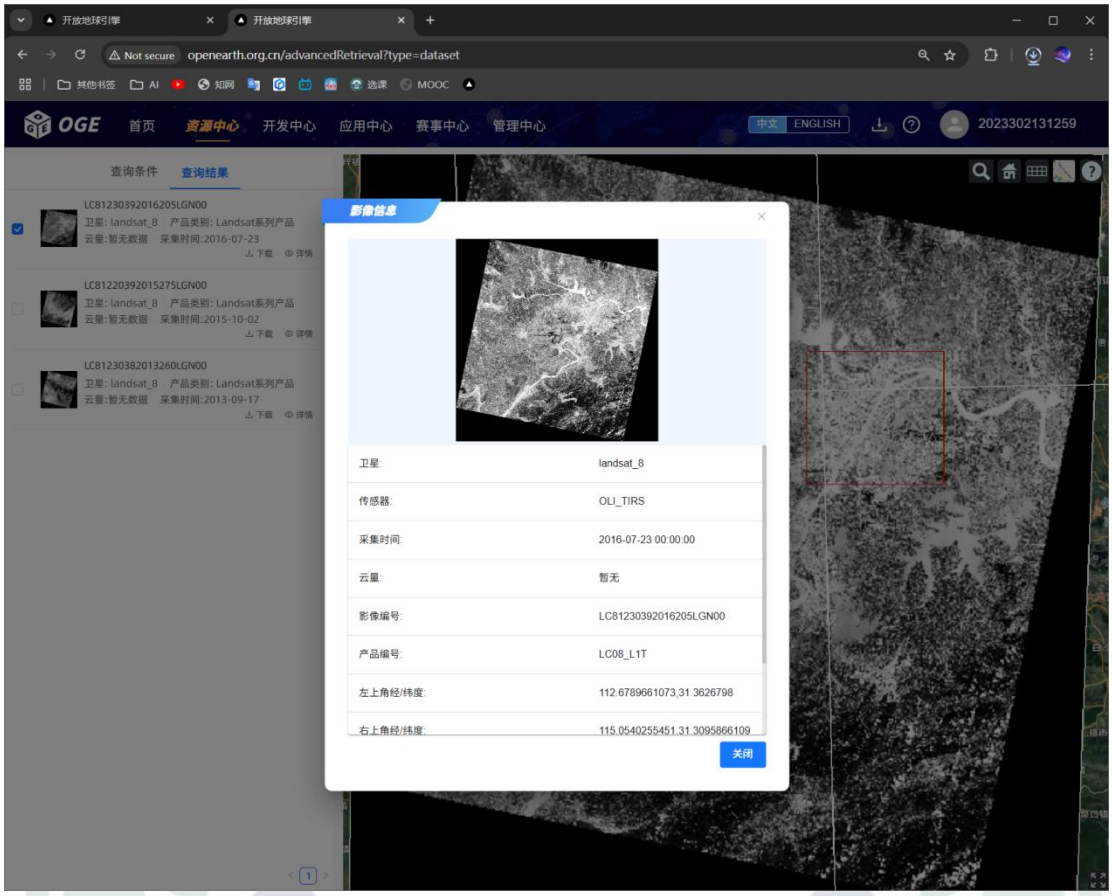


图3. 3. 1. 1-5 查看遥感影像的详细信息

如图 3.3.1.1-5 所示，该遥感影像的详细信息为：

卫星： landsat_8

传感器： OLI_TIRS

采集时间：2016-07-23 00:00:00

云量： 暂无

影像编号：LC81230392016205LGN00

产品编号：LC08_L1T

左上角经/纬度:112.6789661073,31.3626798

右上角经/纬度:115.0540255451,31.3095866109

左下角经/纬度:112.6437882525,29.2787064675

右下角经/纬度:114.96931265,29.2298433839

上述影像编号和产品编号即可在代码中调用（如：img1 = service.getCoverage(coverageID="**LC81230392016205LGN00**", productID="**LC08_L1T**")）。需要注意的是，虽然同样是 Landsat 卫星的影像，Landsat-4、5 上的 TM（Thematic Mapper，专题制图仪）和 Landsat-8、9 上的 OLI（Operational Land Imager，陆地成像仪）的波段设置是不同的（图 3.3.1.1-6）：

图3.3.1.1-6 TM与OLI的波段设置的不同

TM 的波段设置如表 3.3.1.1-1 所示：

波段号	波段名	分辨率（m）
B1	Blue	30
B2	Green	30
B3	Red	30
B4	Near-Infrared	30
B5	SWIR 1	30
B7	SWIR 2	30

表3.3.1.1-1 TM的波段设置

OLI 的波段设置如表 3.3.1.1-2 所示：

波段号	波段名	分辨率（m）
B2	Blue	30
B3	Green	30
B4	Red	30
B5	Near-Infrare	30
B6	SWIR 1	30
B7	SWIR 2	30
B8	Pan	30
B1	Coastal	30
B9	Cirrus	30

表3.3.1.1-1 TM的波段设置

3.3.1.2 上传资源

如图 3.3.1.2-1，点击开发中心-资源面板-上传资源，可以上传 TIF 等格式的遥感影像数据：

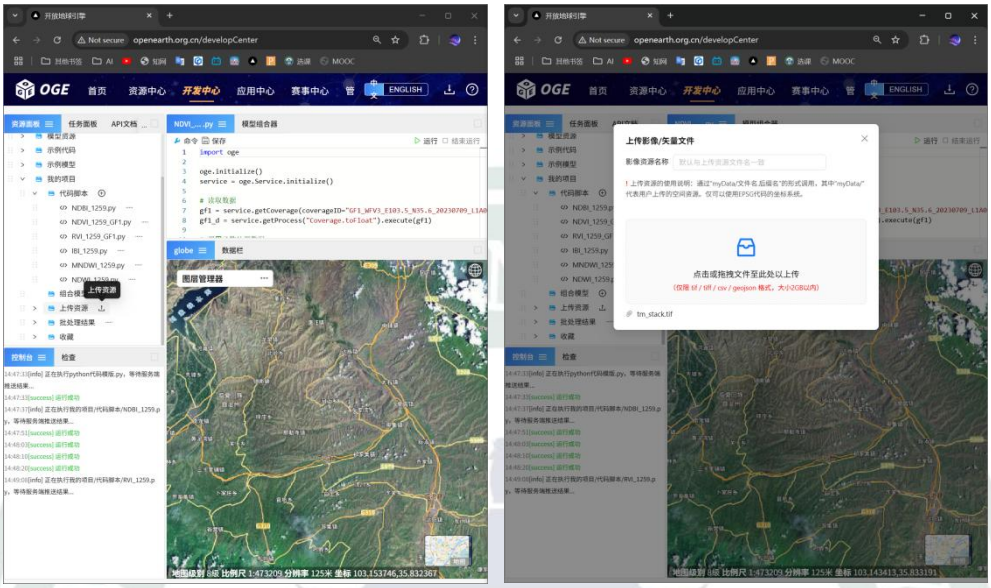


图3. 3. 1. 2-1 上传用户自己的影像数据

导入后，在开发中心-资源面板-上传资源文件夹下（在代码中是“myData”文件夹下）即可看到上传的影像数据（图 3.3.1.2-2）。

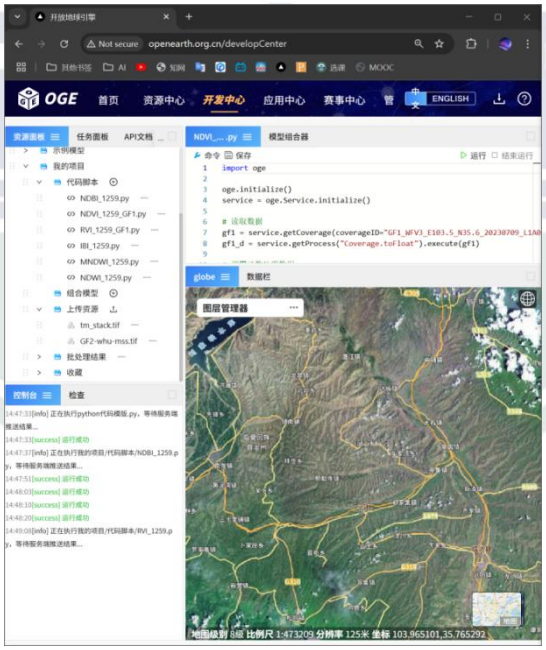


图3. 3. 1. 2-2 上传的影像数据在开发中心-资源面板-上传资源文件夹下

3.3.1.3 导出处理结果

在开发中心运行导出处理结果的代码（如：rvi.styles(vis_params).**export**("rvi")）后，在“任务面板”中点击出现的“提交”按钮（图 3.3.1.3-1）：

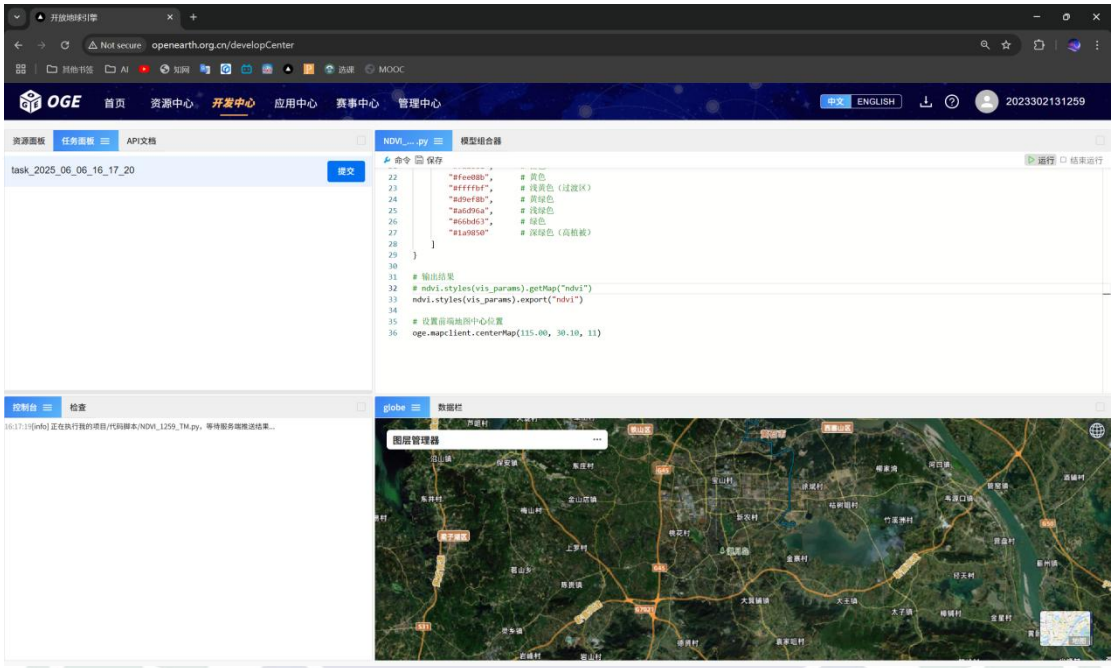


图3. 3. 1. 3-1 点击在“任务面板”中出现的“提交”按钮

然后设置导出文件的文件名、分辨率等信息（图 3.3.1.3-2）。虽然需要导出的遥感影像一般分辨率是 30m，但是由于目前系统算力的限制，最高的分辨率只能设置为 40m。

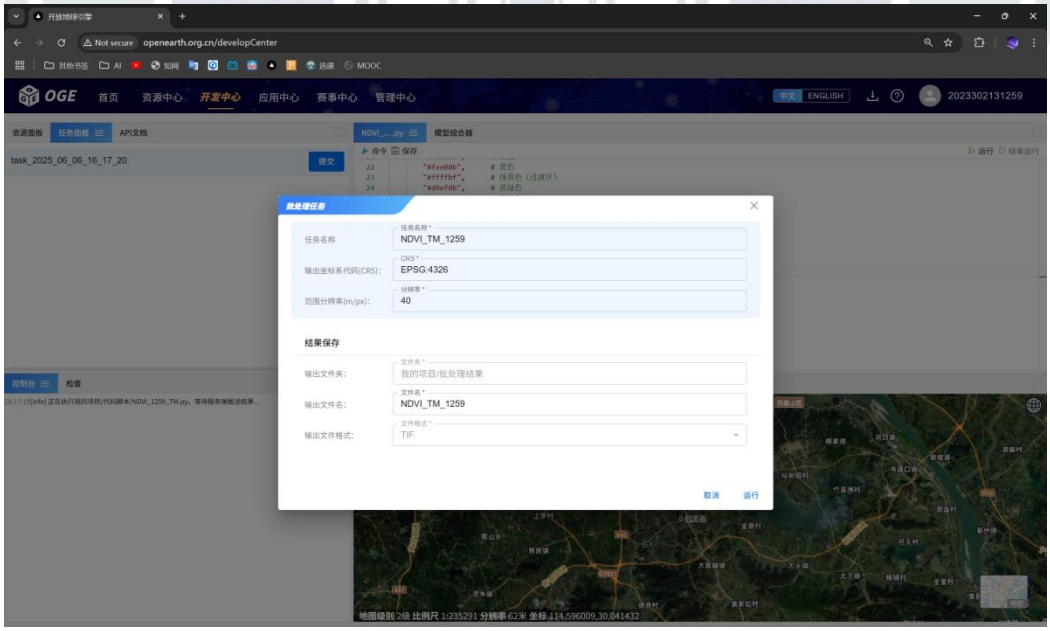


图3. 3. 1. 3-2 设置导出文件的信息

然后等待任务执行完毕（图 3.3.1.3-3）：

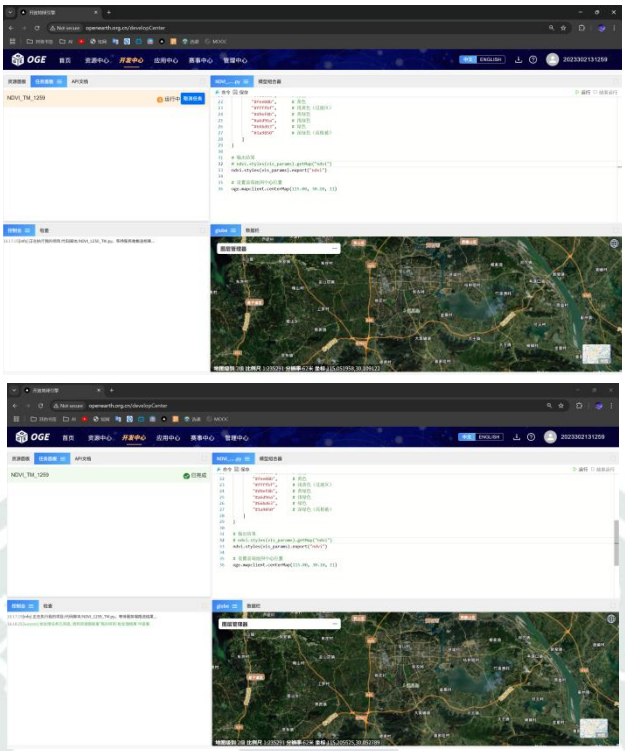


图3.3.1.3-3 等待任务执行完毕

然后在开发中心的“批处理结果”下找到刚刚得到的处理结果，选择“下载”（图3.3.1.3-4）：

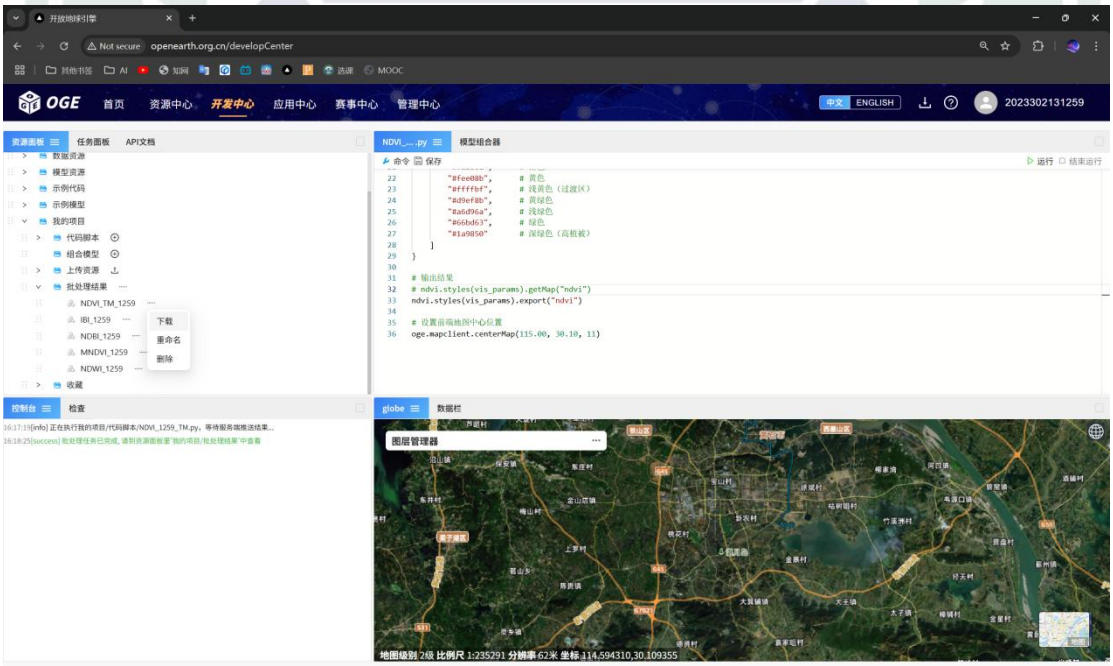


图3.3.1.3-4 在“批处理结果”中下载处理结果

然后等待浏览器下载完成即可（图 3.3.1.3-5）：

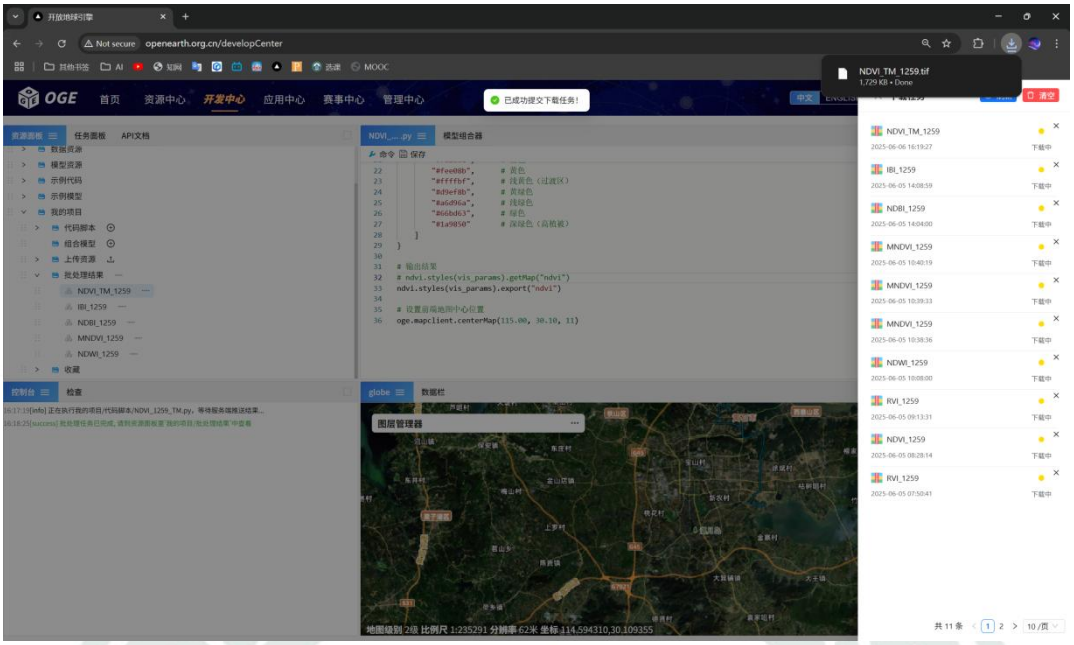


图3.3.1.3-5 等待浏览器下载完成

3.3.2 所用的具体代码与效果展示

注意，本节只对所用的具体代码与效果进行展示，**对所得结果的分析请参见 3.4 节的相关内容**。

3.3.2.1 植被指数 RVI（调用高分一号影像）

提取植被指数 RVI 的对象并不是平台提供的示例影像，而是调用的平台中的高分一号影像。高分一号影像的相关波段信息如图 3.3.2.1-1 所示：

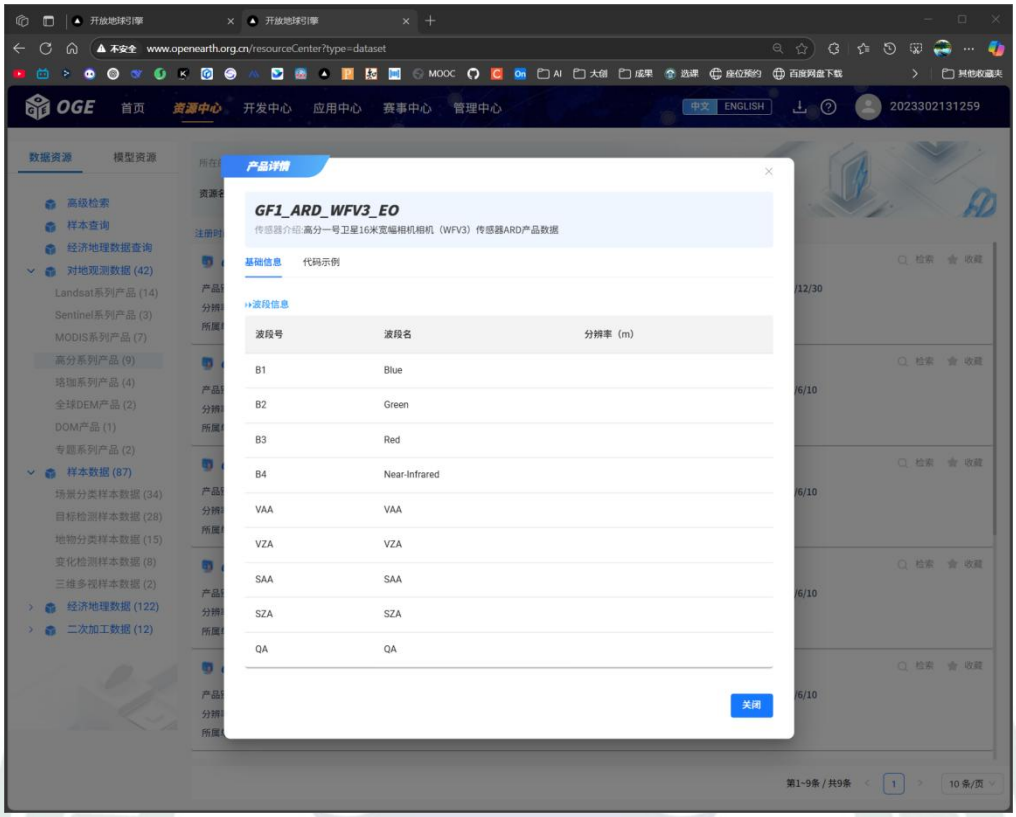


图3.3.2.1-1 高分一号影像的相关波段信息

提取植被指数 RVI 的相关代码如下所示：

```
import oge
# 初始化
oge.initialize()
service = oge.Service()
# 读取数据
gf1 =
service.getCoverage(coverageID="GF1_WFV3_E103.5_N35.6_20230709_L1A0007386026",
productID="GF1_ARD_WFV3_EO")
# 高分一号数据
# Coverage.selectBands（在文档的2.3.3中，意思是从一个Coverage中选择波段）
b3 = service.getProcess("Coverage.selectBands").execute(gf1, ["B3"]) # R
b4 = service.getProcess("Coverage.selectBands").execute(gf1, ["B4"]) # IR
# Coverage.toFloat 将波段数据转换为32Bit浮点型，使得结果也是浮点型
b3 = service.getProcess("Coverage.toFloat").execute(b3) # R
b4 = service.getProcess("Coverage.toFloat").execute(b4) # IR
# 具体哪一个图层是什么波段也可以在资源中心中点击影像信息看到
# 调用函数处理数据：RVI = IR / R
rvi = service.getProcess("Coverage.divide").execute(b4,b3)
# 波段号 波段名
# B1 Blue
# B2 Green
# B3 Red
```

```
# B4 Near-Infrared
# VAA VAA
# VZA VZA
# SAA SAA
# SZA SZA
# QA QA
# 设置渲染模式
vis_params = {
    "palette": [        # 颜色映射方案
        "#fdae61",      # 橙色
        "#fee08b",      # 黄色（低植被/裸地）
        "#ffffbf",      # 浅黄色（过渡区）
        "#d9ef8b",      # 黄绿色
        "#a6d96a",      # 浅绿色
        "#66bd63",      # 绿色
        "#1a9850"       # 深绿色（高植被）
    ]
}
# 输出结果
# rvi.styles(vis_params).getMap("rvi") # 在网页上显示
rvi.styles(vis_params).export("rvi") # 导出处理结果，注意设置分辨率的时候最小到40m
# 设置前端地图中的位置
oge.mapclient.centerMap(103.57, 35.58, 11)
# 三个参数分别是地图显示的中央精度、中央纬度和地图层级
```

提取的植被指数 RVI 的结果如图 3.2.2.1-2 所示：

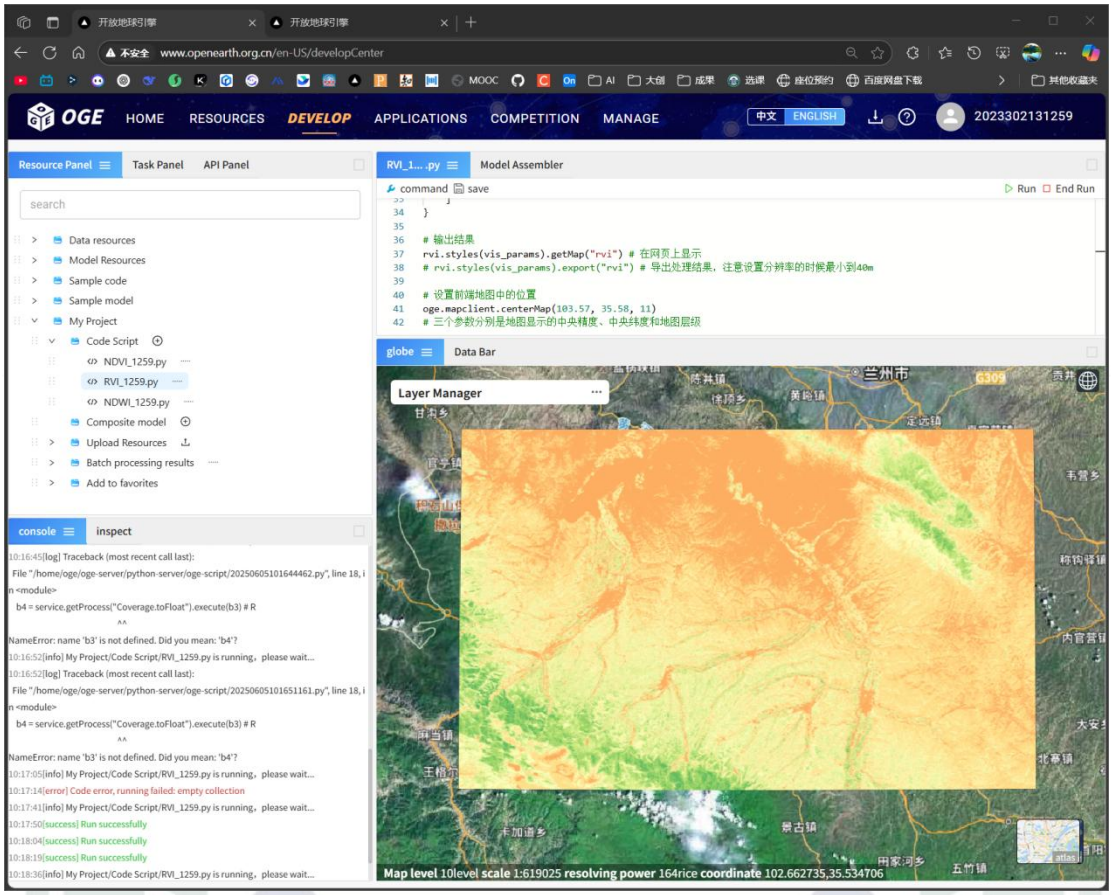


图3.2.1.1-2 对高分一号影像提取的植被指数RVI的结果

3.2.2.2 植被指数 NDVI（调用高分一号影像）

提取植被指数 NDVI 的对象并不是平台提供的示例影像，而是调用的平台中的高分一号影像。提取植被指数 NDVI 的相关代码如下所示：

```
import oge
oge.initialize()
service = oge.Service.initialize()
# 读取数据
gf1 =
service.getCoverage(coverageID="GF1_WFV3_E103.5_N35.6_20230709_L1A0007386026",
productID="GF1_ARD_WFV3_EO")
gf1_d = service.getProcess("Coverage.toFloat").execute(gf1)
# 调用函数处理数据
ndvi = service.getProcess("Coverage.normalizedDifference").execute(gf1_d, ["B4",
"B3"])
# NDVI = (IR - R) / (IR + R)
# 设置渲染模式
vis_params = {
    "min": -1,          # 最小值设为-1（典型NDVI范围下限）
```

```
    "max": 1,           # 最大值设为1（典型NDVI范围上限）
    "palette": [        # 颜色映射方案
        "#d73027",      # 红色（低植被/裸地）
        "#f46d43",      # 橙红色
        "#fdae61",      # 橙色
        "#fee08b",      # 黄色
        "#ffffbf",      # 浅黄色（过渡区）
        "#d9ef8b",      # 黄绿色
        "#a6d96a",      # 浅绿色
        "#66bd63",      # 绿色
        "#1a9850"       # 深绿色（高植被）
    ]
}
# 输出结果
# dvi.styles(vis_params).getMap("ndvi") # 在网页上显示
ndvi.styles(vis_params).export("ndvi") # 导出TIF
# 设置前端地图中心位置
oge.mapclient.centerMap(103.57, 35.58, 11)
```

提取的植被指数 NDVI 的结果如图 3.2.2.2-1 所示：

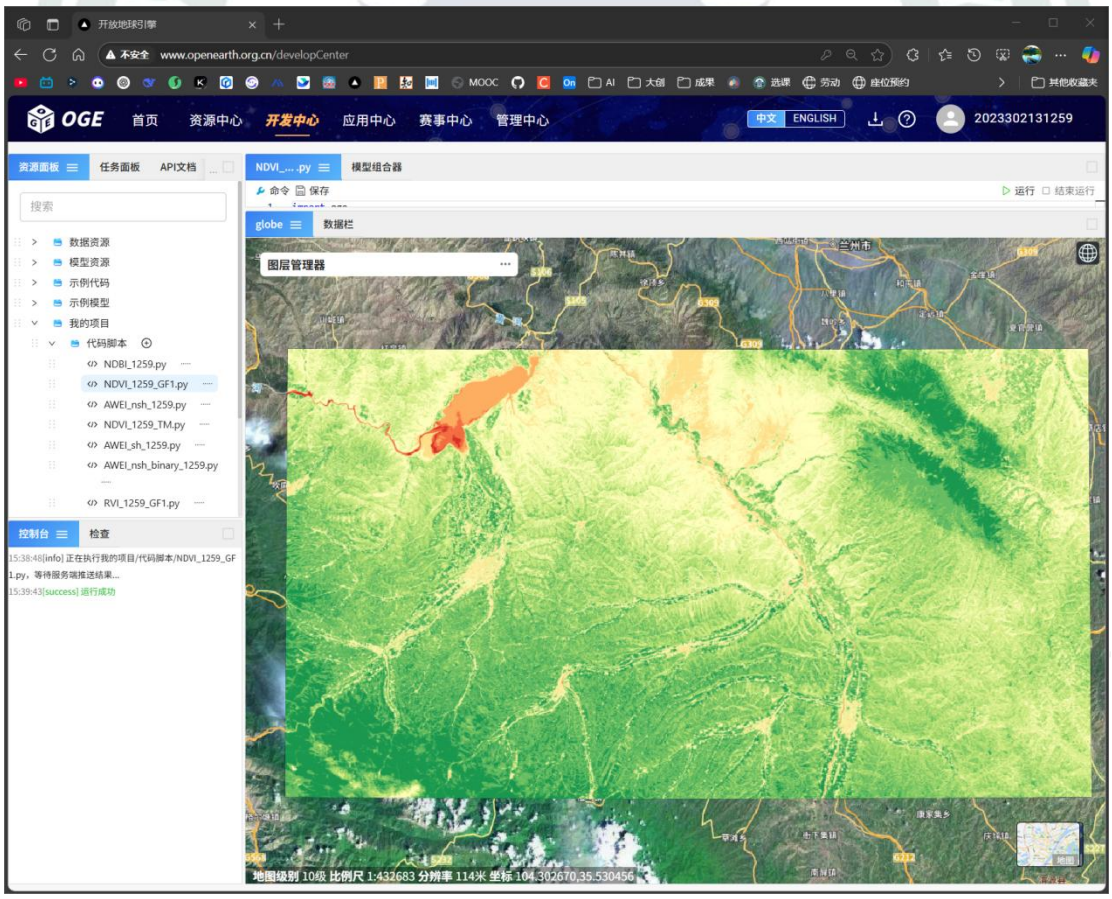


图3.2.2.2-1 对高分一号影像提取的植被指数NDVI的结果

3.2.2.3 植被指数 NDVI（使用自己上传的黄石地区的 TM 影像）

我又换了一张图，**这次提取植被指数 NDVI 的对象并也不是平台提供的示例影像，而是使用的自己上传的黄石地区的 TM 影像。**这张影像需要对下发文件中的各个波段的 TIF 文件应用 2.3.3 节所述的多波段合成的操作方法合成一张多波段的 tm_stack_res.tif 影像。

关于 TM（Thematic Mapper，专题制图仪）所得影像的波段信息，可以参考 Landsat-5 的影像（图 3.2.2.3-1）：

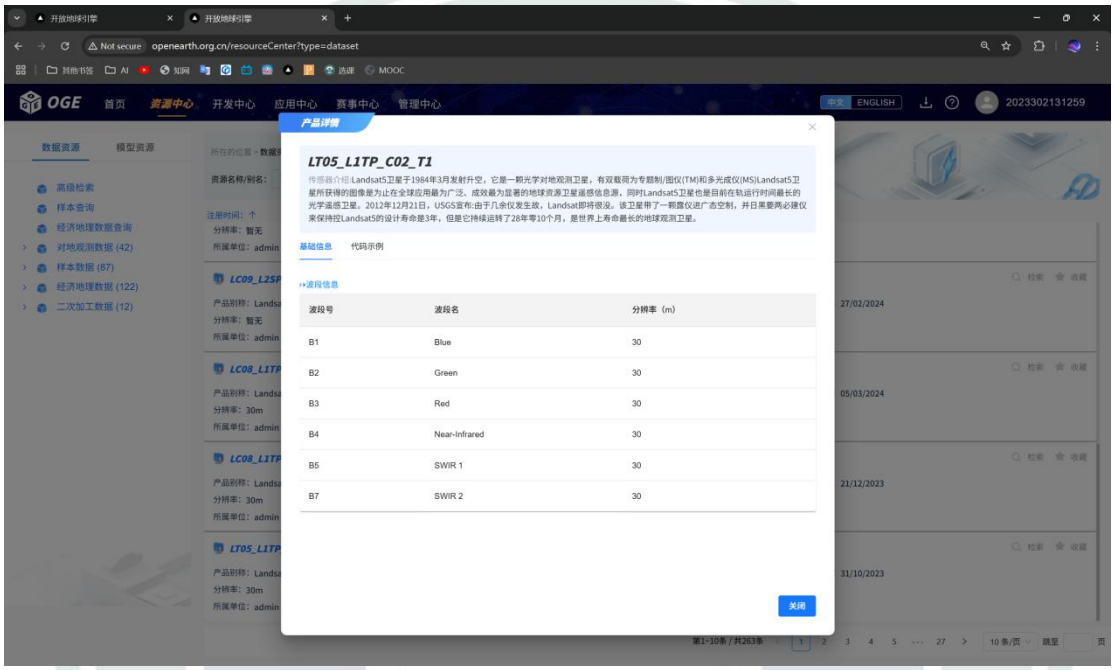


图3.2.2.3-1 TM（Thematic Mapper，专题制图仪）所得影像的波段信息

此外，关于黄石地区的地理坐标，可以通过 OGE 开发中心的“检查”操作从下方的地图可视化区域中选出并读取其经纬度信息（图 3.2.2.3-2）：

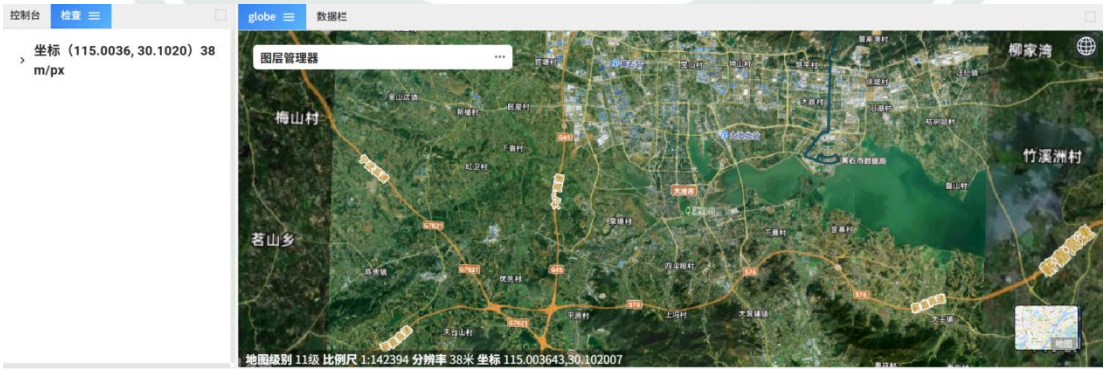


图3.2.2.3-2 使用“检查”操作读取经纬度信息

提取植被指数 NDVI 的相关代码如下所示：

```
import oge
oge.initialize()
service = oge.Service.initialize()
```

```
    # 读取数据
    tm1 = service.getCoverage(coverageID="myData/tm_stack_res.tif")
    tm1_d = service.getProcess("Coverage.toFloat").execute(tm1)
    # 调用函数处理数据
    ndvi = service.getProcess("Coverage.normalizedDifference").execute(tm1_d,
["B4", "B3"])
    # NDVI = (IR - R) / (IR + R)
    # 设置渲染模式
    vis_params = {
        "min": -1,          # 最小值设为-1（典型NDVI范围下限）
        "max": 1,           # 最大值设为1（典型NDVI范围上限）
        "palette": [        # 颜色映射方案
            "#d73027",      # 红色（低植被/裸地）
            "#f46d43",      # 橙红色
            "#fdae61",      # 橙色
            "#fee08b",      # 黄色
            "#ffffbf",      # 浅黄色（过渡区）
            "#d9ef8b",      # 黄绿色
            "#a6d96a",      # 浅绿色
            "#66bd63",      # 绿色
            "#1a9850"       # 深绿色（高植被）
        ]
    }
    # 输出结果
    ndvi.styles(vis_params).getMap("ndvi")
    # ndvi.styles(vis_params).export("ndvi")
    # 设置前端地图中心位置
    oge.mapclient.centerMap(115.00, 30.10, 11)
```

提取的植被指数 NDVI 的结果如图 3.2.2.3-3 所示：

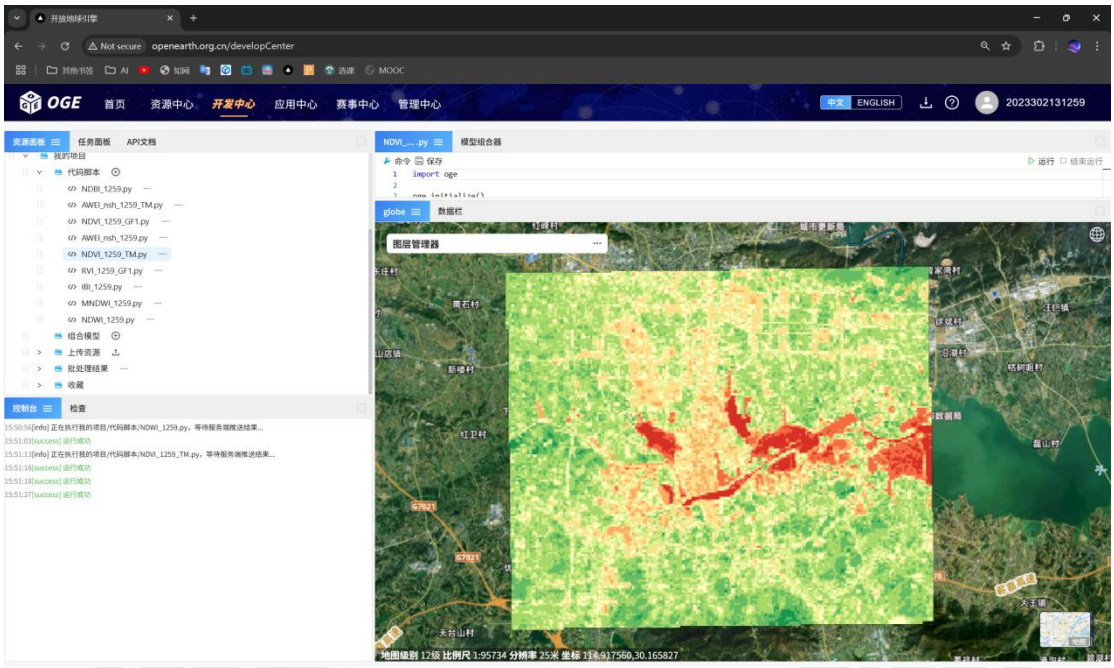


图3.2.2.3-3 提取的植被指数NDVI的结果

此外，可以通过下方的“图层管理器”，展开后拖动透明度调整条，将所得结果图层设置成半透明图层，便于与原始遥感影像进行比对，以更加直观地验证地物指数的提取效果。如图 3.2.2.3-4 所示，所提取的植被指数基本与植被区域吻合，而城市市区与水体的植被指数明显较低。

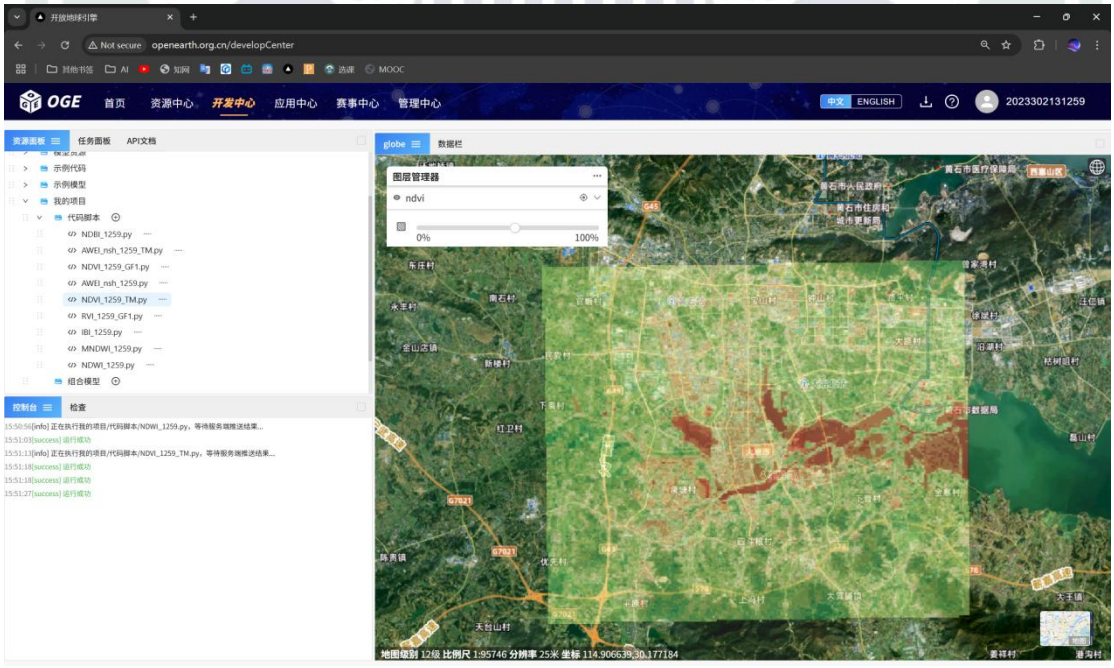


图3.2.2.3-4 将所得结果图层设置成半透明图层，便于与原始遥感影像进行比对

3.2.2.4 水体指数 NDWI（使用默认的 Landsat-8 影像）

系统默认提供的是 Landsat-8 的遥感影像，其波段信息如图 3.2.2.4-1 所示：

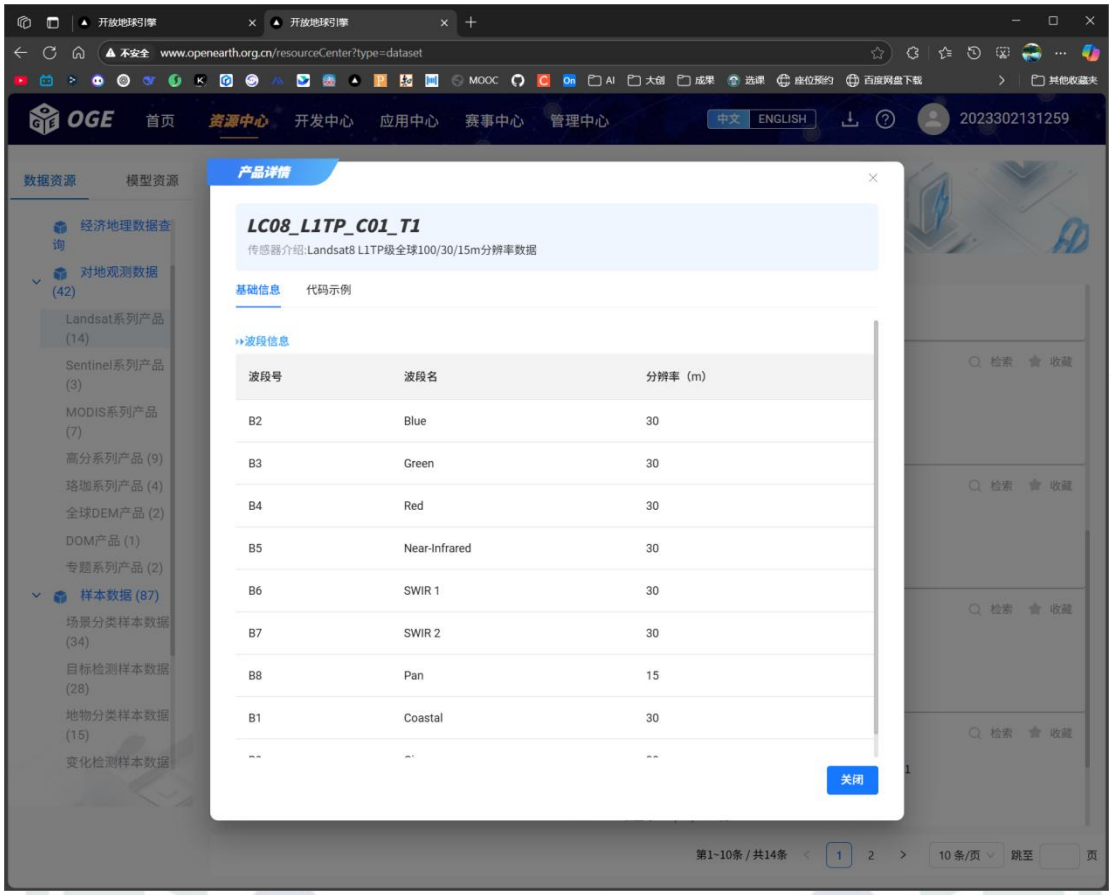


图3.2.2.4-1 Landsat-8遥感影像的波段信息

提取水体指数 NDWI 的相关代码如下所示：

```
import oge
oge.initialize()
service = oge.Service.initialize()
# 读取数据
ls8 = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
ls8_d = service.getProcess("Coverage.toFloat").execute(ls8)
# 调用函数处理数据
ndwi = service.getProcess("Coverage.normalizedDifference").execute(ls8_d, ["B3",
"B5"])
# Landsat-8 的波段对应：
# 波段号 波段名 分辨率（m）
# B2 Blue 30
# B3 Green 30
# B4 Red 30
# B5 Near-Infrared 30
# B6 SWIR 1 30
# B7 SWIR 2 30
# B8 Pan 15
# B1 Coastal 30
```

```
# B9 Cirrus 30
# NDWI = (G - IR) / (G + IR)
# 设置渲染模式
vis_params = {
    "palette": [
        "#D3D3D3",  # 浅灰（非水体）
        "#A9A9A9",  # 中灰
        "#778899",  # 灰蓝色（过渡）
        "#5F9EA0",  # 蓝灰色
        "#4682B4",  # 钢蓝
        "#1E90FF",  # 道奇蓝
        "#0000FF"   # 纯蓝（强水体信号）
    ]
}
# 输出结果
ndwi.styles(vis_params).getMap("ndwi")
# ndwi.styles(vis_params).export("ndwi")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28,30.57,10)
```

提取的水体指数 NDWI 的结果如图 3.2.2.4-2 所示：

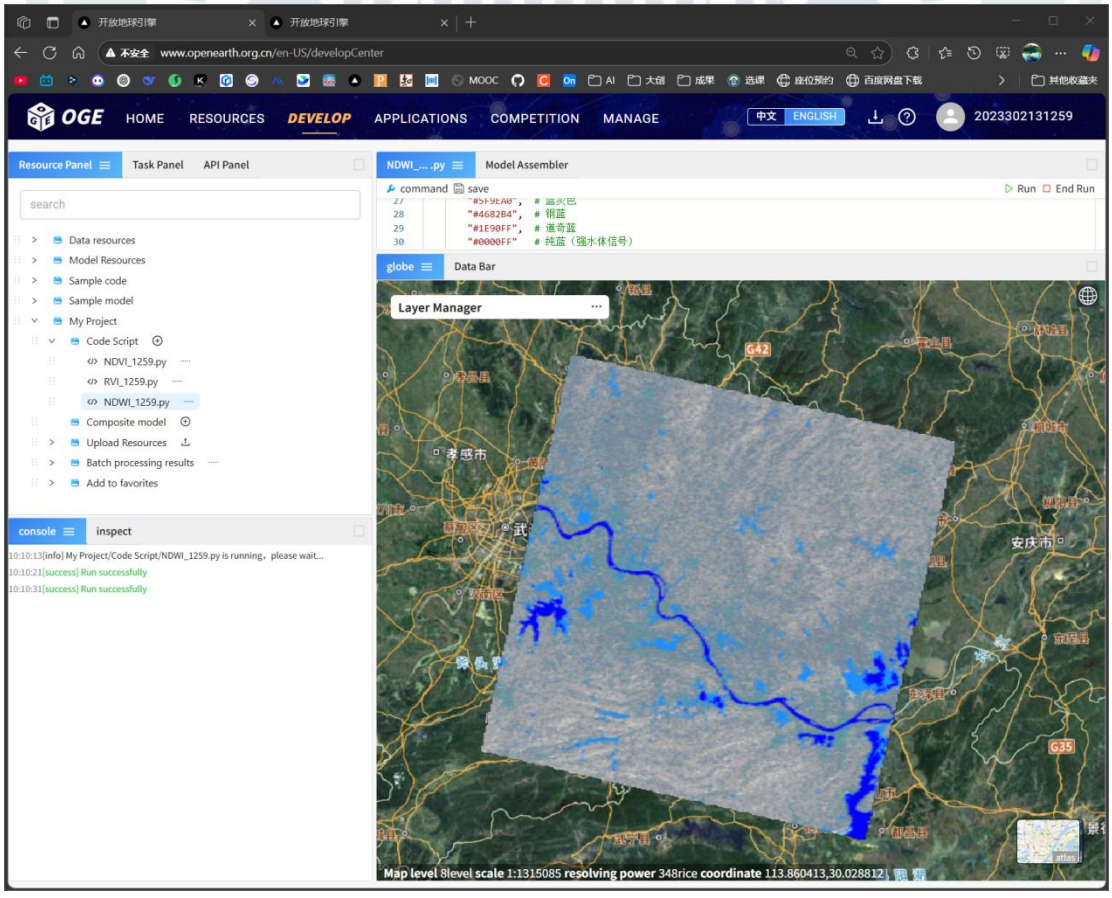


图3.2.2.4-2 提取的水体被指数NDWI的结果

3.2.2.5 水体指数 MNDWI（使用默认的 Landsat-8 影像）

提取水体指数 MNDWI 的相关代码如下所示：

```
import oge
oge.initialize()
service = oge.Service.initialize()
# 读取数据
ls8 = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
ls8_d = service.getProcess("Coverage.toFloat").execute(ls8)
# 调用函数处理数据
# 计算 MNDWI（使用 B3: Green 和 B6: SWIR1）
mndwi = service.getProcess("Coverage.normalizedDifference").execute(ls8_d, ["B3",
"B6"])
# Landsat-8 的波段对应:
# 波段号 波段名 分辨率（m）
# B2 Blue 30
# B3 Green 30
# B4 Red 30
# B5 Near-Infrared 30
# B6 SWIR 1 30
# B7 SWIR 2 30
# B8 Pan 15
# B1 Coastal 30
# B9 Cirrus 30
# NDWI = (G - SWIR1) / (G + SWIR1)
# 设置渲染模式
vis_params = {
    "palette": [
        "#D3D3D3",  # 浅灰（非水体）
        "#A9A9A9",  # 中灰
        "#778899",  # 灰蓝色（过渡）
        "#5F9EA0",  # 蓝灰色
        "#4682B4",  # 钢蓝
        "#1E90FF",  # 道奇蓝
        "#0000FF"   # 纯蓝（强水体信号）
    ]
}
# 输出结果
mndwi.styles(vis_params).getMap("mndwi")
# mndwi.styles(vis_params).export("mndwi")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28,30.57,10)
```

提取的水体被指数 MNDWI 的结果如图 3.2.2.5-1 所示：

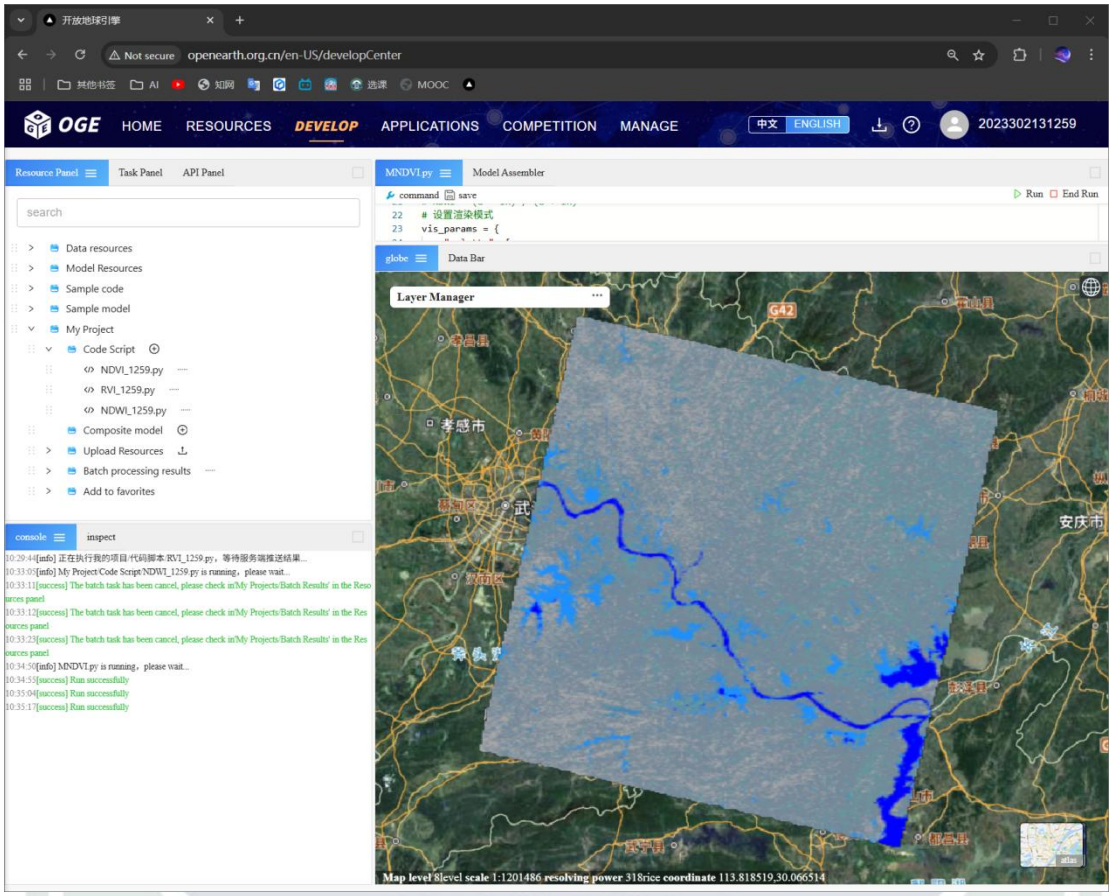


图3.2.2.5-1 提取的水体被指数MNDWI的结果

3.2.2.6 水体指数 $AWEI_{nsh}$（使用默认的 Landsat-8 影像）

提取水体指数$AWEI_{nsh}$（nsh 表示无阴影优化）的相关代码如下所示：

```
   import oge
# 初始化
oge.initialize()
service = oge.Service()
# 读取数据
lc08 = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
lc08_d = service.getProcess("Coverage.toFloat").execute(lc08)
green=service.getProcess("Coverage.selectBands").execute(lc08,["B2"])
nir=service.getProcess("Coverage.selectBands").execute(lc08,["B4"])
swir1=service.getProcess("Coverage.selectBands").execute(lc08,["B5"])
swir2=service.getProcess("Coverage.selectBands").execute(lc08,["B7"])
temp1=service.getProcess("Coverage.subtract").execute(green,swir1)
temp2=service.getProcess("Coverage.multiplyNum").execute(temp1,4)
temp3=service.getProcess("Coverage.multiplyNum").execute(nir,0.25)
```

```
temp4=service.getProcess("Coverage.multiplyNum").execute(swir2,2.75)
temp5=service.getProcess("Coverage.add").execute(temp3,temp4)
AWEI_nsh=service.getProcess("Coverage.subtract").execute(temp2,temp5)
# 设置渲染模式
vis_params = {
    "palette": [
        "#D3D3D3",  # 浅灰（非水体）
        "#A9A9A9",  # 中灰
        "#778899",  # 灰蓝色（过渡）
        "#5F9EA0",  # 蓝灰色
        "#4682B4",  # 钢蓝
        "#1E90FF",  # 道奇蓝
        "#0000FF"   # 纯蓝（强水体信号）
    ]
}
# 输出结果
AWEI_nsh.styles(vis_params).getMap("AWEI_nsh")
# AWEI_nsh.styles(vis_params).export("AWEI_nsh")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28, 30.57, 10)
```

提取的水体指数$AWEI_{nsh}$的结果如图 3.2.2.6-1 所示：

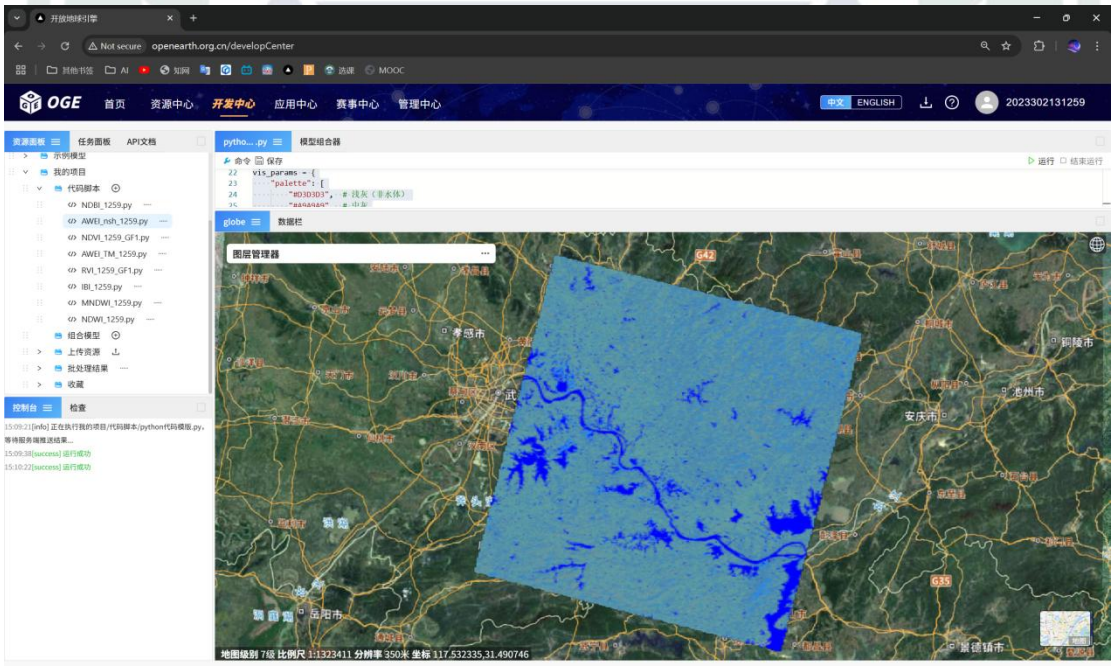


图3. 2. 2. 6-1 提取的水体指数$AWEI_{nsh}$的结果

3.2.2.7 水体指数 $AWEI_{sh}$（使用默认的 Landsat-8 影像）

提取水体指数$AWEI_{sh}$（sh 表示有阴影优化）的相关代码如下所示：

```
import oge
```

```
# 初始化
oge.initialize()
service = oge.Service()
# 读取数据
lc08 = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
lc08_d = service.getProcess("Coverage.toFloat").execute(lc08)
# 选择需要的波段
blue = service.getProcess("Coverage.selectBands").execute(lc08, ["B1"])
green = service.getProcess("Coverage.selectBands").execute(lc08, ["B2"])
red = service.getProcess("Coverage.selectBands").execute(lc08, ["B3"])
nir = service.getProcess("Coverage.selectBands").execute(lc08, ["B4"])
swir1 = service.getProcess("Coverage.selectBands").execute(lc08, ["B5"])
swir2 = service.getProcess("Coverage.selectBands").execute(lc08, ["B7"])
# 计算 AWEI_sh
# 公式: AWEI_sh = Blue + 2.5*Green - 1.5*(NIR + SWIR1) - 0.25*SWIR2
temp1 = service.getProcess("Coverage.multiplyNum").execute(green, 2.5)
temp2 = service.getProcess("Coverage.add").execute(blue, temp1)
temp3 = service.getProcess("Coverage.add").execute(nir, swir1)
temp4 = service.getProcess("Coverage.multiplyNum").execute(temp3, 1.5)
temp5 = service.getProcess("Coverage.subtract").execute(temp2, temp4)
temp6 = service.getProcess("Coverage.multiplyNum").execute(swir2, 0.25)
AWEI_sh = service.getProcess("Coverage.subtract").execute(temp5, temp6)
# 设置渲染模式 - 可以使用与 AWEI_nsh 相同的参数或调整
vis_params = {
    "palette": [
        "#D3D3D3",  # 浅灰（非水体）
        "#A9A9A9",  # 中灰
        "#778899",  # 灰蓝色（过渡）
        "#5F9EA0",  # 蓝灰色
        "#4682B4",  # 钢蓝
        "#1E90FF",  # 道奇蓝
        "#0000FF"   # 纯蓝（强水体信号）
    ]
}
# 输出结果
AWEI_sh.styles(vis_params).getMap("AWEI_sh")
# AWEI_sh.styles(vis_params).export("AWEI_sh")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28, 30.57, 10)
```

提取的水体指数$AWEI_{sh}$的结果如图 3.2.2.7-1 所示：

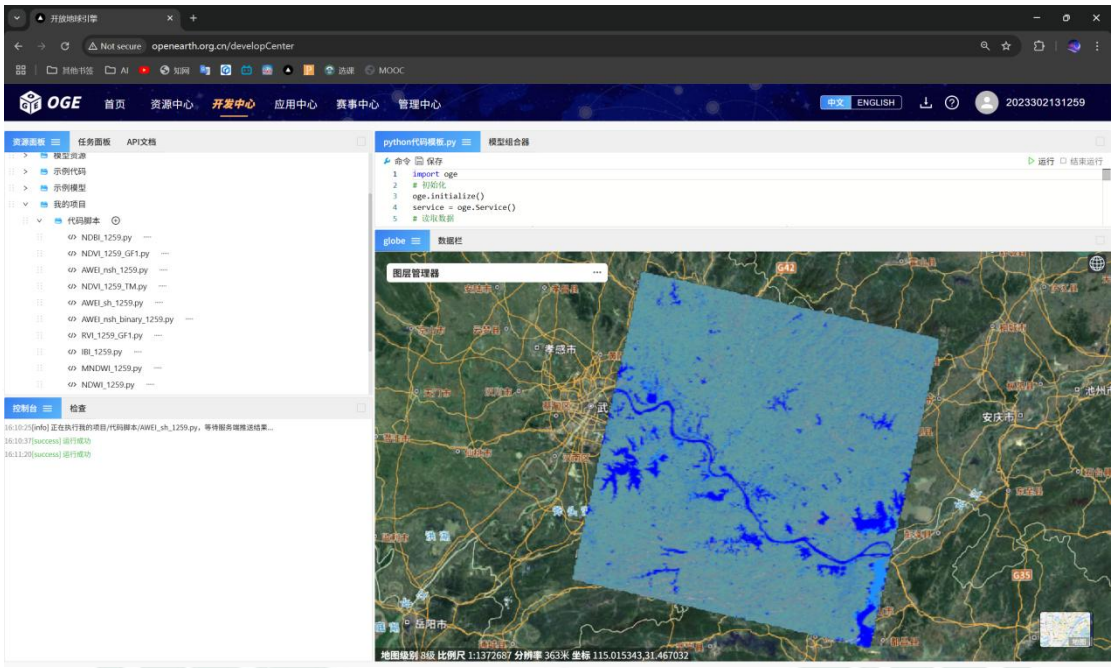


图3.2.2.7-1 提取的水体指数$AWEI_{sh}$的结果

3.2.2.8 水体指数 $AWEI_{nsh}$（使用默认的 Landsat-8 影像，二值化）

提取水体指数$AWEI_{nsh}$（nsh 表示无阴影优化）后对其进行**二值化**显示的相关代码如下所示：

```
import oge
# 初始化
oge.initialize()
service = oge.Service()
# 读取数据
lc08 = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
lc08_d = service.getProcess("Coverage.toFloat").execute(lc08)
green=service.getProcess("Coverage.selectBands").execute(lc08,["B2"])
nir=service.getProcess("Coverage.selectBands").execute(lc08,["B4"])
swir1=service.getProcess("Coverage.selectBands").execute(lc08,["B5"])
swir2=service.getProcess("Coverage.selectBands").execute(lc08,["B7"])
temp1=service.getProcess("Coverage.subtract").execute(green,swir1)
temp2=service.getProcess("Coverage.multiplyNum").execute(temp1,4)
temp3=service.getProcess("Coverage.multiplyNum").execute(nir,0.25)
temp4=service.getProcess("Coverage.multiplyNum").execute(swir2,2.75)
temp5=service.getProcess("Coverage.add").execute(temp3,temp4)
AWEI_nsh=service.getProcess("Coverage.subtract").execute(temp2,temp5)
AWEI_binary=service.getProcess("Coverage.binarization").execute(AWEI_nsh,-1000
0)
# 设置渲染模式
```

```
vis_params = {
    "palette": [
        "#D3D3D3",  # 浅灰（非水体）
#         "#A9A9A9",  # 中灰
#         "#778899",  # 灰蓝色（过渡）
#         "#5F9EA0",  # 蓝灰色
#         "#4682B4",  # 钢蓝
#         "#1E90FF",  # 道奇蓝
        "#0000FF"   # 纯蓝（强水体信号）
    ]
}
# 输出结果
# AWEI_nsh.styles(vis_params).getMap("AWEI_nsh")
# AWEI_nsh.styles(vis_params).export("AWEI_nsh")
AWEI_binary.styles(vis_params).getMap("AWEI_binary")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28, 30.57, 10)
```

提取水体指数$AWEI_{nsh}$（nsh 表示无阴影优化）后对其进行**二值化**显示的结果如图 3.2.2.8-2 所示：

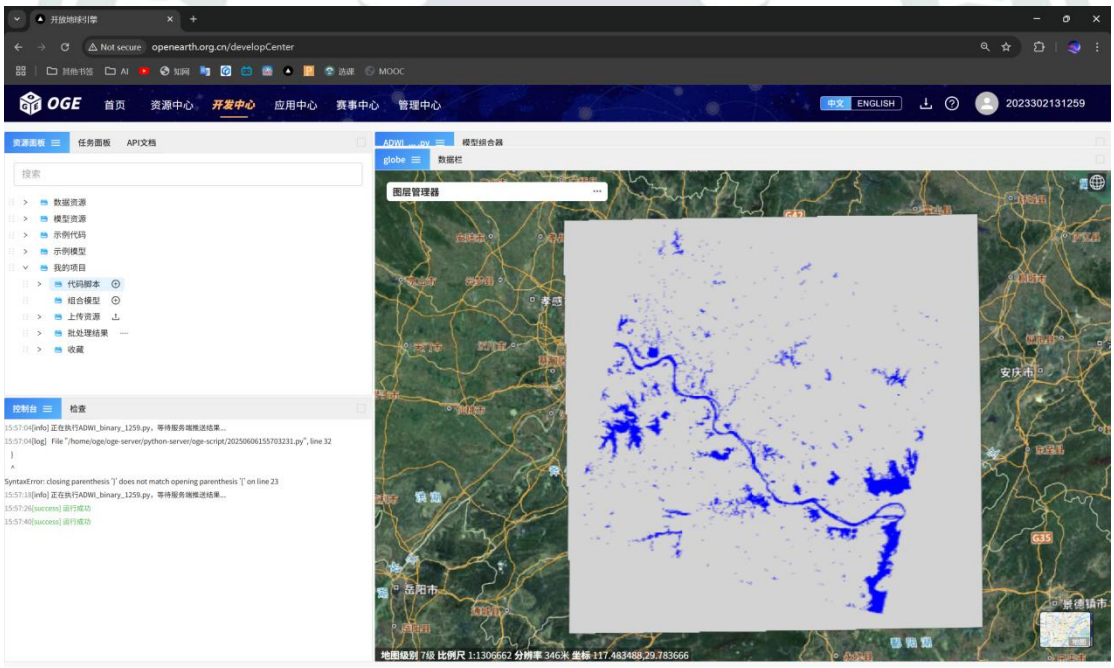


图3.2.2.8-2 提取水体指数$AWEI_{nsh}$（nsh表示无阴影优化）后对其进行二值化显示的结果

调整其透明度之后的效果图如图 3.2.2.8-3 所示：

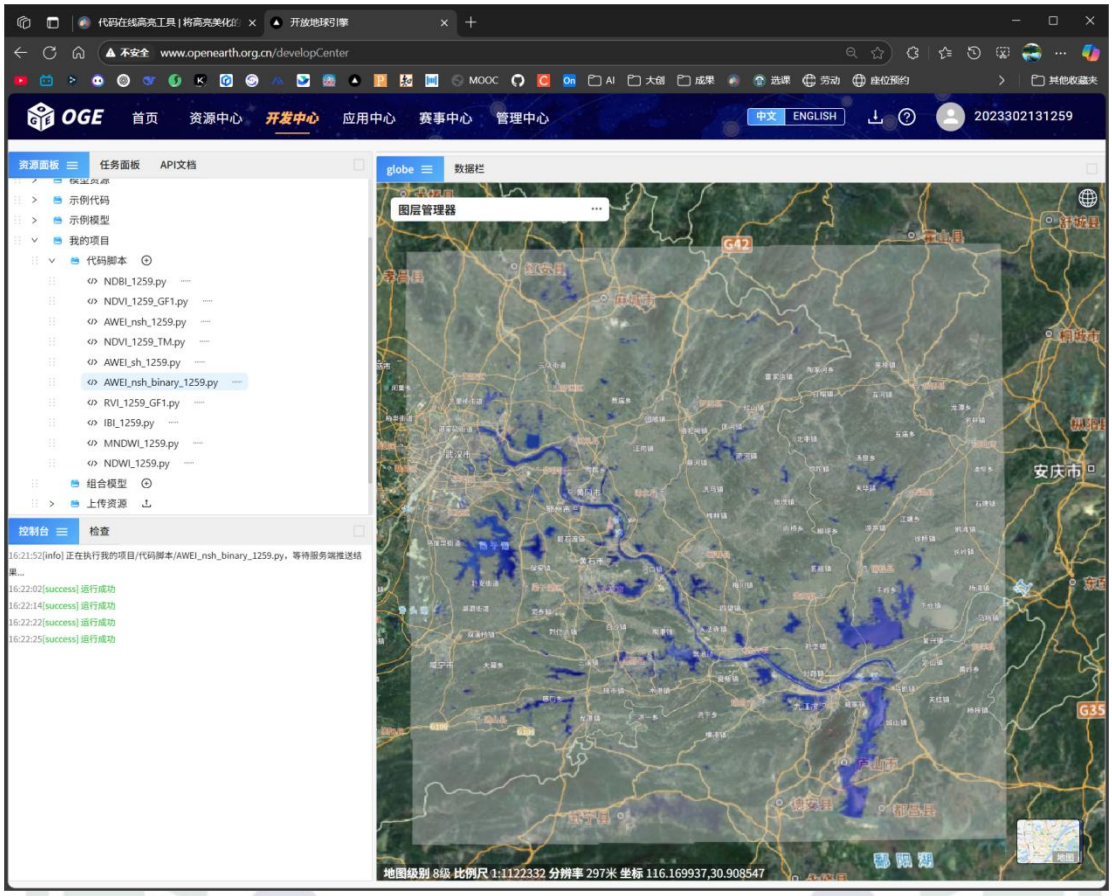


图3.2.2.8-3 提取水体指数$AWEI_{nsh}$（nsh表示无阴影优化）后对其进行二值化显示的半透明图层结果

3.2.2.9 建筑指数 NDBI（使用默认的 Landsat-8 影像）

提取建筑指数 NDBI 的相关代码如下所示：

```
# 初始化
import oge
oge.initialize()
service = oge.Service.initialize()
# 读取数据
img = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
img_d = service.getProcess("Coverage.toFloat").execute(img)
# 调用函数处理数据
green = service.getProcess("Coverage.selectBands").execute(img_d,["B3"])
red = service.getProcess("Coverage.selectBands").execute(img_d,["B4"])
nir = service.getProcess("Coverage.selectBands").execute(img_d,["B5"])
mnir = service.getProcess("Coverage.selectBands").execute(img_d,["B6"])
# 计算 SAVI 指数,L 取值为 0.1
SAVI_temp1 = service.getProcess("Coverage.subtract").execute(nir, red)
```

```
SAVI_temp2 = service.getProcess("Coverage.multiplyNum").execute(SAVI_temp1, 1.1)
SAVI_temp3 = service.getProcess("Coverage.add").execute(nir, red)
SAVI_temp4 = service.getProcess("Coverage.addNum").execute(SAVI_temp3, 0.1)
SAVI = service.getProcess("Coverage.divide").execute(SAVI_temp2, SAVI_temp4)
#计算NDBI 指数
NDBI = service.getProcess("Coverage.normalizedDifference").execute(img_d, ["B6",
"B5"])
# Landsat-8 的波段对应：
# 波段号 波段名 分辨率（m）
# B2 Blue 30
# B3 Green 30
# B4 Red 30
# B5 Near-Infrared 30
# B6 SWIR 1 30
# B7 SWIR 2 30
# B8 Pan 15
# B1 Coastal 30
# B9 Cirrus 30
# 设置渲染模式
vis_params = {
}
# 输出结果
# NDBI.styles(vis_params).getMap("ndbi")
NDBI.styles(vis_params).export("ndbi")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28,30.57,10)
```

提取的建筑指数 NDBI 的结果如图 3.2.2.9-1 所示：

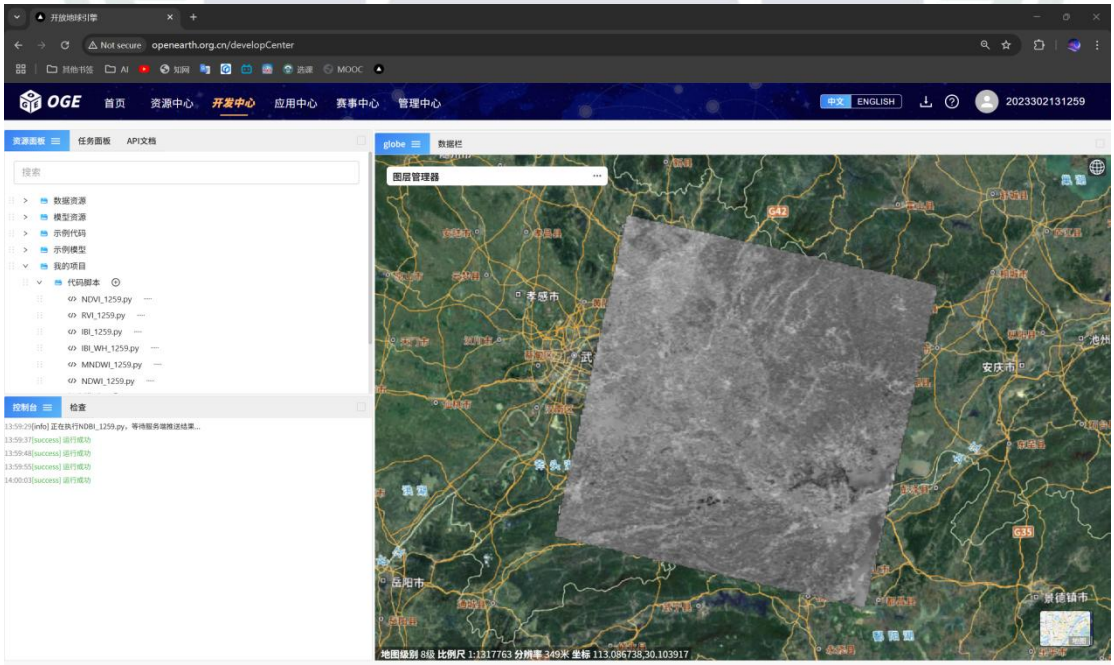


图3.2.2.9-1 提取的建筑指数NDBI的结果

3.2.2.10 建筑指数 IBI（使用默认的 Landsat-8 影像）

提取建筑指数 IBI 的相关代码如下所示：

```
# 初始化
import oge
oge.initialize()
service = oge.Service.initialize()
# 读取数据
img = service.getCoverage(coverageID="LC81220392015275LGN00",
productID="LC08_L1T")
img_d = service.getProcess("Coverage.toFloat").execute(img)
# 调用函数处理数据
green = service.getProcess("Coverage.selectBands").execute(img_d,["B3"])
red = service.getProcess("Coverage.selectBands").execute(img_d,["B4"])
nir = service.getProcess("Coverage.selectBands").execute(img_d,["B5"])
mnir = service.getProcess("Coverage.selectBands").execute(img_d,["B6"])
# 计算SAVI 指数,L 取值为0.1
SAVI_temp1 = service.getProcess("Coverage.subtract").execute(nir, red)
SAVI_temp2 = service.getProcess("Coverage.multiplyNum").execute(SAVI_temp1, 1.1)
SAVI_temp3 = service.getProcess("Coverage.add").execute(nir, red)
SAVI_temp4 = service.getProcess("Coverage.addNum").execute(SAVI_temp3, 0.1)
SAVI = service.getProcess("Coverage.divide").execute(SAVI_temp2, SAVI_temp4)
#计算NDBI 指数
NDBI = service.getProcess("Coverage.normalizedDifference").execute(img_d, ["B6",
"B5"])
#计算MNDWI 指数
MNDWI = service.getProcess("Coverage.normalizedDifference").execute(img_d, ["B3",
"B6"])
#计算IBI 指数
IBI_temp1 = service.getProcess("Coverage.add").execute(SAVI, MNDWI)
IBI_temp2 = service.getProcess("Coverage.divideNum").execute(IBI_temp1, 2)
IBI_temp3 = service.getProcess("Coverage.subtract").execute(NDBI, IBI_temp2)
IBI_temp4 = service.getProcess("Coverage.add").execute(NDBI, IBI_temp2)
IBI = service.getProcess("Coverage.divide").execute(IBI_temp3, IBI_temp4)
IBI = service.getProcess("Coverage.toInt8").execute(IBI)
IBI = service.getProcess("Coverage.binarization").execute(IBI,0)
# Landsat-8 的波段对应：
# 波段号 波段名 分辨率（m）
# B2 Blue 30
# B3 Green 30
# B4 Red 30
# B5 Near-Infrared 30
# B6 SWIR 1 30
```

```
# B7 SWIR 2 30
# B8 Pan 15
# B1 Coastal 30
# B9 Cirrus 30
# 设置渲染模式
vis_params = {}
# 输出结果
# IBI.styles(vis_params).getMap("ibi")
IBI.styles(vis_params).export("ibi")
# 设置前端地图中心位置
oge.mapclient.centerMap(114.28,30.57,10)
```

提取的建筑指数 IBI 的结果如图 3.2.2.10-1 所示：

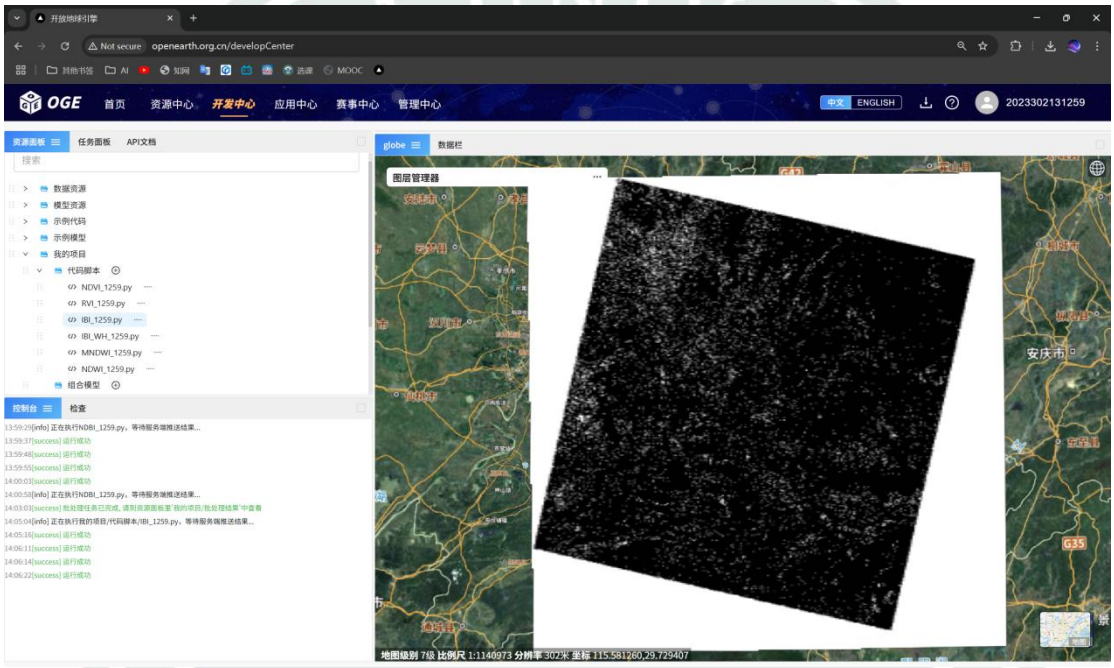


图3.2.2.10-1 提取建筑指数IBI的结果

3.4 实习结果与分析

3.4.1 OGE 植被检测结果截图

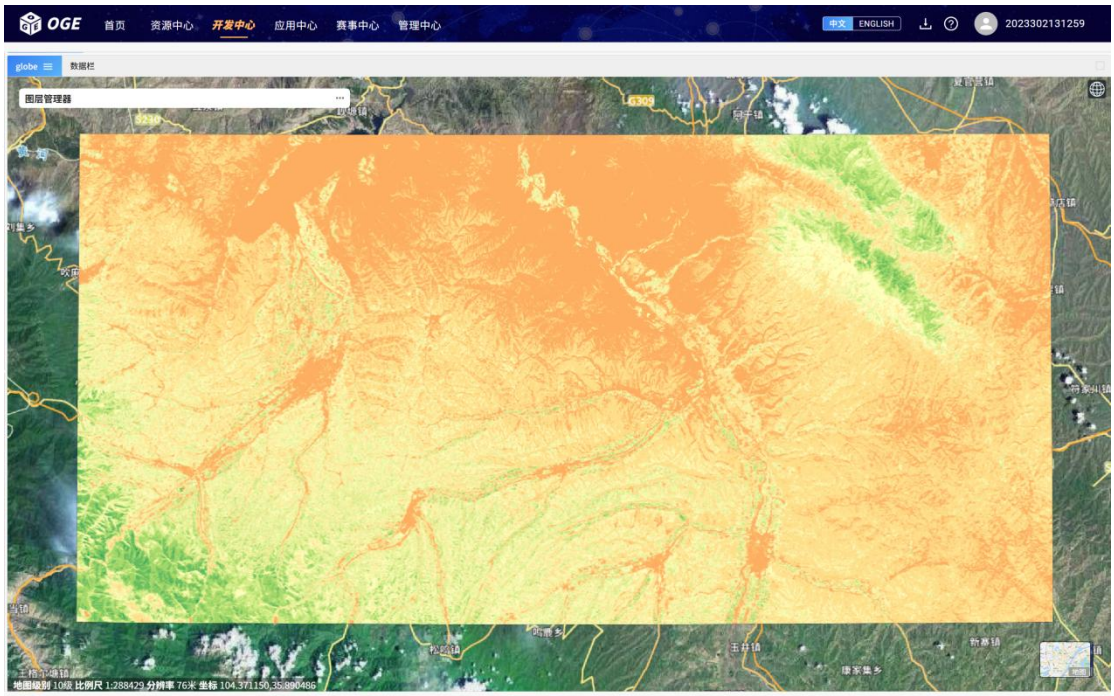


图3. 4. 1-1 对非默认影像的高分一号影像提取的植被指数RVI的结果

结果分析：从图 3.4.1-1 对高分一号影像提取的植被指数 RVI 的结果来看，植被区域的 RVI 值明显高于非植被区域，能够较好地反映出植被的分布情况。植被茂盛的区域 RVI 值较大，呈现出较高的植被覆盖度，这与植被指数 RVI 的原理相符，即植被在近红外波段反射率高，红光波段反射率低，RVI 通过比值运算放大了这种差异，从而有效突出了植被信息。

然而，RVI 在植被覆盖度较低的区域（一般为植被覆盖度小于 50%）敏感性显著降低，对植被的检测效果不够理想，因此在图 3.4.1-2 所示的 NDVI 的影像中许多显示为绿色的区域（大概率是植被）在图 3.4.1-1 中显示为黄色（不太像是植被）。此外，RVI 易受大气和土壤背景影响，大气效应会降低其对植被检测的灵敏度，这可能导致在不同环境条件下植被检测的准确性存在差异。

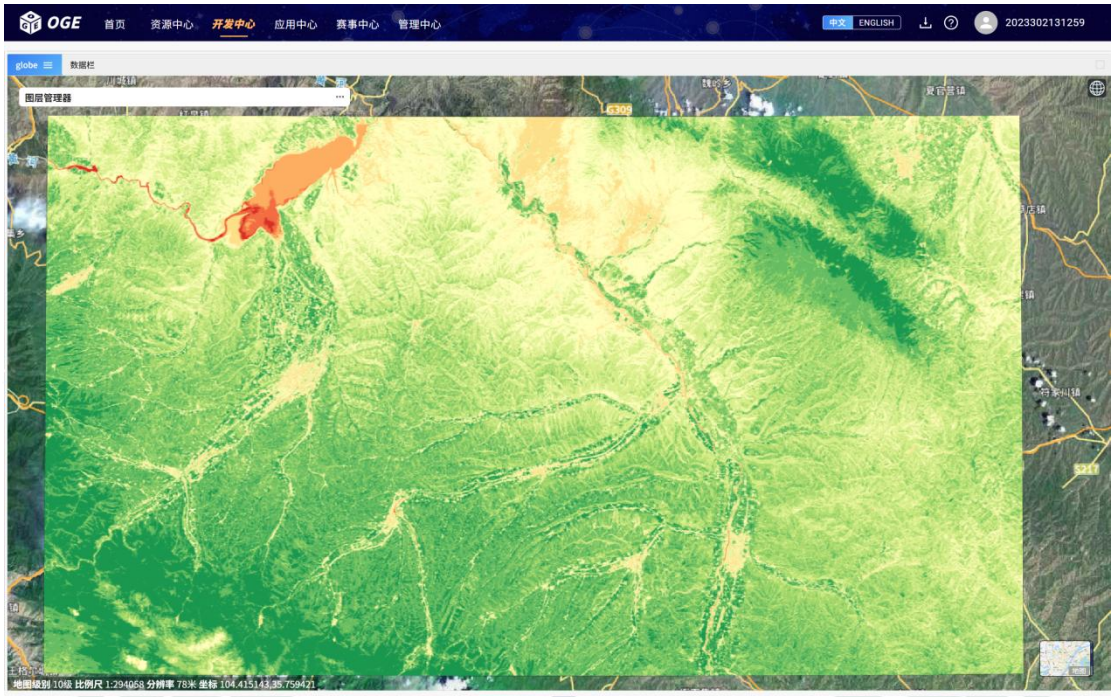


图3.4.1-2 对非默认影像的高分一号影像提取的植被指数NDVI的结果

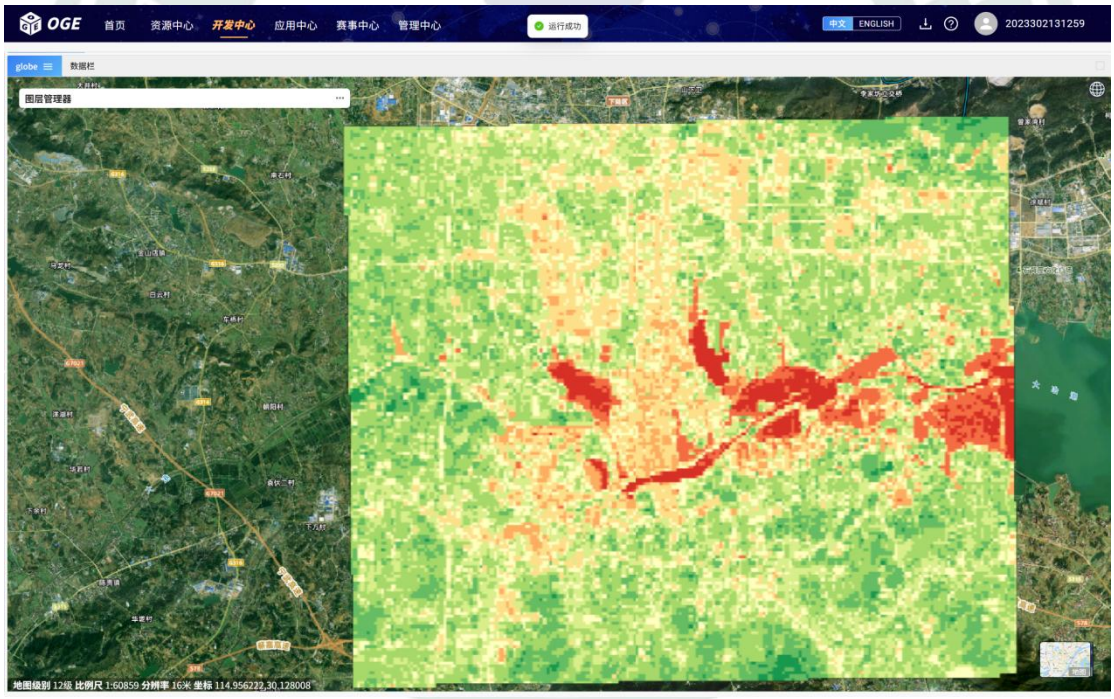


图3.4.1-3 对自己上传的黄石地区的TM影像提取的植被指数NDVI的结果

结果分析：图 3.4.1-2 和图 3.4.1-3 分别展示了对高分一号影像和黄石地区的 TM 影像提取的植被指数 NDVI 的结果。从结果可以看出，NDVI 能够较好地区分植被与非植被区域，植被覆盖度较高的区域 NDVI 值较大，呈现出深绿色，而裸土或建筑用地等非植被区域 NDVI 值接近 0 或小于 0，水体的 NDVI 值通常小于 0，这种明显的差异使得植被检测更为准确。NDVI 通过归一化处理克服了 RVI 的不足，不依赖绝对反射率，具有使用广泛的特点。

然而尽管 NDVI 具有诸多优点，但 NDVI 也容易在植被稀疏的区域受到裸土的反射、大气、土壤等其他因素的影响。因此在实际应用中，可以考虑结合 EVI 或 SAVI 等其他植被指数，以降低这些因素对植被检测的影响。

3.4.2 OGE 水体检测结果截图

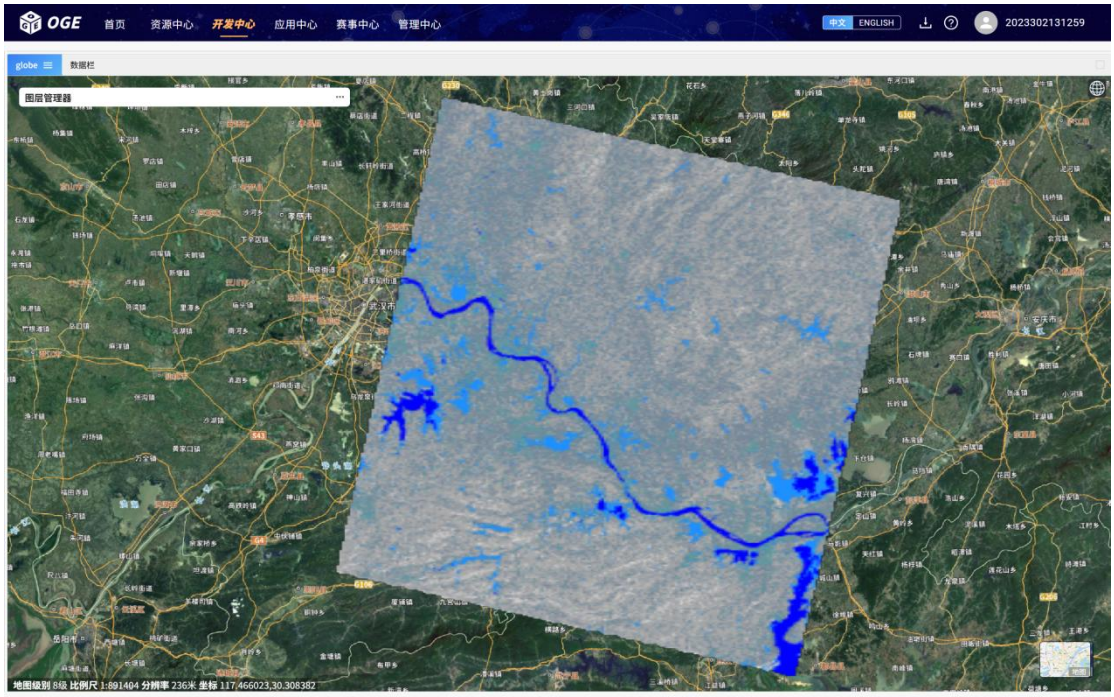


图3. 4. 2-1 提取的水体指数NDWI的结果

结果分析：从图 3.4.2-1 提取的水体指数 NDWI 的结果来看，水体区域的 NDWI 值较高，呈现出较强的水体信号，能够较好地识别出水体。NDWI 利用绿光和近红外波段的反射差异来识别水体，计算简便，适用于快速提取水体信息。然而，NDWI 易受建筑与阴影的影响，常常将其误认为是水体，导致水体检测的准确性降低，因此在图 3.4.2-1 中可以看出，武汉市市区内或江汉平原周围的丘陵地区，建筑物与丘陵的阴影干扰了 NDWI 的计算，从而产生误判为水体的问题。

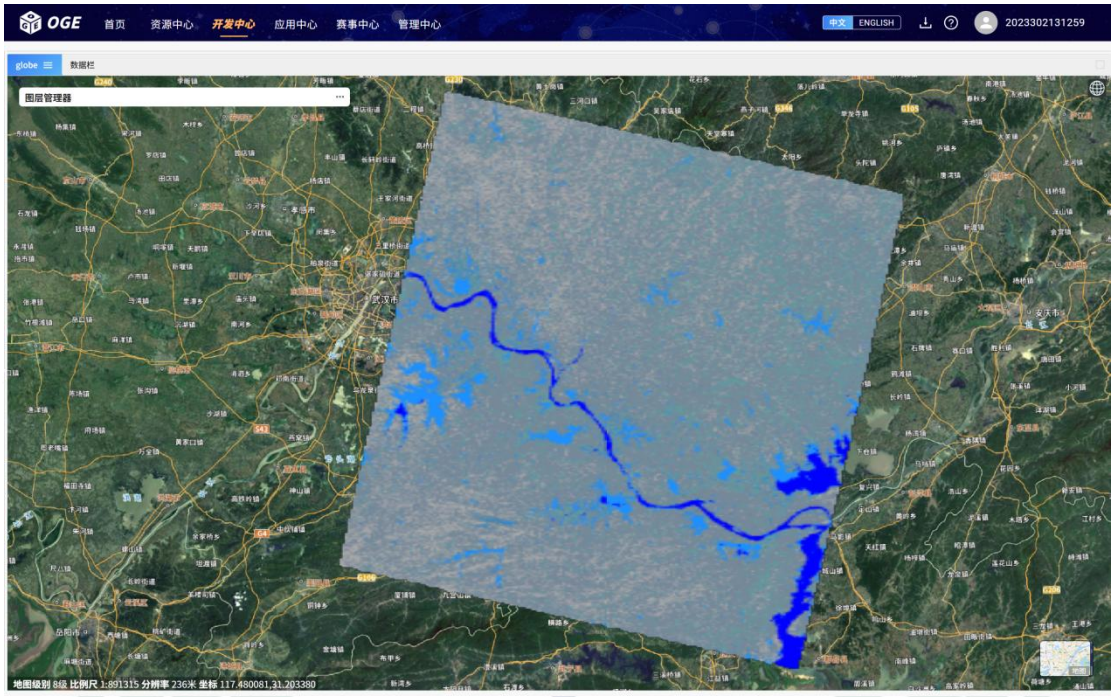


图3.4.2-2 提取的水体指数MNDWI的结果

结果分析：图 3.4.2-2 展示了提取的水体指数 MNDWI 的结果。MNDWI 用短波红外波段替代近红外波段，提高了对建筑阴影的区分能力，能够更准确地识别水体。与 NDWI 相比，MNDWI 在抑制建筑的干扰方面表现更好，在武汉市市区的误判为水体的情况减少，对于一些湖泊的判别能力也更高，从而提高了水体检测的精度。

然而，尽管 MNDWI 在一定程度上克服了 NDWI 的不足，但其仍受到其他因素的影响，如植被、土壤等的反射特性会对 MNDWI 的计算产生一定的干扰，从而影响水体检测的准确性。

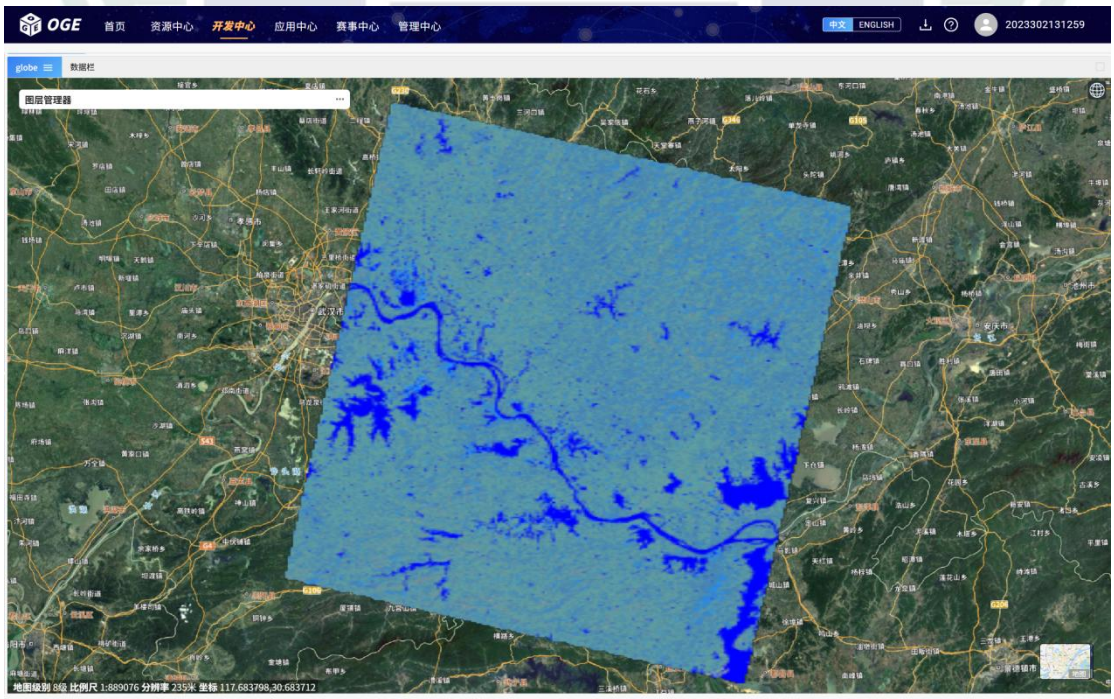


图3.4.2-3 提取的水体指数$AWEI_{nsh}$的结果

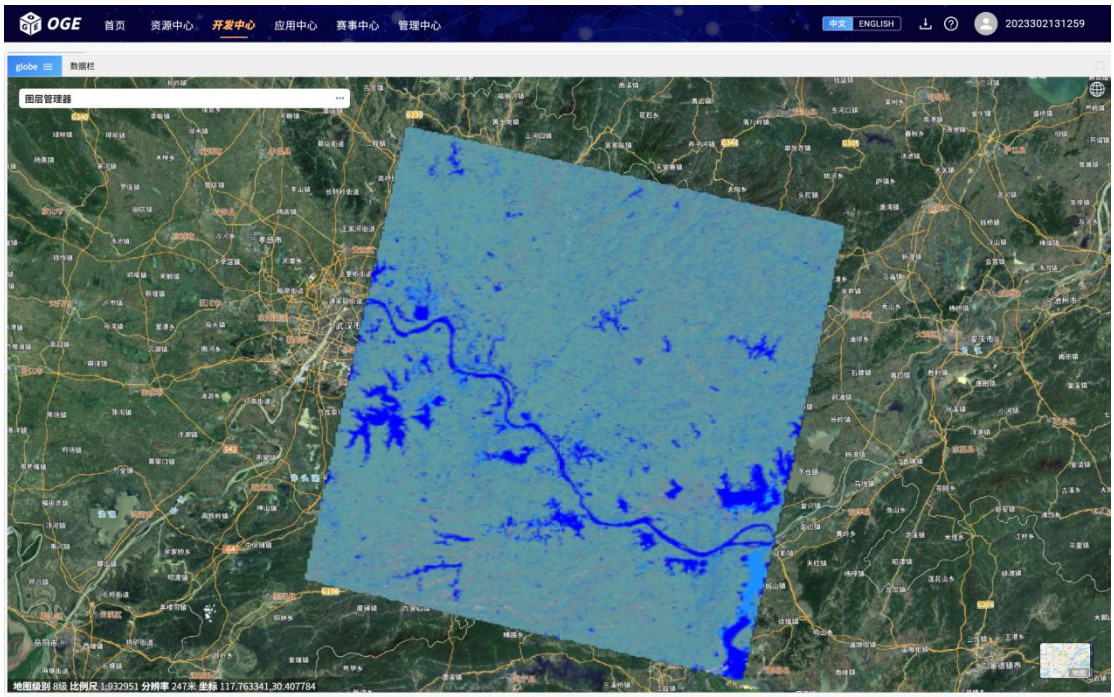


图3.4.2-4 提取的水体指数$AWEI_{sh}$的结果

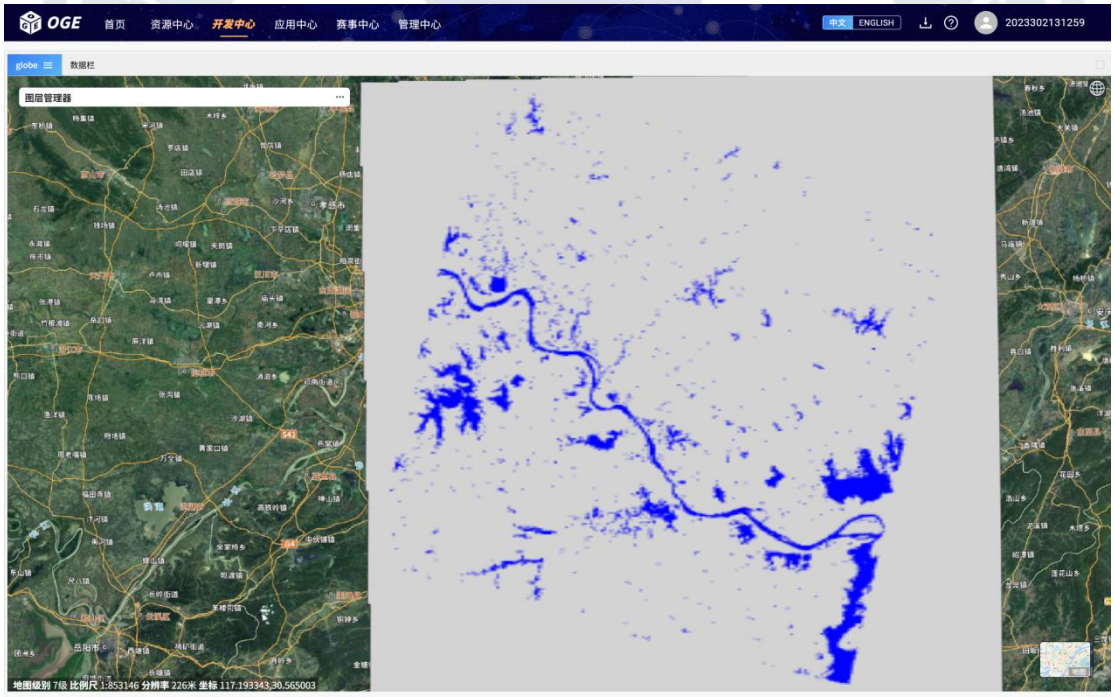


图3.4.2-5 基于提取的水体指数$AWEI_{nsh}$进行二值化的结果

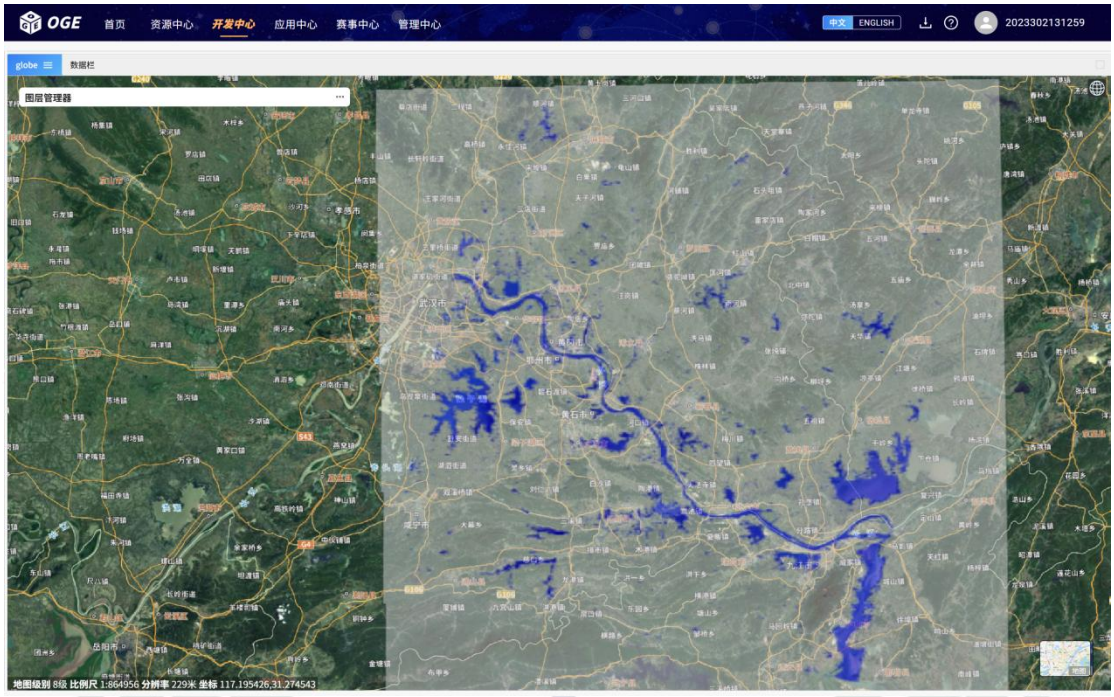


图3. 4. 2-6 基于提取的水体指数$AWEI_{nsh}$进行二值化的结果（半透明图层）

结果分析：图 3.4.2-3 和图 3.4.2-4 分别展示了提取的水体指数$AWEI_{nsh}$和$AWEI_{sh}$的结果。AWEI 具有抑制阴影与建筑干扰的优点，从结果来看，AWEI 能够较好地识别出水体，水体区域呈现出较强的水体信号，与非水体区域有明显的区分。

此外，AWEI 有阴影和非阴影两种公式，可以根据不同的地形条件选择不同的公式。对比图 3.4.2-3 和图 3.4.2-4 可以看出，对于地形起伏较大的山区或者房屋密集的城区，由于阴影区域面积较大，选用有阴影优化的 AWEI 公式得到的结果图 3.4.2-4 中误判“蓝色噪点”更少，能够更准确地提取水体信息。

然而，AWEI 的计算相对较为复杂，需要考虑多个波段的组合运算，会增加计算成本和时间。此外，AWEI 的适用性受到特定地区水体特性和环境条件的限制，在不同地区或不同季节，其参数设置需要进行相应的调整。

图 3.4.2-5 和图 3.4.2-6 展示了基于提取的水体指数$AWEI_{nsh}$进行二值化的结果。二值化处理后，水体与非水体区域的界限更加清晰，便于进行后续的分析和应用。通过设置合适的阈值，可以将水体区域与非水体区域明确区分开来，提高了水体检测的效率和准确性。将二值化结果设置为半透明图层（如图 3.4.2-6 所示），可以更直观地与原始遥感影像进行比对，验证水体检测的效果。

3.4.3 OGE 建筑检测结果截图

提取的建筑指数 NDVI 的结果如图 3.4.3-1 所示：

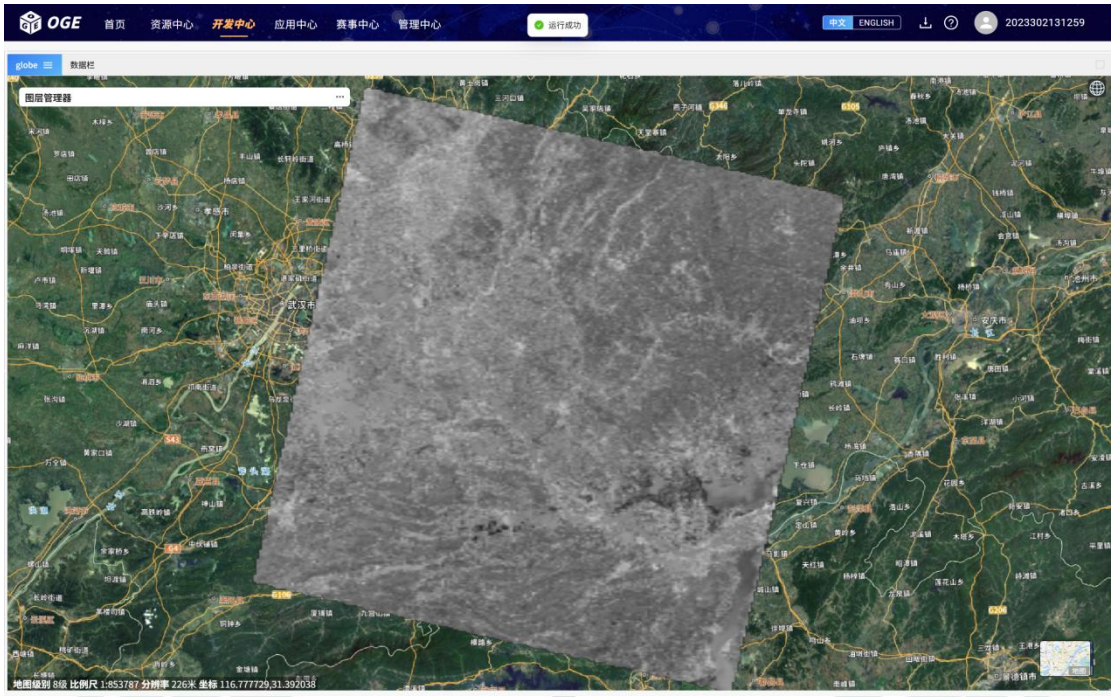


图3.4.3-1 提取的建筑指数NDVI的结果

结果分析：从图 3.4.3-1 提取的建筑指数 NDBI 的结果来看，建筑区域（主要集中于城市市区）的 NDBI 值较高，丘陵地区的 NDBI 值较低，因此 NDBI 在一定程度上能够识别出建筑用地。但是 NDBI 的识别结果其实没有之前看到的其他遥感指数的识别效果好，这是因为 NDBI 在区分建筑与植被等其他地物时可能会存在一定的困难——植被在近红外波段反射率较高，而建筑在短波红外波段反射率较高，这会导致 NDBI 值在某些情况下出现重叠，从而影响建筑检测的准确性；NDBI 也容易受到大气、土壤等其他因素的影响，从而降低建筑检测的精度。此外，建筑的分布本身也比植被、水体等的分布更加零散、随机，客观上建筑指数的提取也会比植被、水体等的提取更加困难。

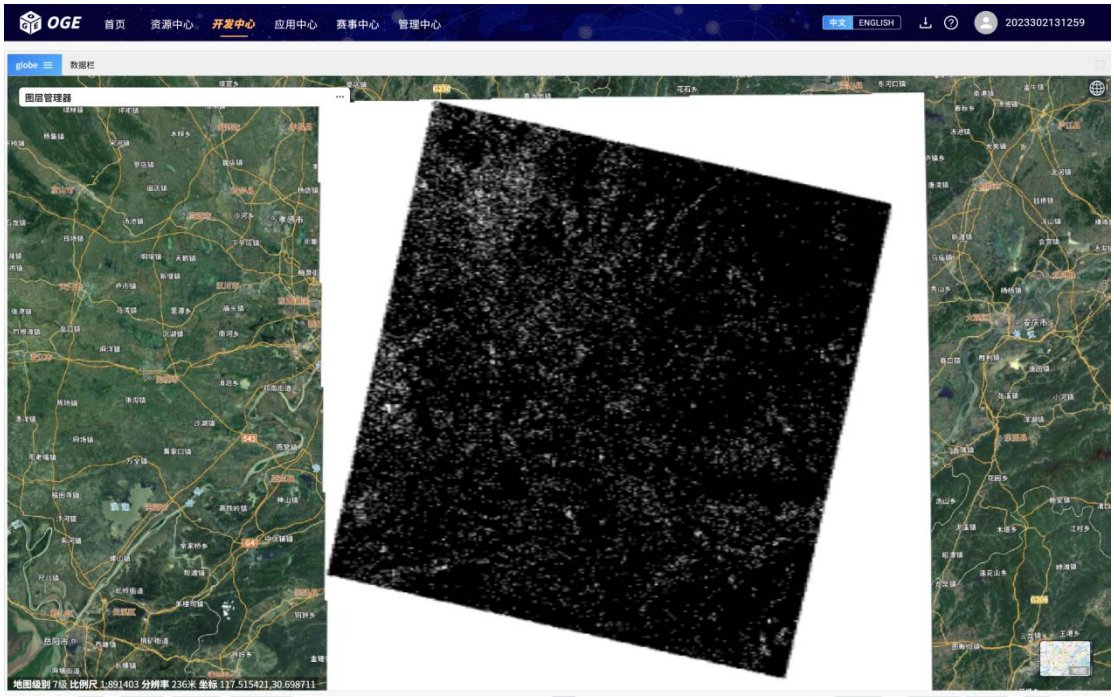


图3.4.3-1 提取的建筑指数IBI的结果

结果分析：图 3.4.3-2 展示了提取的建筑指数 IBI 的结果。IBI 是一个复合指数，综合考虑了 NDBI、SAVI 和 MNDWI，对建筑区域的识别精度比单纯使用 NDBI 高。通过融合建筑、水体和植被的指数信息，IBI 能够更全面地反映地物的特性，从而更准确地识别建筑区域。从结果来看，IBI 能够更好地区分建筑与非建筑区域，在靠近武汉市区等集中的建筑区域的 IBI 值较高，呈现出地表的建筑分布特征。

尽管 IBI 具有较高的识别精度，但其计算过程较为繁琐，需要计算三个指数（NDBI、SAVI 和 MNDWI），这可能会增加计算成本和时间。此外，IBI 的适用性也会受到特定地区地物特性和环境条件的限制，在不同地区或不同应用场景下，其参数设置需要进行相应的调整。

3.5 实习过程中遇到的问题及其解决方法

遇到的问题	解决方法
在OGE平台上进行植被指数计算时，代码编写过程中出现变量名未定义等小错误，导致程序无法正常运行	仔细检查代码并修改
在计算NDWI和MNDWI时，由于对Landsat-8波段的特性不够熟悉，导致在选择波段	查看OGE平台的资源中心里面对相应卫星的传感器的波段信息的介绍，并反复核对代码中的波段选用（如下图所示）

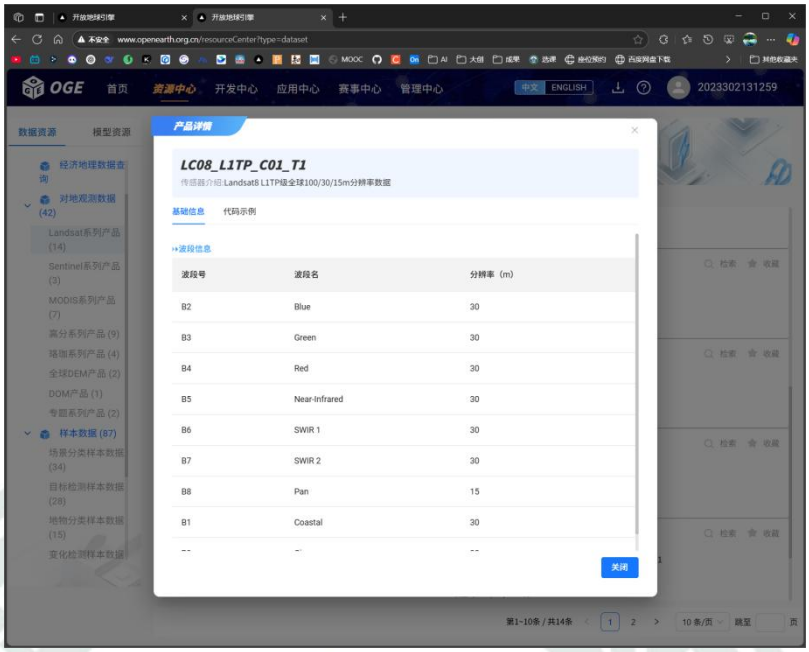
时出现错误		
在OGE平台中，点击保存代码之后，退出再回来的时候发现代码不见了		其实是因为它每次点击保存都会把代码另存为一个相同文件名的代码，而且这个新增的代码在下次打开的时候才会出现，因此下次打开的时候需要删除老一版本的那个代码，只保留新一版本的那个代码

表3.5-1 在应用OGE平台的基于特征指数遥感专题信息提取的实习过程中遇到的问题及其解决方法

3.6 实习小结

通过本节的基于开放地球引擎（OGE）平台的遥感专题信息提取实习，我深入掌握了利用遥感特征指数进行地物信息提取的原理与方法，并熟悉了 OGE 平台的操作流程与功能。以下是对本次实习的全面总结：

在这次实习之前，我在遥感原理与应用的理论课程中对遥感特征指数已有初步了解，但通过本次实习，我更加深入地理解了植被指数（如 RVI、NDVI）、水体指数（如 NDWI、MNDWI、AWEI）、建筑指数（如 NDBI、IBI）的构建原理与物理意义。例如，植被指数基于植被在不同波段的光谱反射特性差异，通过数学运算突出植被信息；水体指数则利用水体在特定波段的低反射率特征来识别水体。这些原理为后续的地物信息提取奠

定了坚实的理论基础。通过对不同指数的计算与结果分析，我清晰地认识到每种指数的适用场景与局限性。例如，RVI 在植被覆盖度较高时敏感性高，但在植被覆盖度低于50%时效果不佳；NDVI 虽然应用广泛，但在植被稀疏区域易受土壤背景等影响。这种对指数优缺点的深入理解，使我能够在实际应用中根据具体需求选择合适的指数或组合多种指数以提高信息提取的准确性。

此外，我还熟练掌握了我们武汉大学自主开发的 OGE 平台的常用功能，包括查找和调用卫星影像数据、上传自定义数据、导出处理结果等。我感受到 OGE 平台为遥感数据处理与分析提供了强大的支持。它集海量地理空间数据、交互式编程分析、实时分布式计算和数据可视化为一体，具备多源海量卫星影像、航空影像、高程产品等多类型数据的资源整合能力，能够满足不同用户的需求。在本次实习中，我深刻体会到了 OGE 平台的便捷性与高效性。通过平台提供的丰富数据资源和强大的计算能力，我能够快速获取所需的遥感影像，并进行各种复杂的指数计算与分析，无需担心数据存储和计算资源的限制。同时，平台的交互式编程环境和可视化功能，使我能够实时查看处理结果，及时调整分析策略，大大提高了工作效率和分析的准确性。

我使用 Python 风格的语言调用 OGE 平台提供的相关库函数，编写了多种指数计算代码，如植被指数 RVI、NDVI，水体指数 NDWI、MNDWI、AWEI，建筑指数 NDBI、IBI 等。通过这些编程实践，我不仅巩固了编程基础，还提升了对遥感数据处理流程的理解，学会了如何将理论公式转化为可执行的代码，并通过调试优化代码以获得准确的结果。

通过对提取结果的分析，我学会了如何根据指数值的范围和分布特征来判断地物类型，并结合实际地理背景对结果进行解读。例如，在植被指数结果中，能够区分出植被茂盛区域、植被稀疏区域以及非植被区域；在水体指数结果中，能够准确识别出河流、湖泊等水体，并分析其受阴影和建筑干扰的情况；在建筑指数结果中，能够识别出城市建筑区域，并探讨其与周边环境的关系。我意识到，遥感技术作为一种非接触、远距离的探测技术，能够在短时间内获取大范围地表信息，为地物信息提取提供了高效、便捷的手段。从本部分的实习就可以看出，利用植被指数可以快速评估植被覆盖度和生长状况，为生态环境保护和资源管理提供依据；水体指数能够准确提取水体信息，有助于水资源管理、洪涝灾害监测等工作；建筑指数则可用于城市规划、土地利用监测等。这些应用展示了遥感技术在解决实际问题中的巨大潜力，也让我更加坚定了深入学习和研究遥感技术的决心。

4 应用 Python GDAL 库编程实现遥感影像的镶嵌

4.1 实习目的

本次部分的实习旨在通过应用 Python GDAL 库编程实现遥感影像镶嵌技术，深入理解和掌握遥感数据处理的核心理论与实践方法。

具体目的包括：通过灵活运用所学习的理论知识，全面掌握遥感影像镶嵌的基本原理和技术流程，深入理解几何校正与配准、辐射校正与色彩平衡、重叠区域处理等关键技术环节的理论基础，并通过编程实现加深对相关算法的理解，熟练掌握 Python GDAL 库在遥感数据处理中的应用方法[29]。通过实际编程操作，学会使用 GDAL 库进行遥感影像的读取、处理、分析和输出，掌握地理坐标系统变换、影像重投影、重采样等核心技术，为后续的遥感应用开发奠定坚实的技术基础；深入研究和编程实现影像镶嵌算法，并在探索算法优化策略，提升解决复杂遥感数据处理问题的技术水平；学会从问题分析、方案设计、算法实现到结果评估的全流程技术开发方法，提升独立解决实际遥感应用问题的综合能力，为开发实用的遥感数据处理工具积累经验。

4.2 实习基本原理

遥感影像镶嵌是将多幅具有重叠区域的遥感影像拼接成一幅完整、连续影像的技术过程。该技术在遥感数据处理中具有重要意义，能够扩大观测范围、提高空间覆盖度，为大范围地理信息获取提供基础数据支撑。影像镶嵌的核心挑战在于消除重叠区域的拼接痕迹，实现视觉上的无缝融合。

一般而言，进行遥感影像镶嵌包括以下步骤：

1. 几何校正与配准原理

遥感影像镶嵌的首要前提是实现精确的几何校正。由于遥感平台的姿态变化、地形起伏、大气折射等因素影响，原始影像往往存在几何畸变[30]。几何校正通过建立影像坐标系与地理坐标系之间的数学变换关系，消除系统性几何误差。常用的变换模型包括仿射变换、多项式变换[31]和有理函数模型等[32]。

影像配准是镶嵌的核心环节，其原理基于同名点的识别与匹配。通过在重叠区域提取特征点（如角点、边缘点或 SIFT 特征点），建立影像间的对应关系。配准精度直接影响镶嵌质量，通常要求亚像素级精度。

此外，大范围影像镶嵌还需要考虑地球曲率的影响，选择合适的地图投影至关重要。

不同的投影方式在面积、角度、距离保持方面各有特点。通用横轴墨卡托投影（UTM）、兰伯特等角圆锥投影等在遥感应用中较为常见。投影变换过程中需要进行重采样，常用方法包括最近邻插值、双线性插值和三次卷积插值[33]。插值方法的选择需要在计算效率和图像质量之间权衡。

本部分实习中，所提供的两幅影像已经是经过几何校正与配准的，因此本步骤在本部分实习中可以不进行。

2. 辐射校正与色彩平衡原理

不同时相、不同传感器获取的影像在辐射特性上存在差异，表现为亮度、对比度和色彩的不一致性。辐射校正的目的是消除大气散射、传感器响应差异等因素造成的辐射畸变[34]。

色彩平衡通过统计学方法调整影像的直方图分布，使相邻影像在重叠区域的辐射特性趋于一致。常用方法包括直方图匹配、回归分析法和 Wallis 滤波等。这一过程确保镶嵌后的影像在视觉上呈现连续性和一致性。

本部分实习主要使用直方图匹配方法。

3. 重叠区域处理：融合算法/拼接算法

重叠区域的处理是镶嵌技术的关键难点。由于成像时间、观测角度、大气条件的差异，重叠区域往往存在辐射差异和几何不一致性。将两张有重叠区域的影像进行镶嵌（Mosaic）的方法一般分为两大类：依赖于镶嵌线的拼接算法与不依赖于镶嵌线的融合算法。

对于拼接算法，拼接缝的选择和优化是影响镶嵌质量的重要因素。理想的拼接缝应避开地物边界，选择灰度变化平缓的区域。动态规划算法通过构建代价函数，寻找最优拼接路径[35][36]。代价函数通常考虑像素差值、梯度信息和纹理特征等因素。

融合算法也有多种选择，简单的像素替换方法虽然计算效率高，但容易产生明显的拼接缝。加权平均法根据像素到影像边界的距离分配权重，实现平滑过渡。其他方法如羽化处理（Feathering）[37]、多分辨率融合和基于梯度域的融合技术，能够更好地保持影像的纹理细节和边缘信息。

本部分实习将会使用拼接算法中的覆盖镶嵌（Layer Stack）与镶嵌线（Seamline Stack）镶嵌方法，但也会在后处理时使用羽化等融合算法。

综上所述，遥感影像镶嵌是一个集几何处理、辐射校正、图像融合于一体的综合性技术过程。其基本原理涉及数字图像处理、计算机视觉、测量学等多个学科领域的理论基础，需要综合考虑精度、效率和视觉效果等多重因素，以实现高质量的影像镶嵌产品。

4.3 实习过程

我在本次实习中对遥感影像进行镶嵌的处理主要分为四个步骤：（1）波段叠加处理：将多个单波段 TIF 文件合并为多波段影像；（2）直方图匹配：基于重叠区域进行辐射校正；（3）基础覆盖镶嵌：生成初步镶嵌结果；（4）镶嵌线优化镶嵌：应用先进的镶嵌线算法优化镶嵌质量。在接下来对实习过程的介绍里，限于篇幅，我不会在本实习报告的正文中展示我所使用的所有代码，只展示关键的功能函数的代码，完整的代码请参见我在实习系统中提交的压缩包中的内容。**请注意：本节不重点阐述相关操作的原理（对相关原理的解释请参见 4.2 节），也不对结果展开具体分析（对所得结果的分析请参见 4.4 节）。**

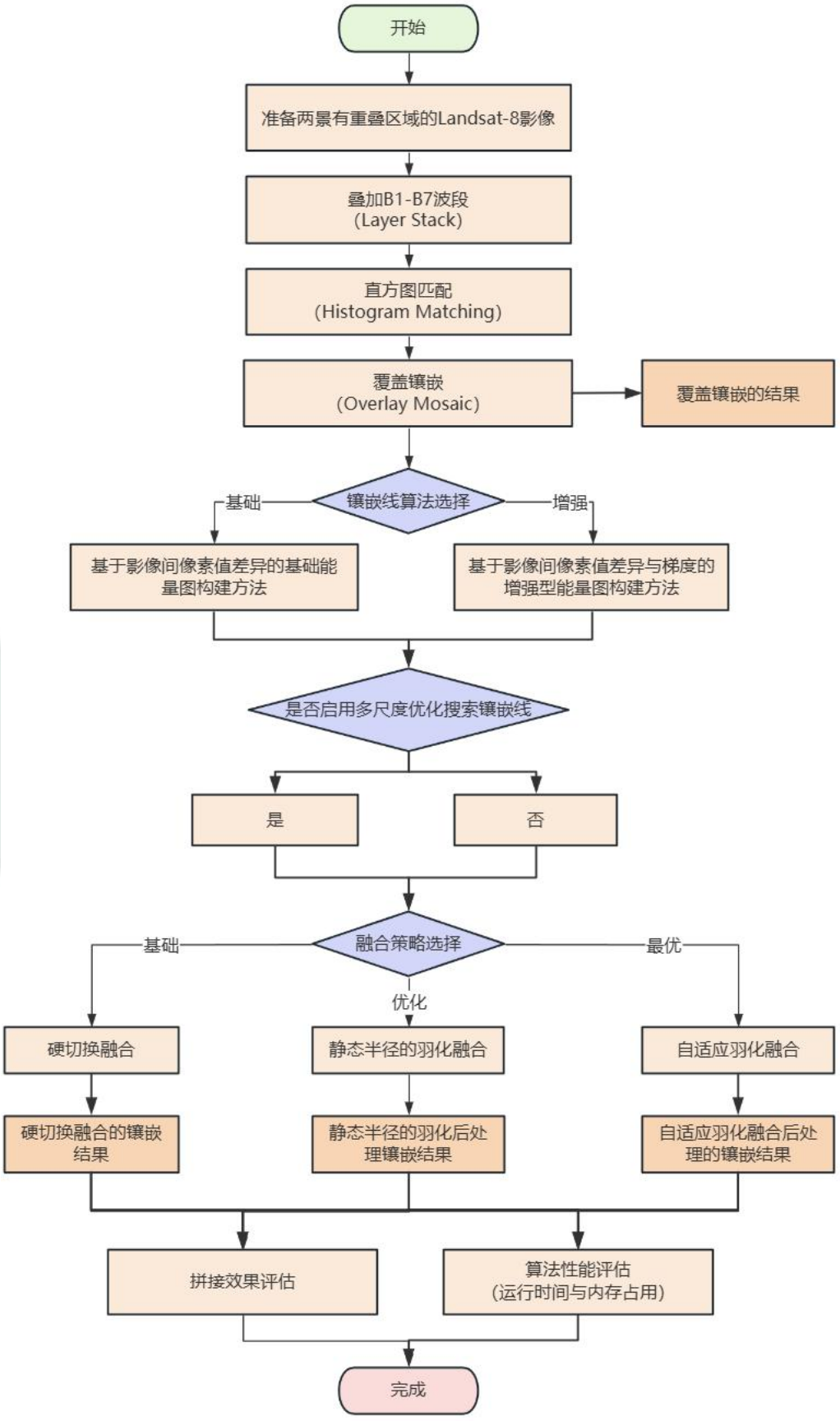


图4.3-1 应用Python GDAL库编程实现遥感影像的镶嵌流程图

4.3.1 层叠（Layer Stack）：将两景影像的多个波段的 TIF 合成一个多波段的 TIF

首先进行波段叠加（Layer Stack，层叠），将 Landsat 8 影像的多个波段其中的 7 个光谱波段（B1-B7，在提供的原始数据中每个波段以独立的 TIFF 文件形式存储）合并为一个多波段影像 TIF 文件。这一步骤为后续的多光谱分析和处理奠定了基础。

在层叠过程中，首先利用 GDAL（Geospatial Data Abstraction Library）库读取各个波段的地理参考信息，包括地理变换参数和投影信息。地理变换参数以六元组形式表示：$(X_0, \Delta X, R_1, Y_0, R_2, \Delta Y)$，其中$X_0$和$Y_0$表示影像左上角的地理坐标，$\Delta X$和$\Delta Y$表示像素在 X 和 Y 方向的分辨率，$R_1$和$R_2$表示旋转参数（通常为 0）。像素坐标与地理坐标的转换关系为：

$$X_{geo} = X_0 + X_{pix} \times \Delta X + Y_{pix} \times R_1$$

$$Y_{geo} = Y_0 + X_{pix} \times R_2 + Y_{pix} \times \Delta Y$$

```
#读取原始输入影像的地理信息，及像素坐标与地理坐标的转换信息
geotransform = input.GetGeoTransform()
projection = input.GetProjectionRef()
meta = input.GetMetadata()
print(f"坐标转换参数: Xgeo = {geotransform[0]} + Xpix * {geotransform[1]} + Ypix
* {geotransform[2]}")
print(f"Ygeo = {geotransform[3]} + Xpix * {geotransform[4]} + Ypix *
({geotransform[5]})")
print(f"投影信息: {projection}")
```

通过创建三维数组结构存储多波段数据，其中第一维表示波段数，第二维和第三维分别表示影像的行数和列数。这种数据组织方式不仅便于后续的数值计算，也为影像的可视化和分析提供了便利。层叠完成后，将合成的多波段影像以 TIF 格式保存，确保地理参考信息的完整保留。

可以通过第 2 章使用的 ERDAS IMAGINE 2015 软件打开得到的 TIF 文件以进行查看。在将得到的 TIF 文件导入 ERDAS IMAGINE 2015 软件时可能会出现如图 4.3.1-1 所示的提示：

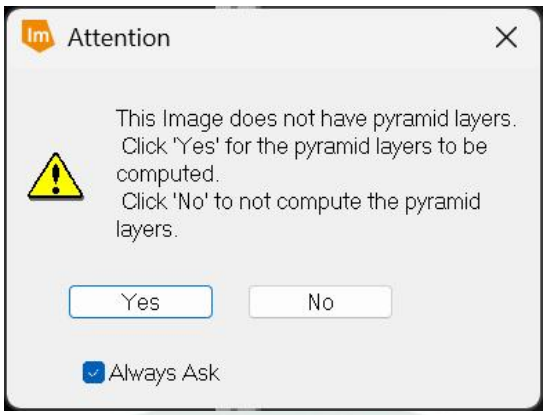


图4. 3. 1-1 提示先进行影像金字塔层的构建

金字塔层（Pyramid Layers）是图像处理中的一种技术，用于创建图像的多分辨率表示，有助于提高某些操作的性能和质量，类似于在 GIS（Geographic Information System，地理信息系统）中分层级地、以不同详略程度显示地图的技术。点击“Yes”，就会看见影像金字塔层正在构建（图 4.3.1-2）：

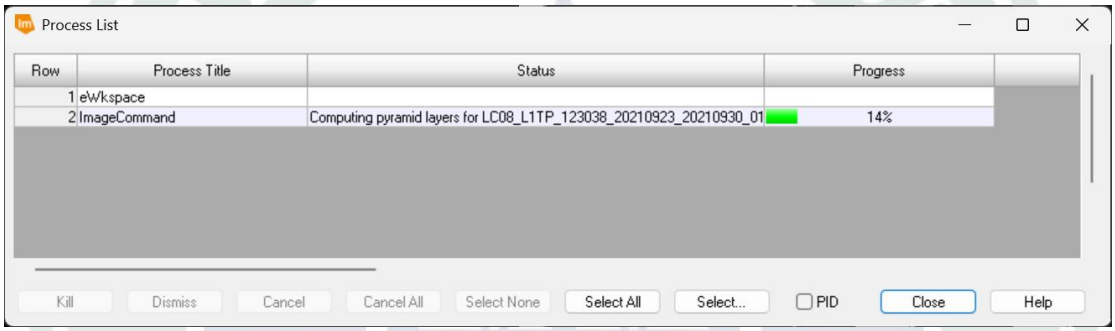


图4. 3. 1-2 构建遥感影像的金字塔层

金字塔层构建完毕后，可以在这个 TIF 文件的相同路径下看见新产生了 2 个文件，即构建的金字塔层的数据文件（图 4.3.1-3）：

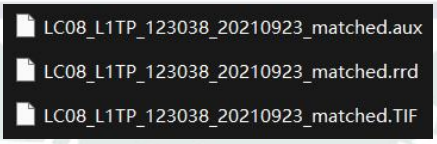


图4. 3. 1-3 导入的TIF文件及其金字塔层文件

此时可能会看见导入的 TIF 图层右下角有一个红色圆形叉符号或者一个黄色三角形感叹号，此时可以右键点击导入的 TIF 图层，选择 Correct the Alert Problem（图 4.3.1-4）：

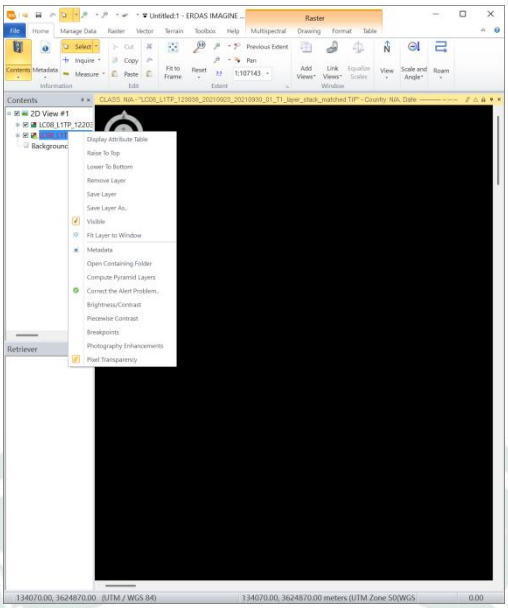


图4.3.1-4 Correct the Alert Problem

然后点击 Fit Layer To Window 来将图层缩放到屏幕，即可看到层叠后得到的遥感影像（图 4.3.1-5）：

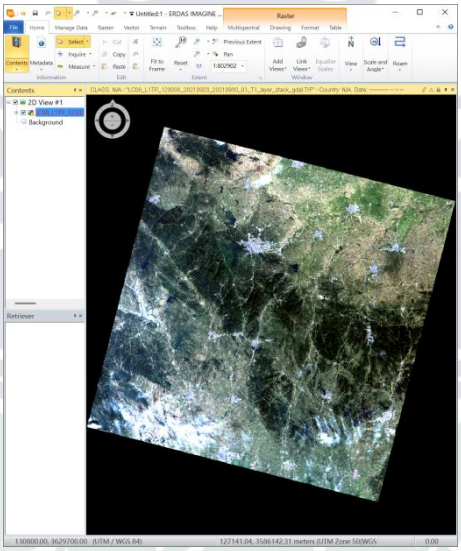


图4.3.1-5 层叠后得到的遥感影像

将左右两景影像都进行层叠处理后导入 ERDAS IMAGINE 2015 软件如图 4.3.1-6 所示：

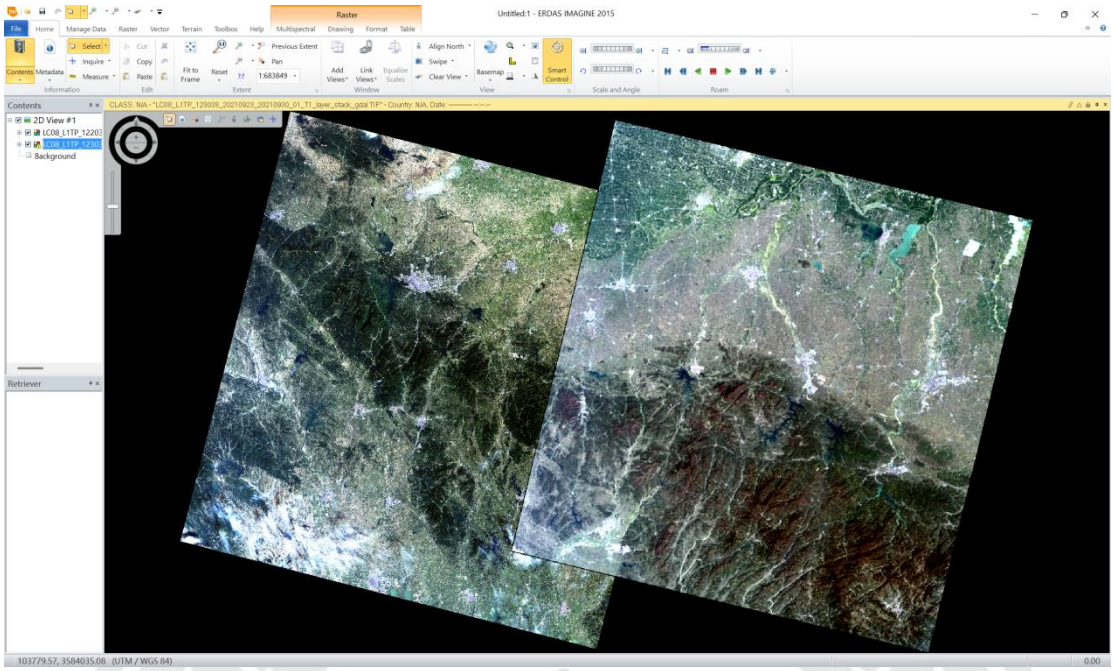


图4. 3. 1-6 层叠处理后的左右两景影像

可以将图 4.3.1-6 与图 4.3.2-1 对比，以验证 4.3.2 节中进行直方图匹配的效果。

4.3.2 根据重叠区域的像素进行直方图匹配（Histogram Matching）

由于两景影像获取时间不同，大气条件、太阳高度角、传感器状态等因素的差异会导致影像间存在显著的辐射差异。这种差异如果不加以校正，将在镶嵌结果中产生明显的色调跳跃现象。因此，影响融合的第二个关键步骤是进行基于重叠区域的直方图匹配（Histogram Matching）。

直方图匹配的理论基础是统计学中的累积分布函数（CDF）变换。对于源影像 S 和参考影像 R，设其像素值的概率密度函数分别为$p_S(s)$和$p_R(r)$，对应的累积分布函数为：

$$CDF_S(s) = \int_0^s p_S(u)\,du$$

$$CDF_R(r) = \int_0^r p_R(u)\,du$$

直方图匹配的目标是找到一个变换函数 T，使得变换后的源影像 T(S)与参考影像 R 具有相同的直方图分布。根据概率论原理，这一变换可通过以下关系实现：

$$T(s) = {CDF_S}^{-1}(CDF_S(s))$$

我在本次实习的实际实现中，首先，为了避免无效像素对直方图匹配结果造成干扰，需要识别并移除影像边缘的无效区域（所有波段的像素值为 0），进而识别出包含有效数据的最小外接矩形区域。具体实现的方法是通过 histogram_matching()函数创建布尔掩膜标识所有波段均不为无效值（通常为 0）的像素，然后计算这些有效像素的空间分布范围。

```
def crop_to_valid_data(data_array, geotransform, nodata_value=0):
    """
    将影像裁剪到有效数据区域

    Parameters:
    data_array: 输入数据数组 (bands, rows, cols)
    geotransform: 地理变换参数
    nodata_value: 无效数据值

    Returns:
    cropped_data: 裁剪后的数据
    new_geotransform: 新的地理变换参数
    """
    bounds = get_valid_bounds(data_array, nodata_value)

    if bounds is None:
        print("警告：没有找到有效数据！")
        return data_array, geotransform

    min_row, max_row, min_col, max_col = bounds

    # 裁剪数据
    cropped_data = data_array[:, min_row:max_row, min_col:max_col]

    # 更新地理变换参数
    new_geotransform = list(geotransform)
    new_geotransform[0] = geotransform[0] + min_col * geotransform[1]  # 更新左上角X坐标
    new_geotransform[3] = geotransform[3] + min_row * geotransform[5]  # 更新左上角Y坐标

    print(f"裁剪区域：行[{min_row}:{max_row}], 列[{min_col}:{max_col}]")
    print(f"裁剪后尺寸：{cropped_data.shape[2]}x{cropped_data.shape[1]}")

    return cropped_data, tuple(new_geotransform)
```

然后，基于两幅影像的地理变换参数，计算影像间的空间重叠区域。具体来说，首先根据地理变换参数计算每幅影像的地理边界范围，包括最小和最大的 X、Y 坐标值。然后通过坐标比较确定重叠区域的地理范围，最后将地理坐标转换为对应的像素坐标范围。这一过程确保了后续**直方图统计仅基于空间上真实重叠的区域进行**。

```
def get_overlap_region(geo1, rows1, cols1, geo2, rows2, cols2):
    """
    计算两个影像的重叠区域
```

```
    Parameters:
    geo1, geo2: 地理变换参数
    rows1, cols1, rows2, cols2: 影像尺寸

    Returns:
    重叠区域的像素坐标范围
    """
    # 计算影像1的地理范围
    x1_min = geo1[0]
    y1_max = geo1[3]
    x1_max = geo1[0] + cols1 * geo1[1]
    y1_min = geo1[3] + rows1 * geo1[5]

    # 计算影像2的地理范围
    x2_min = geo2[0]
    y2_max = geo2[3]
    x2_max = geo2[0] + cols2 * geo2[1]
    y2_min = geo2[3] + rows2 * geo2[5]

    # 计算重叠区域的地理范围
    overlap_x_min = max(x1_min, x2_min)
    overlap_x_max = min(x1_max, x2_max)
    overlap_y_min = max(y1_min, y2_min)
    overlap_y_max = min(y1_max, y2_max)

    # 检查是否有重叠
    if overlap_x_min >= overlap_x_max or overlap_y_min >= overlap_y_max:
        print("警告：两个影像没有重叠区域！")
        return None, None, None, None

    # 将地理坐标转换为像素坐标
    # 影像1的重叠区域像素坐标
    col1_start = int((overlap_x_min - geo1[0]) / geo1[1])
```

```
    col1_end = int((overlap_x_max - geo1[0]) / geo1[1])
    row1_start = int((overlap_y_max - geo1[3]) / geo1[5])
    row1_end = int((overlap_y_min - geo1[3]) / geo1[5])

    # 影像 2 的重叠区域像素坐标
    col2_start = int((overlap_x_min - geo2[0]) / geo2[1])
    col2_end = int((overlap_x_max - geo2[0]) / geo2[1])
    row2_start = int((overlap_y_max - geo2[3]) / geo2[5])
    row2_end = int((overlap_y_min - geo2[3]) / geo2[5])

    return (row1_start, row1_end, col1_start, col1_end), \
           (row2_start, row2_end, col2_start, col2_end)
```

然后**对影像的每个波段分别进行**直方图匹配处理。histogram_matching()函数首先对输入数据进行预处理，移除无效像素值，然后计算源影像和参考影像的**重叠区域的直方图分布**。通过 numpy 的 histogram()函数生成 256 个分箱的直方图，并计算对应的累积分布函数。利用线性插值方法建立像素值映射关系，将源影像**整幅影像**的像素值重新分配以匹配参考影像在**两幅影像重叠区域**的直方图分布。

```
def histogram_matching(source, template, bins=256):
    """
    直方图匹配函数

    Parameters:
    source: 源影像数据
    template: 参考影像数据
    bins: 直方图分箱数

    Returns:
    匹配后的影像数据
    """
    # 展平数组
    source_flat = source.flatten()
    template_flat = template.flatten()

    # 移除无效值
    source_valid = source_flat[source_flat > 0]
    template_valid = template_flat[template_flat > 0]

    if len(source_valid) == 0 or len(template_valid) == 0:
        return source
```

```
    # 计算累积分布函数
    source_hist, source_bins = np.histogram(source_valid, bins=bins,
density=True)
    template_hist, template_bins = np.histogram(template_valid, bins=bins,
density=True)

    source_cdf = np.cumsum(source_hist) * (source_bins[1] - source_bins[0])
    template_cdf = np.cumsum(template_hist) * (template_bins[1] -
template_bins[0])

    # 创建映射函数
    source_cdf = np.clip(source_cdf, 0, 1)
    template_cdf = np.clip(template_cdf, 0, 1)

    # 插值映射
    matched = np.interp(source_flat, source_bins[:-1], source_cdf)
    matched = np.interp(matched, template_cdf, template_bins[:-1])

    # 保持原始的0值不变
    matched_reshaped = matched.reshape(source.shape)
    matched_reshaped[source == 0] = 0

    return matched_reshaped
```

匹配完成后，将处理结果保存为 TIF 格式文件。使用 GDAL 的 GTiff 驱动创建输出文件，设置 LZW 压缩和分块存储选项以优化文件大小和读取性能。同时设置正确的地理变换参数和投影信息，确保输出影像具有完整的地理参考信息。对于每个波段，将匹配后的像素值限制在有效范围内并转换为 16 位无符号整型格式。

直方图匹配的效果如图 4.3.2-1 所示，右景是参考影像 LC08_L1TP_122038_20210324_20210401_01_T1_layer_stack_gdal.TIF（准确地说，是左右两景的**重叠区域**的右景影像的直方图分布作为参考），左景是经过直方图匹配后的 LC08_L1TP_123038_20210923_matched.TIF。

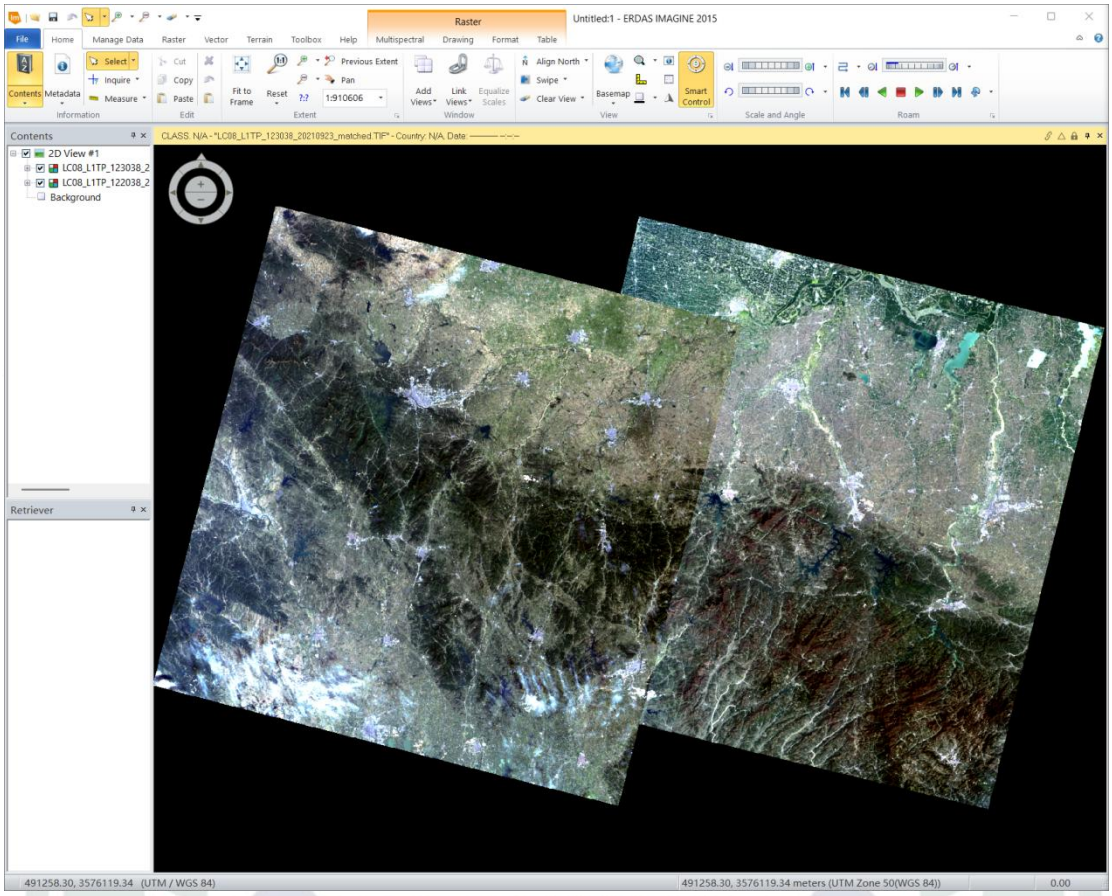


图4. 3. 2-1 直方图匹配的效果

将图 4.3.2-1 与图 4.3.1-6 相对比，从视觉上来看左景影像的色彩特征更接近右景了，证明直方图匹配有效。

此外，为了能够从另一个角度展示直方图匹配的效果，验证匹配后影像的直方图分布是否接近参考影像实现了直方图可视化功能，我还通过 matplotlib 库绘制参考影像、原始影像和匹配后影像的直方图对比图（图 4.3.2-2）：

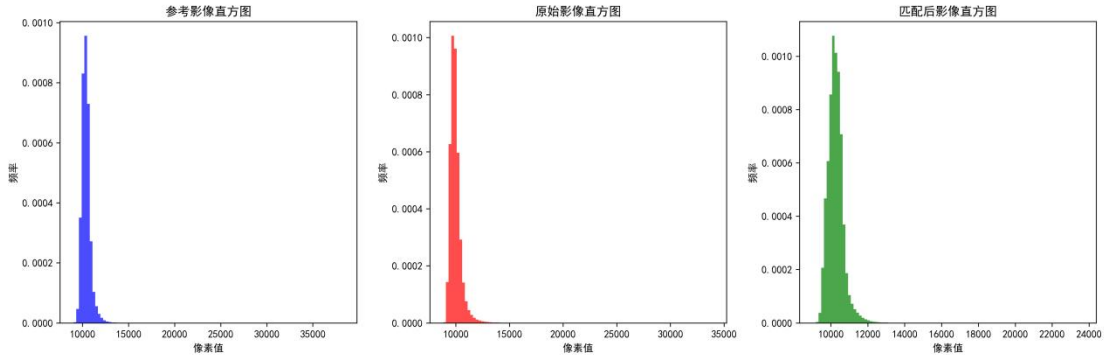


图4. 3. 2-1 直方图对比图

4.3.3 覆盖镶嵌（Overlay Mosaic）

在完成直方图匹配后，首先实施基础的覆盖镶嵌操作。覆盖镶嵌是一种简单直接的

影像镶嵌方法，其基本原理是在重叠区域采用“后来者居上”的策略，将后处理的影像直接覆盖在基础影像之上，类似于计算机图形学中的画家算法（Painter 's Algorithm）。

覆盖镶嵌的实现过程涉及复杂的地理坐标系统计算，因此需要使用基于 GDAL 库的专业遥感数据处理方法。首先通过 gdal.Open()函数以只读模式打开两幅输入影像，读取影像的像素数据和地理参考信息，包括地理变换参数和投影坐标系统。地理变换参数包含了影像左上角的地理坐标、像元分辨率以及旋转参数，这些信息对于后续的空间配准至关重要。

首先需要确定输出镶嵌影像的地理范围，这通过计算两幅输入影像地理边界的并集来实现：

$$X_{min} = min(X_{1_{min}}, X_{2_{min}})$$

$$X_{max} = max(X_{1_{max}}, X_{2_{max}})$$

$$Y_{min} = min(Y_{1_{min}}, Y_{2_{min}})$$

$$Y_{max} = max(Y_{1_{max}}, Y_{2_{max}})$$

```
# 检查投影是否一致
if proj1 != proj2:
    print("警告：影像投影不一致，可能影响镶嵌结果！")
# 计算输出影像的范围
x_min = min(geo1[0], geo2[0])
y_max = max(geo1[3], geo2[3])
x_max = max(geo1[0] + base_ds.RasterXSize * geo1[1],
            geo2[0] + matched_ds.RasterXSize * geo2[1])
y_min = min(geo1[3] + base_ds.RasterYSize * geo1[5],
            geo2[3] + matched_ds.RasterYSize * geo2[5])
```

输出影像的分辨率选择输入影像中的最高分辨率，以保证镶嵌结果的空间细节。输出影像的尺寸计算公式为：

$$Cols = \left\lfloor \frac{X_{max} - X_{min}}{Res_X} \right\rfloor$$

$$Rows = \left\lfloor \frac{Y_{max} - Y_{min}}{Res_Y} \right\rfloor$$

其中Res_X和Res_Y分别表示 X 和 Y 方向的像素分辨率。

在镶嵌过程中，采用 GDAL 库实现高效的影像重投影和重采样，并设置多线程处理选项，充分利用计算机的多核处理能力，显著提升镶嵌效率。

```
# 设置波段属性
```

```
    for i in range(1, base_ds.RasterCount + 1):
        out_band = out_ds.GetRasterBand(i)
        out_band.SetNoDataValue(0)

    # 使用 gdal.Warp 进行镶嵌
    print("正在进行影像镶嵌...")
    gdal.Warp(out_ds, [base_image_path, matched_image_path],
              format='GTiff',
              outputBounds=(x_min, y_min, x_max, y_max),
              resampleAlg=gdalconst.GRA_NearestNeighbour,
              srcNodata=0,
              dstNodata=0,
              multithread=True,
              warpOptions=['NUM_THREADS=ALL_CPUS'])
```

在输出 GeoTIFF 格式影像时，作者使用 COMPRESS=LZW 参数启用无损压缩，有效减少了文件大小；TILED=YES 参数采用了瓦片存储结构，提高了大影像的读写效率；BIGTIFF=IF_NEEDED 参数确保了对大于 4GB 文件的支持。这些技术细节体现了专业遥感数据处理中对效率和兼容性的综合考虑。

```
    # 计算输出影像的尺寸
    res_x = min(abs(geo1[1]), abs(geo2[1]))
    res_y = min(abs(geo1[5]), abs(geo2[5]))

    cols = int((x_max - x_min) / res_x)
    rows = int((y_max - y_min) / res_y)

    # 创建输出影像
    print("正在创建输出影像...")
    driver = gdal.GetDriverByName('GTiff')
    options = ['COMPRESS=LZW', 'TILED=YES', 'BIGTIFF=IF_NEEDED']
    out_ds = driver.Create(output_path, cols, rows, base_ds.RasterCount,
                           gdal.GDT_UInt16, options=options)

    # 设置输出影像的地理参考
    out_geo = (x_min, res_x, 0, y_max, 0, -res_y)
    out_ds.SetGeoTransform(out_geo)
    out_ds.SetProjection(proj1)
```

覆盖镶嵌的效果如图 4.3.3-1 所示：

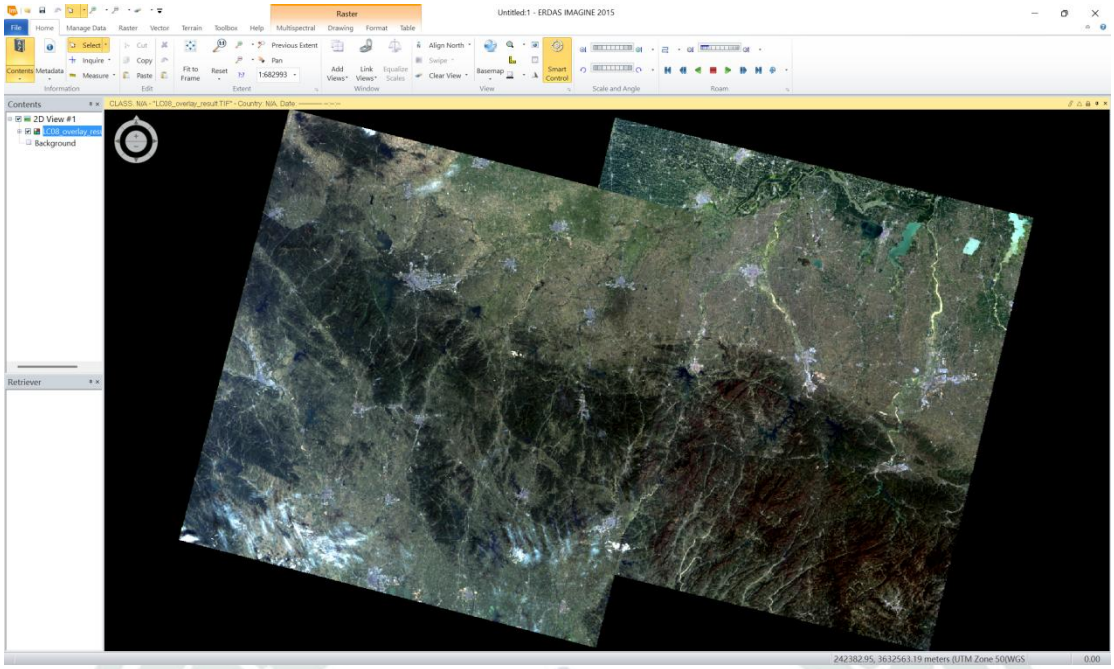


图4. 3. 3-1 覆盖镶嵌的效果

可见，覆盖镶嵌的结果中有明显的覆盖边界的“分界线”拼接痕迹，因此需要进一步使用更强大的影像镶嵌算法。

4.3.4 镶嵌线镶嵌（Seamline Mosaic）

覆盖镶嵌虽然操作简便，但在重叠区域往往产生明显的拼接痕迹（图 4.3.3-1）。为获得更高质量的镶嵌结果，实习的核心技术环节是实现基于镶嵌线的影像镶嵌算法，并在此基础上进一步优化。

4.3.4.1 重叠区域精确定位

镶嵌线算法的前提是准确识别两幅影像的重叠区域。这一过程需要进行精确的地理坐标计算。设两幅影像的地理变换参数分别为 Geo_1 和 Geo_2，影像尺寸分别为($Rows_1$, $Cols_1$)和($Rows_2$, $Cols_2$)，则各自的地理边界为：

影像 1：

$$X_{1min} = Geo_1[0]$$

$$X_{1max} = Geo_1[0] + Cols_1 \times Geo_1[1]$$

$$Y_{1max} = Geo_1[3]$$

$$Y_{1min} = Geo_1[3] + Rows_1 \times Geo_1[5]$$

重叠区域的地理边界通过求交集获得：

$$Overlap_{X_{min}} = max(X_{1min}, X_{2min})$$

$$Overlap_{X_{max}} = min(X_{1max}, X_{2max})$$

$$Overlap_{Y_{min}} = max(Y_{1min}, Y_{2min})$$

$$Overlap_{Y_{max}} = min(Y_{1max}, Y_{2max})$$

随后将地理坐标转换为各影像的像素坐标，为后续的镶嵌线计算提供精确的数据范围。

```
def get_overlap_bbox(self, geo1, shape1, geo2, shape2):
        """计算重叠区域"""
        x1_min, y1_max = geo1[0], geo1[3]
        x1_max = x1_min + shape1[2] * geo1[1]
        y1_min = y1_max + shape1[1] * geo1[5]
        x2_min, y2_max = geo2[0], geo2[3]
        x2_max = x2_min + shape2[2] * geo2[1]
        y2_min = y2_max + shape2[1] * geo2[5]
        overlap_x_min = max(x1_min, x2_min)
        overlap_x_max = min(x1_max, x2_max)
        overlap_y_min = max(y1_min, y2_min)
        overlap_y_max = min(y1_max, y2_max)
        if overlap_x_min >= overlap_x_max or overlap_y_min >= overlap_y_max:
            return None, None
        def geo_to_pixel(geo, x, y):
            px = int((x - geo[0]) / geo[1])
            py = int((y - geo[3]) / geo[5])
            return px, py
        c1_start, r1_start = geo_to_pixel(geo1, overlap_x_min, overlap_y_max)
        c1_end, r1_end = geo_to_pixel(geo1, overlap_x_max, overlap_y_min)
        c2_start, r2_start = geo_to_pixel(geo2, overlap_x_min, overlap_y_max)
        c2_end, r2_end = geo_to_pixel(geo2, overlap_x_max, overlap_y_min)
        return (r1_start, r1_end, c1_start, c1_end), (r2_start, r2_end, c2_start,
c2_end)
```

4.3.4.2 能量图构建与优化

能量图是镶嵌线算法的核心组件，它量化了重叠区域内每个像素位置作为拼接点的适宜程度。基础能量图通过计算对应像素间的**各个波段上**的像素值绝对差值构建：

$$E_{basic}(i,j) = \left|I_1(i,j) - I_2(i,j)\right|$$

```
def compute_energy_map(self, over1, over2):
    """基础能量图计算"""
    # 确保输入是 3 维数组
    if over1.ndim == 2:
        over1 = over1.reshape(1, *over1.shape)
    if over2.ndim == 2:
        over2 = over2.reshape(1, *over2.shape)
    diff = np.abs(over1.astype(np.int32) - over2.astype(np.int32))
    return np.mean(diff, axis=0)
```

为提高镶嵌线质量，我还实现了**增强型能量图**，该方法在基础差值的基础上**引入梯度信息**：

$$E_{enhanced}(i,j) = \left|I_1(i,j) - I_2(i,j)\right| + \alpha \times \left|\nabla I_1(i,j) - \nabla I_2(i,j)\right|$$

其中梯度信息通过 Sobel 算子计算：

$$\nabla I(i,j) = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

梯度信息的引入使得算法**倾向于在纹理变化较小的区域寻找镶嵌线，从而减少拼接痕迹的可见性**。权重系数α是一个超参数（Hyperparameter），其具体取值需要根据具体影像特征来调参。

```
def compute_energy_map_enhanced(self, over1, over2):
    """增强的能量图计算"""
# 确保输入是 3 维数组
    if over1.ndim == 2:
        over1 = over1.reshape(1, *over1.shape)
    if over2.ndim == 2:
        over2 = over2.reshape(1, *over2.shape)

    # 原始差值
    diff = np.abs(over1.astype(np.int32) - over2.astype(np.int32))

    # 添加梯度信息
    def compute_gradient(img):
        grad_x = np.abs(np.diff(img, axis=2))
        grad_y = np.abs(np.diff(img, axis=1))
        # 填充边界
        grad_x = np.pad(grad_x, ((0,0), (0,0), (0,1)), mode='edge')
        grad_y = np.pad(grad_y, ((0,0), (0,1), (0,0)), mode='edge')
        return grad_x + grad_y
```

```
        grad1 = compute_gradient(over1)
        grad2 = compute_gradient(over2)

        # 综合能量：差值 + 梯度差异
        energy = np.mean(diff, axis=0) + self.config['gradient_weight'] *
np.mean(np.abs(grad1 - grad2), axis=0)

        return energy
```

4.3.4.3 基于动态规划的镶嵌线搜索

镶嵌线搜索采用**动态规划算法**。该算法将寻找最优镶嵌线的问题转化为在能量图上寻找从顶部到底部的最小代价路径问题。

设能量图的尺寸为 M×N（M 行 N 列），定义状态转移方程：

$$Cost(i,j) = E(i,j) + min\{Cost(i-1,j-1), Cost(i-1,j), Cost(i-1,j+1)\}$$

其中$Cost(i,j)$表示从第一行到达位置(i,j)的最小累积代价。算法从第二行开始，逐行计算每个位置的最小累积代价，同时记录最优路径的来源方向。

在路径回溯阶段，从最后一行中选择累积代价最小的位置作为终点，然后根据记录的路径信息逆向追踪到起点，从而得到完整的镶嵌线路径。

```
        def find_seamline(self, energy):
            """基础镶嵌线查找"""
            rows, cols = energy.shape
            cost = energy.copy()
            path = np.zeros_like(cost, dtype=int)
            for i in range(1, rows):
                for j in range(cols):
                    left = cost[i-1, j-1] if j > 0 else 1e10
                    up = cost[i-1, j]
                    right = cost[i-1, j+1] if j < cols - 1 else 1e10
                    min_val = min(left, up, right)
                    if min_val == left:
                        path[i, j] = j - 1
                    elif min_val == right:
                        path[i, j] = j + 1
                    else:
                        path[i, j] = j
                    cost[i, j] += min_val
            seam = np.zeros(rows, dtype=int)
            seam[-1] = np.argmin(cost[-1])
            for i in range(rows - 2, -1, -1):
                seam[i] = path[i + 1, seam[i + 1]]
```

```
            return seam
```

4.3.4.4 最终镶嵌结果生成

镶嵌线融合完成后，需要将融合结果与基础镶嵌影像进行最终的叠置操作。这一过程涉及复杂的地理坐标系统对齐和数据融合。

首先读取基础镶嵌影像和镶嵌线融合结果，计算两者的地理范围并集，确定最终输出影像的空间范围。然后创建输出影像数据结构，设置适当的地理变换参数和投影信息。

```
        def write_output(self, path, data, geo, proj):
                """写出结果"""
                bands, rows, cols = data.shape
                driver = gdal.GetDriverByName('GTiff')
                out_ds = driver.Create(path, cols, rows, bands,
gdal.GDT_UInt16, options=['COMPRESS=LZW'])
                out_ds.SetGeoTransform(geo)
                out_ds.SetProjection(proj)
                for i in range(bands):
                        out_band = out_ds.GetRasterBand(i+1)
                        out_band.WriteArray(data[i])
                        out_band.SetNoDataValue(0)
                out_ds.FlushCache()
                out_ds = None
```

在数据写入过程中，采用分层叠置策略：首先将由 4.3.3 中通过覆盖镶嵌得到的基础镶嵌影像 LC08_overlay_result.TIF 作为底层写入输出影像，然后将镶嵌线融合结果作为上层进行叠置。在重叠区域，镶嵌线融合结果具有更高的优先级，从而实现高质量的无缝拼接效果。

```
        def overlay_seamline(self, base_mosaic_path, seamline_mosaic_path,
output_path):
                """将 seamline 镶嵌结果叠置到基础覆盖镶嵌结果中"""
                self.status_updated.emit("读取基础镶嵌影像...")
                base_ds = gdal.Open(base_mosaic_path, gdalconst.GA_ReadOnly)
                seamline_ds = gdal.Open(seamline_mosaic_path,
gdalconst.GA_ReadOnly)
                if base_ds is None or seamline_ds is None:
                        raise Exception("无法打开影像文件！")
                # 获取影像信息
                base_geo = base_ds.GetGeoTransform()
                seamline_geo = seamline_ds.GetGeoTransform()
                base_proj = base_ds.GetProjectionRef()
                seamline_proj = seamline_ds.GetProjectionRef()
                # 计算输出影像的范围
```

```
                x_min = min(base_geo[0], seamline_geo[0])
                y_max = max(base_geo[3], seamline_geo[3])
                x_max = max(base_geo[0] + base_ds.RasterXSize *
base_geo[1],
                                    seamline_geo[0] +
seamline_ds.RasterXSize * seamline_geo[1])
                y_min = min(base_geo[3] + base_ds.RasterYSize *
base_geo[5],
                                    seamline_geo[3] +
seamline_ds.RasterYSize * seamline_geo[5])
                # 计算输出影像的分辨率
                res_x = min(abs(base_geo[1]), abs(seamline_geo[1]))
                res_y = min(abs(base_geo[5]), abs(seamline_geo[5]))
                cols = int((x_max - x_min) / res_x)
                rows = int((y_max - y_min) / res_y)
                # 创建输出影像
                self.status_updated.emit("创建最终输出影像...")
                self.progress_updated.emit(70)

                driver = gdal.GetDriverByName('GTiff')
                options = ['COMPRESS=LZW', 'TILED=YES', 'BIGTIFF=IF_NEEDED']
                out_ds = driver.Create(output_path, cols, rows,
base_ds.RasterCount,
                                                    gdal.GDT_UInt16,
options=options)
                # 设置输出影像的地理参考
                out_geo = (x_min, res_x, 0, y_max, 0, -res_y)
                out_ds.SetGeoTransform(out_geo)
                out_ds.SetProjection(base_proj)
                # 设置波段属性并初始化为 0
                for i in range(1, base_ds.RasterCount + 1):
                        out_band = out_ds.GetRasterBand(i)
                        out_band.SetNoDataValue(0)
                        out_band.WriteArray(np.zeros((rows, cols),
dtype=np.uint16))
                # 将基础影像写入输出影像
                self.status_updated.emit("叠置基础影像...")
                self.progress_updated.emit(75)

                gdal.Warp(out_ds, [base_mosaic_path],
                                format='GTiff',
                                outputBounds=(x_min, y_min, x_max, y_max),
                                resampleAlg=gdalconst.GRA_NearestNeighbour,
                                srcNodata=0,
```

```
                        dstNodata=0,
                        multithread=True,
                        warpOptions=['NUM_THREADS=ALL_CPUS'])
        #  将seamline影像写入输出影像
        self.status_updated.emit("叠置seamline影像...")
        self.progress_updated.emit(80)

        gdal.Warp(out_ds,  [seamline_mosaic_path],
                        format='GTiff',
                        outputBounds=(x_min,  y_min,  x_max,  y_max),
                        resampleAlg=gdalconst.GRA_NearestNeighbour,
                        srcNodata=0,
                        dstNodata=0,
                        multithread=True,
                        warpOptions=['NUM_THREADS=ALL_CPUS'])
        #  清理内存
        out_ds  =  None
        base_ds  =  None
        seamline_ds  =  None
```

最终得到的基础的/梯度增强镶嵌线算法的镶嵌结果如图 4.3.4.4-1、图 4.3.4.4-2 所示：

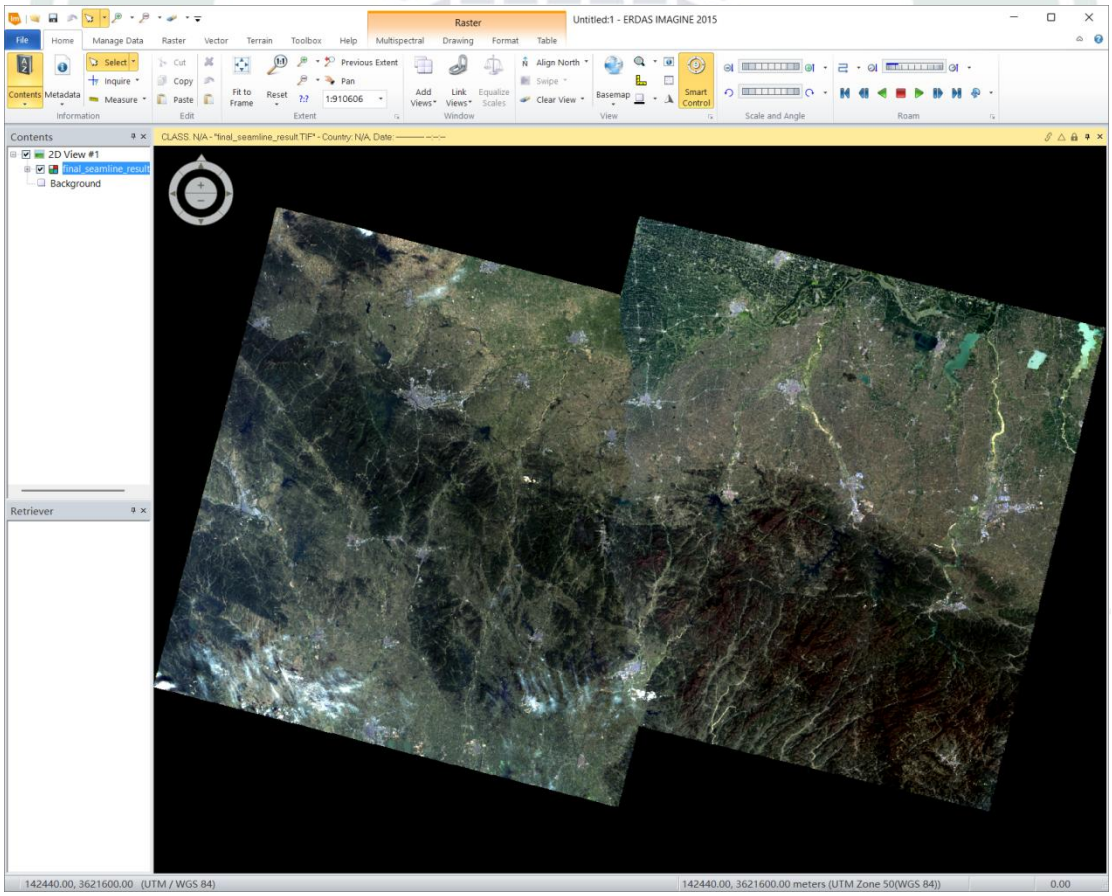


图4.3.4.4-1 基础的镶嵌线算法的镶嵌结果

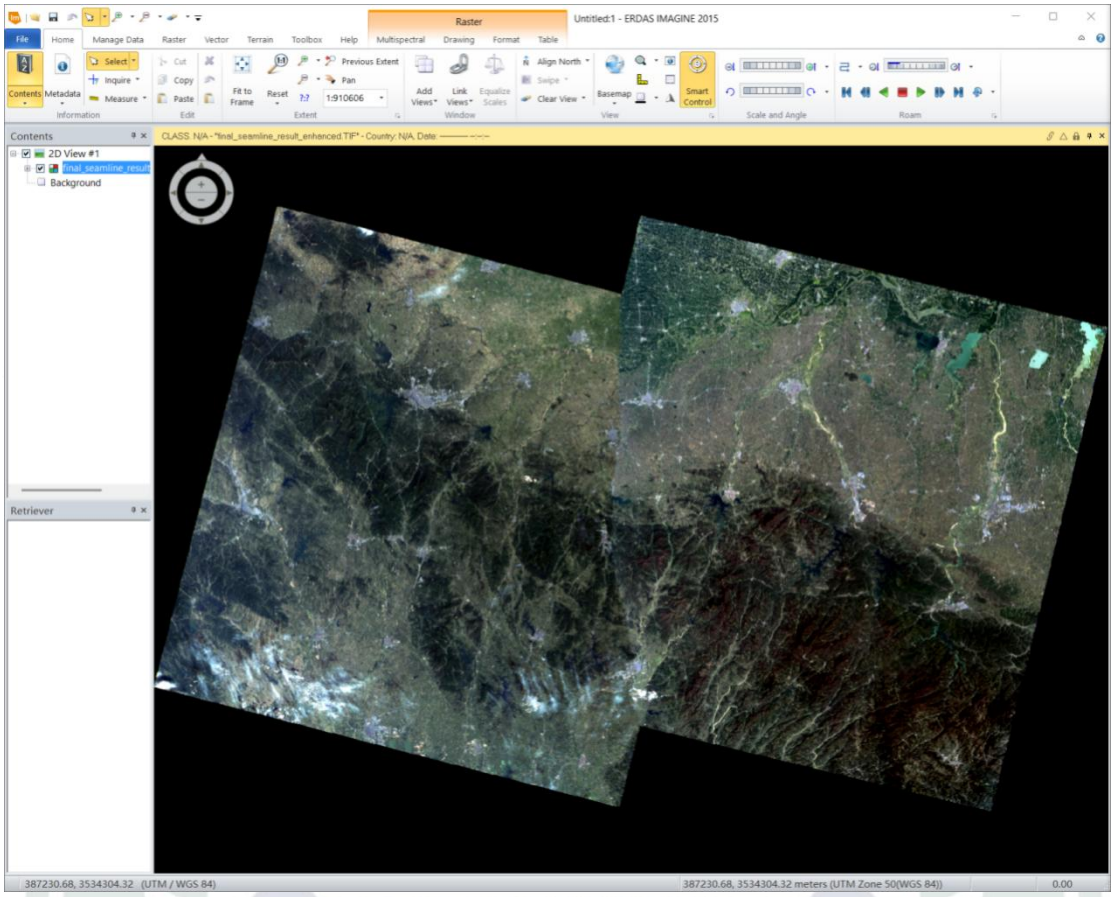


图4. 3. 4. 4-2 梯度增强的镶嵌线算法的镶嵌结果

为直观地评估镶嵌质量，实习过程中生成了镶嵌线可视化图，将计算得到的镶嵌线叠加在能量图上显示，直观展现镶嵌线的走向和能量分布特征。

```
        def save_visualization(self, energy, seam, output_path):
            """保存可视化"""
            plt.figure(figsize=(10, 8))
            plt.imshow(energy, cmap='gray')
            plt.plot(seam, np.arange(len(seam)), color='red',
linewidth=2)
            plt.title("镶嵌线可视化")
            plt.xlabel("列")
            plt.ylabel("行")
            plt.colorbar(label='能量值')
            vis_path = output_path.replace('.TIF', '_visualization.png')
            plt.savefig(vis_path, dpi=300, bbox_inches='tight')
            plt.close()
```

基础的/梯度增强镶嵌线算法的镶嵌线可视化结果如图 4.3.4.4-3 所示：

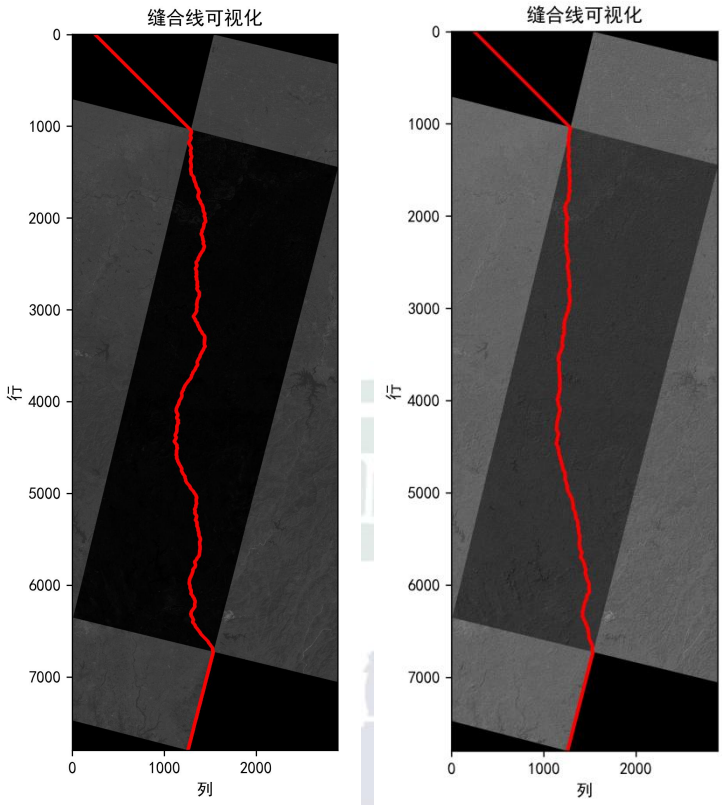


图4.3.4.4-3 基础的（左）/梯度增强的（右）镶嵌线算法的镶嵌线可视化结果

可见经过梯度增强的镶嵌线与基础镶嵌线有所不同，更加平顺一些。

4.3.5 镶嵌线算法进一步优化探索

4.3.5.1 多尺度优化策略

在 2024 年的 CVPR 上的一篇有武汉大学遥感信息工程学院的师生参与发表的论文 ***SkySense: A Multi-Modal Remote Sensing Foundation Model Towards Universal Interpretation for Earth Observation Imagery***（图 4.3.5.1-1）中提到了一种方法：Multi-Granularity Contrastive Learning（MGCL，多粒度对比学习），通过在空间上的像素级、对象级和图像级三个粒度上进行对比学习，其中像素级学习通过直接对比时间序列中每个空间位置的特征来捕获细粒度的局部信息，对象级学习通过无监督聚类将相似像素归为一类后进行对比来学习中等粒度的语义对象表示，而图像级学习则通过全局平均池化后的特征对比来获取整体场景的高层语义信息，使得模型能够同时学习到适用于不同任务需求的多层次特征表示，既能支持需要精确像素分类的语义分割任务，又能胜任需要对象级理解的目标检测任务，还能处理需要全局语义的场景分类任务，最终作为一个统一的基础模型在多种遥感解译任务中获得了优异性能[38]。

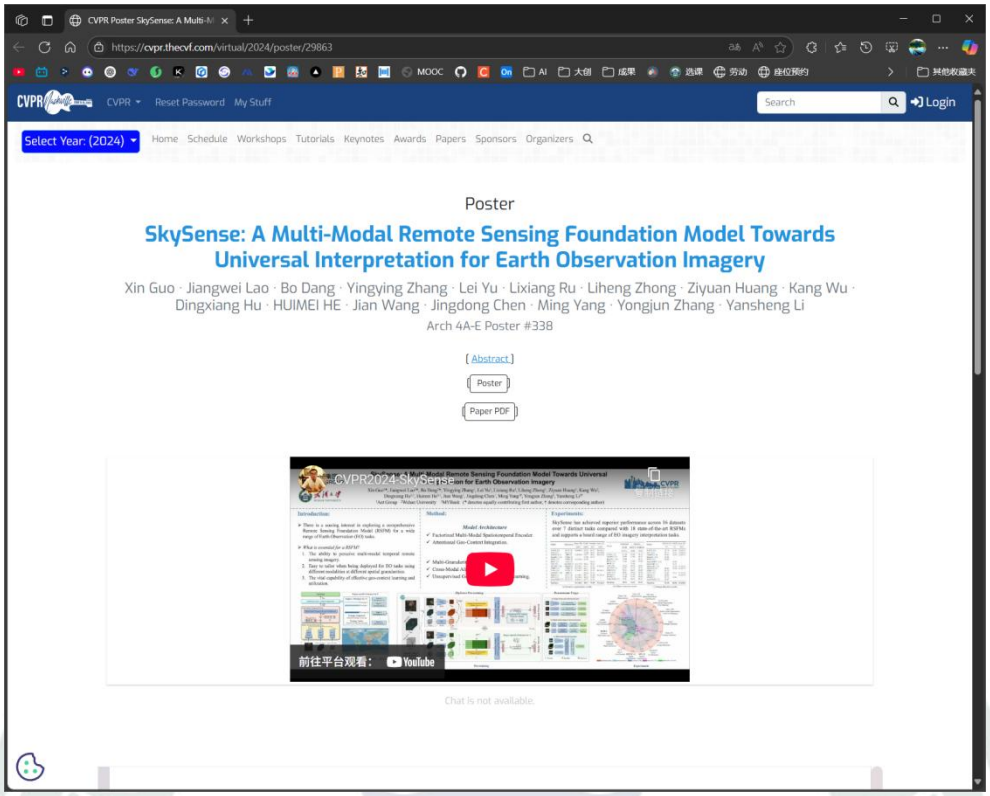


图4.3.5.1-1 SkySense: A Multi-Modal Remote Sensing Foundation Model Towards Universal Interpretation for Earth Observation Imagery 论文（CVPR 2024）[39]

受此启发，为提高镶嵌线的全局最优性，我实现了**多尺度镶嵌线搜索策略**。该方法首先在下采样的低分辨率影像上进行粗略的镶嵌线搜索，然后将结果上采样到原始分辨率作为精细搜索的初始解。

多尺度策略的数学描述为：设原始影像分辨率为 R_0 ，下采样尺度为 S，则低分辨率影像的尺寸为$\left\lfloor \frac{M}{S} \right\rfloor \times \left\lfloor \frac{N}{S} \right\rfloor$。在低分辨率上得到镶嵌线$L_{low}$后，通过线性插值将其映射到原始分辨率：

$$L_{high}(i) = L_{low}\left(\left\lfloor \frac{i}{S} \right\rfloor\right) \times S$$

这种多尺度策略不仅在一定程度上提高了算法的计算效率，还相当于增大了机器学习中优化器（Optimizer）进行梯度下降（Gradient Descent）的步长（Step），使得在动态规划搜索镶嵌线的过程中有更大的可能跳出局部最优解，增强了镶嵌线的全局最优性。

```
    def find_seamline_multiscale(self, energy, scales):
        """多尺度镶嵌线查找"""
        best_seam = None
        best_cost = float('inf')

        for scale in scales:
            if scale > 1:
                # 下采样
                h, w = energy.shape
```

```
                small_energy = energy[::scale, ::scale]
                small_seam = self.find_seamline(small_energy)

                # 上采样回原尺寸
                seam = np.repeat(small_seam * scale, scale)[:h]
            else:
                seam = self.find_seamline(energy)

            # 计算总成本
            cost = sum(energy[i, seam[i]] for i in range(len(seam)) if seam[i] < energy.shape[1])
            if cost < best_cost:
                best_cost = cost
                best_seam = seam

        return best_seam
```

通过多尺度优化策略得到的镶嵌线效果及镶嵌线可视化结果如图 4.3.5.1-2 所示：

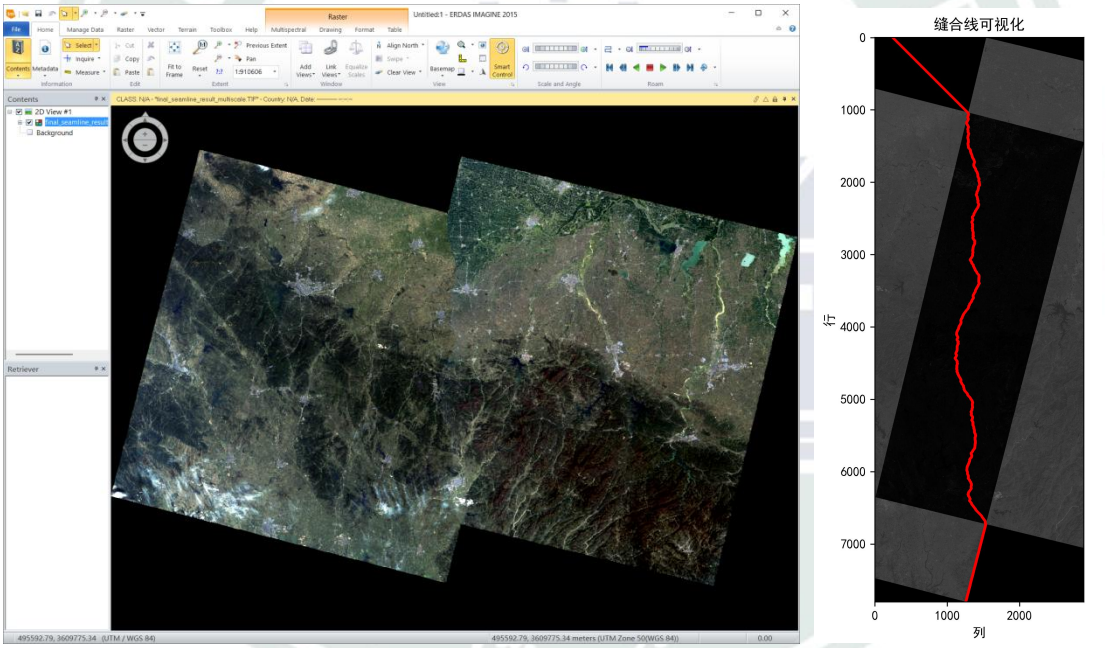


图4. 3. 5. 1-2 通过多尺度优化策略得到的镶嵌线效果及镶嵌线可视化结果

4.3.5.2 镶嵌线融合时的静态及自适应地羽化后处理算法设计

获得镶嵌线后，需要沿着镶嵌线进行影像融合。实习中我对镶嵌线镶嵌的算法进行了许多优化探索。

4.3.5.2.1 基础镶嵌线融合

基础融合方法采用硬切换策略，即在镶嵌线的左侧使用第一幅影像的像素值，右侧

使用第二幅影像的像素值。数学表达式为：

$$Result(i,j) = \begin{cases} I_1(i,j), if\ j < Seamline(i) \\ I_2(i,j), if\ j \geq Seamline(i) \end{cases}$$

```
    def find_seamline(self, energy):
        """基础镶嵌线查找"""
        rows, cols = energy.shape
        cost = energy.copy()
        path = np.zeros_like(cost, dtype=int)
        for i in range(1, rows):
            for j in range(cols):
                left = cost[i-1, j-1] if j > 0 else 1e10
                up = cost[i-1, j]
                right = cost[i-1, j+1] if j < cols - 1 else 1e10
                min_val = min(left, up, right)
                if min_val == left:
                    path[i, j] = j - 1
                elif min_val == right:
                    path[i, j] = j + 1
                else:
                    path[i, j] = j
                cost[i, j] += min_val
        seam = np.zeros(rows, dtype=int)
        seam[-1] = np.argmin(cost[-1])
        for i in range(rows - 2, -1, -1):
            seam[i] = path[i + 1, seam[i + 1]]
        return seam
```

这种方法实现简单，计算效率高，但在某些情况下可能产生轻微的边缘效应，即在镶嵌线处还是看得出拼接痕迹（图 4.3.5.2.1-1）。

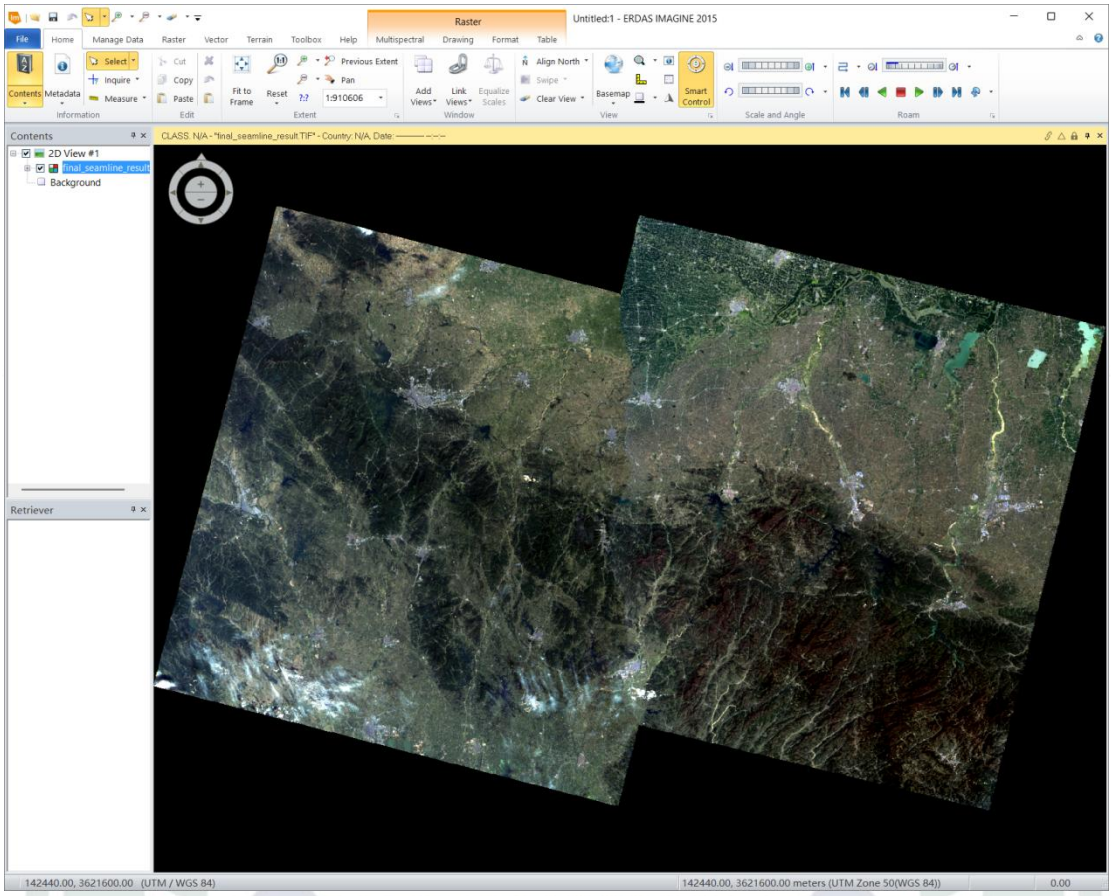


图4. 3. 5. 2. 1-1 基础镶嵌线的融合结果

4.3.5.2.2 静态羽化半径的羽化融合技术

羽化融合通过在镶嵌线附近建立过渡带来实现平滑拼接。在过渡带内，两幅影像的像素值按照距离镶嵌线的远近进行加权平均。设羽化宽度为 W，则融合公式为：

对于位置(i,j)，如果$|j - Seamline(i)| \leq W$，则：

$$t = \frac{j - [Seamline(i) - W]}{2W}$$

$$Result(i,j) = (1-t) \times I_1(i,j) + t \times I_2(i,j)$$

其中t为归一化的权重系数，采用余弦插值函数以获得更平滑的过渡效果：

$$t = 0.5 \times [1 - cos(\pi \times t_{linear})]$$

羽化宽度的可以自行设置，并且程序确保了过渡带不超出重叠区域的边界。

```
def blend_with_feathering(self, over1, over2, seam, feather_width):
    """
    羽化融合 - 羽化范围限制在重叠区域边界内
    处理像素值为 0 的情况：如果其中一个像素为 0，则使用非 0 像素值
    """
    rows, cols = over1.shape[1:]
    result = np.zeros_like(over1)
    if self.overlap_width is not None:
```

```
            # 羽化宽度不应超过重叠区域宽度
            max_feather_width = max(5, self.overlap_width)
            adaptive_feather_width = min(feather_width, max_feather_width)
            self.status_updated.emit(f"羽化参数 - 设置值: {feather_width}, 实际使
用: {adaptive_feather_width}, 重叠区域:
{self.overlap_width}×{self.overlap_height}")
        else:
            adaptive_feather_width = feather_width
        for r in range(rows):
            c = min(seam[r], cols - 1)
            # 基础分割，直接复制重叠区域内的像素值
            result[:, r, :c] = over2[:, r, :c]
            result[:, r, c:] = over1[:, r, c:]
            # 羽化区域 - 限制在重叠区域边界内
            start_feather = max(0, c - adaptive_feather_width)
            end_feather = min(cols, c + adaptive_feather_width)
            start_feather = max(start_feather, 0)  # 不能小于重叠区域左边界
            end_feather = min(end_feather, cols)   # 不能大于重叠区域右边界
            if end_feather > start_feather:
                for fc in range(start_feather, end_feather):
                    # 使用平滑的余弦权重函数而不是线性权重
                    t = (fc - start_feather) / (end_feather - start_feather)
                    # 余弦插值，更平滑
                    weight = 0.5 * (1 - np.cos(np.pi * t))

                    # 修改：处理像素值为0的情况
                    pixel1 = over1[:, r, fc]
                    pixel2 = over2[:, r, fc]

                    # 检查是否有像素值为0（对所有波段进行检查）
                    pixel1_is_zero = np.all(pixel1 == 0)
                    pixel2_is_zero = np.all(pixel2 == 0)

                    if pixel1_is_zero and not pixel2_is_zero:
                        # 如果pixel1为0，使用pixel2
                        result[:, r, fc] = pixel2
                    elif pixel2_is_zero and not pixel1_is_zero:
                        # 如果pixel2为0，使用pixel1
                        result[:, r, fc] = pixel1
                    elif pixel1_is_zero and pixel2_is_zero:
                       # 如果两个都为0，保持为0
                        result[:, r, fc] = 0
                    else:
                        # 如果两个都不为0，执行正常的插值
```

```
                    result[:, r, fc] = (1 - weight) * pixel2 + weight * pixel1
        return result
```

羽化融合的效果如图 4.3.5.2.2-1 所示：

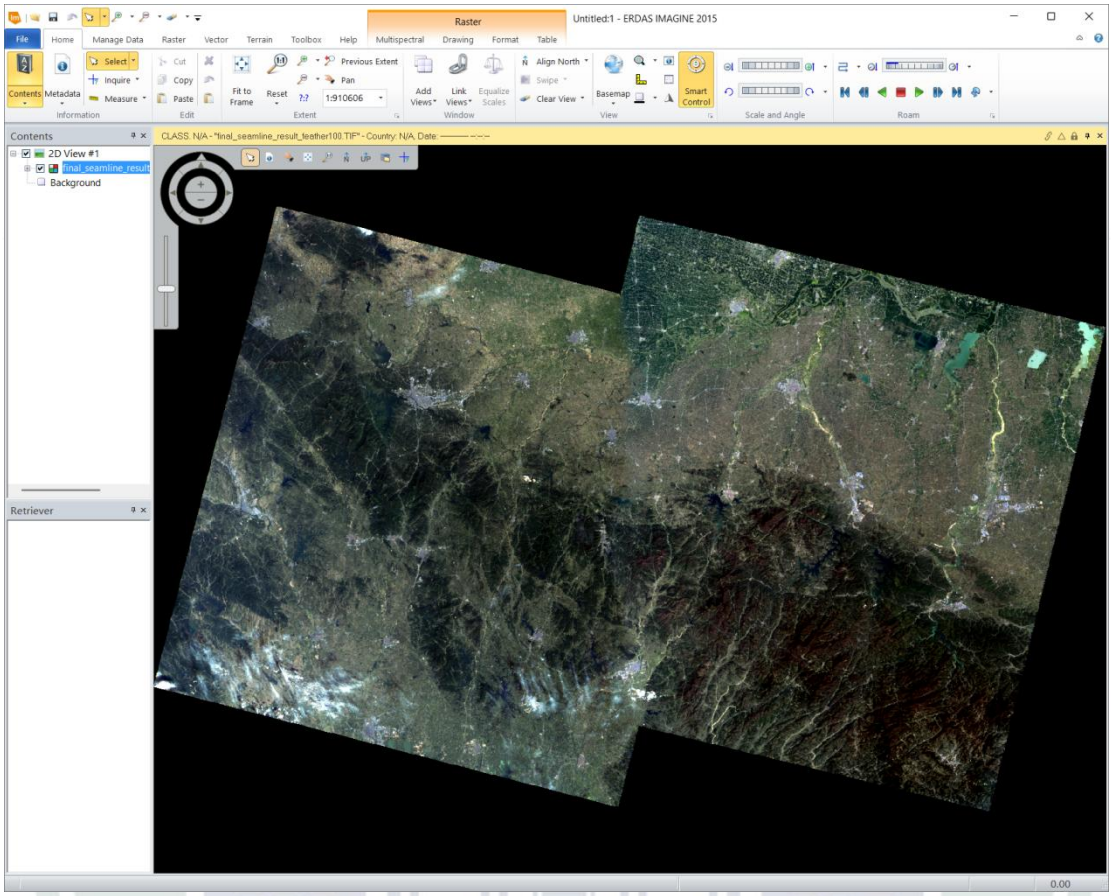


图4. 3. 5. 2. 2-1 羽化融合的效果（羽化半径为100）

可见，经过羽化处理后，原本在图 4.3.4.4-1 中较为明显的在镶嵌线处的拼接痕迹在图 4.3.5.2.2-1 中得到了改善，两幅影像之间的过渡变得更为平滑了。

进一步增大羽化半径，在镶嵌线处的拼接痕迹会进一步减小，过渡能够变得进一步平滑（图 4.3.5.2.2-2）：

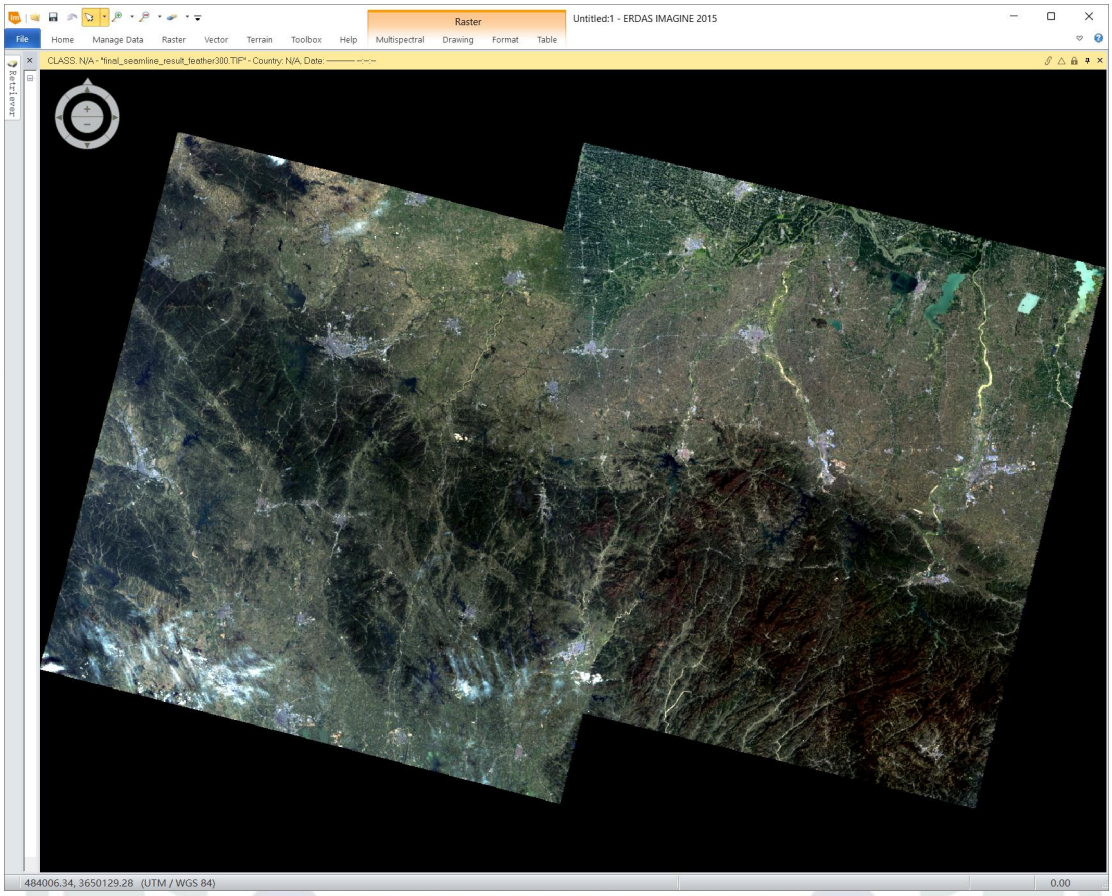


图4. 3. 5. 2. 2-2 羽化融合的效果（羽化半径为300）

4.3.5.2.3 自适应羽化融合

自适应羽化融合进一步优化了 4.3.5.2.3 中的羽化技术，根据局部纹理复杂度动态调整羽化宽度。纹理复杂度通过梯度幅值计算：

$$Texture(i,j) = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

羽化宽度的自适应调整公式为：

$$W_{adaptive}(i) = W_{min} + (W_{max} - W_{min}) \times Sigmoid[Texture_{norm}(i)]$$

其中，$Sigmoid$函数在生物学中被称为$Logistic$函数，由数学家皮埃尔·弗朗索瓦·热涅提出，在生态学领域中用于描述一个种群在资源有限的情况下的增长趋势，又称为 S 型生长曲线。基础的$Sigmoid$函数的表达式为：

$$Sigmoid(x) \overset{\text{def}}{=} \sigma(x) = \frac{1}{1+e^{-x}}$$

它的一个重要的性质为：

$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \sigma(x)[1-\sigma(x)]$$

这个性质反映出$Sigmoid$函数的斜率（增速）随着函数值的增加先增大后减小，因

此在对传染病传染人数的数学建模问题中可以用$Sigmoid$函数粗略地表示一个地区在无人员流动、无人口出生、死亡与治愈、刚刚爆发某种传染病的一小段时间内传染病的感染人数变化趋势。

在动物生理学中，神经冲动在神经纤维上的产生和传导过程大致为（图 4.3.5.2.3-1）：在未受到刺激时，神经纤维处于电位内负外正静息状态（极化），当神经纤维某一部位受到刺激时，膜两侧出现暂时性的电位变化（去极化），表现为内正外负的兴奋状态（反极化），此时的膜电位称为动作电位（action potential），而邻近的未兴奋部位仍然是内负外正（极化），在兴奋部位和未兴奋部位之间由于电位差的存在而发生电荷移动，这样就形成了局部电流，局部电流又刺激相近的未兴奋部位发生同样的电位变化，如此进行下去，将兴奋向前传导，后方又恢复为静息电位（复极化、超极化）。

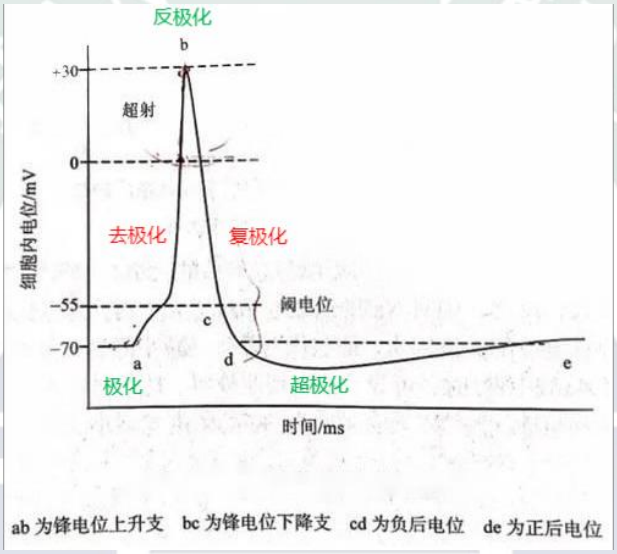


图4. 3. 5. 2. 3-1 神经纤维动作电位示意图

其中，去极化（神经元受到激活）只发生在对神经元的刺激达到一定强度时，因此可以认为神经纤维动作电位的曲线是一种激活函数。受此启发（图 4.3.5.2.3-2），在机器学习领域，$Sigmoid$函数由于其特性常被用作神经网络的激活函数，将输入值映射到 0 和 1 之间的范围（相当于对每个神经元的输出都做了归一化，Normalization），并且给出了明确的预测结果（在两端非常接近 1 或 0），因此适用于二元分类任务。在代码中，作者实际使用的函数的表达式为：

$$Sigmoid(x) = \frac{1}{1 + e^{-6(x-0.5)}}$$

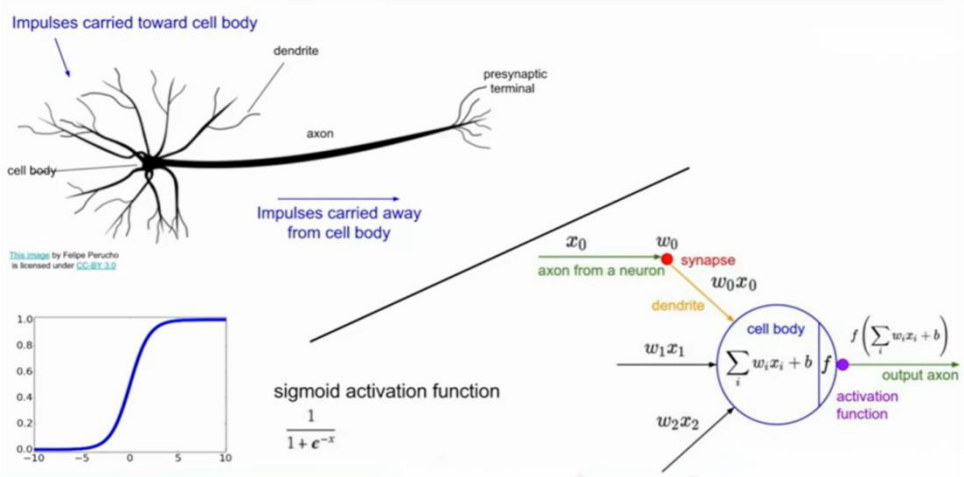


图4.3.5.2.3-2 在机器学习领域，Sigmoid函数由于其特性常被用作神经网络的激活函数

其实这种性质的曲线并不止有$Sigmoid$函数。我去年在数字图像处理课程设计中进行 PAN 和 MS 影像的加权数据融合时，为了改进权重的性质，对权重进行缓冲时用到的 tanh 函数（双曲正切函数，hyperbolic tangent function）也是一种在机器学习中常用的激活函数，具有类似的性质（图 4.3.5.2.3-3）：

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$tanh'(x) = \frac{1}{cosh^2(x)}$$

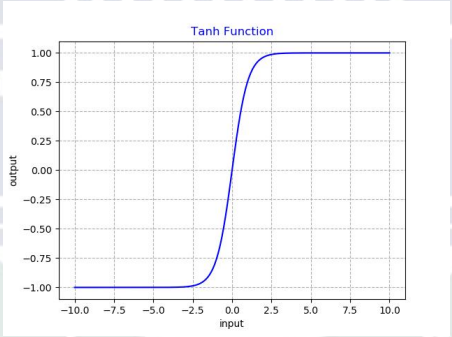


图4.3.5.2.3-3 tanh函数的图像

在本次实习中，作者在这里使用$Sigmoid$函数是为了自适应地确定羽化范围，在纹理复杂的区域使用较窄的羽化带以保持细节，在纹理平坦的区域使用较宽的羽化带以确保平滑过渡。同样地，作者在灰度值的权重上也没有使用线性的权重，而是使用平滑的余弦曲线，同样是希望灰度值的变化更加平缓，避免重叠区域的插值结果与非重叠区域有较大色差。

```
    def adaptive_blend(self, over1, over2, seam):
        """
        自适应羽化宽度 - 羽化带宽度根据纹理复杂度连续变化，并限制在重叠区域边界内
        处理像素值为 0 的情况：如果其中一个像素为 0，则使用非 0 像素值
        """
        rows, cols = over1.shape[1:]
```

```
        result = np.zeros_like(over1)

        # 根据重叠区域宽度自适应调整融合带宽度范围
        if self.overlap_width is not None:
            # 最小融合带：重叠宽度的1/10，最少5像素
            min_width = max(5, self.overlap_width // 10)
            # 最大融合带：重叠宽度的1/2
            max_width = max(min_width, self.overlap_width // 2)
            self.status_updated.emit(f"自适应羽化参数 - 最小带宽: {min_width}px,
最大带宽: {max_width}px, 重叠区域: {self.overlap_width}×{self.overlap_height}")
        else:
            # 默认值
            min_width = 3
            max_width = 20

        # 计算纹理复杂度
        def texture_complexity(img):
            gray = np.mean(img, axis=0)
            grad_x = np.abs(np.gradient(gray, axis=1))
            grad_y = np.abs(np.gradient(gray, axis=0))
            return grad_x + grad_y

        texture1 = texture_complexity(over1)
        texture2 = texture_complexity(over2)

        # 计算全局纹理统计信息用于归一化
        all_texture = np.maximum(texture1, texture2)
        texture_mean = np.mean(all_texture)
        texture_std = np.std(all_texture)

        for r in range(rows):
            c = min(seam[r], cols-1)

            # 计算局部纹理复杂度
            local_texture = max(texture1[r, c] if c < texture1.shape[1] else 0,
                                texture2[r, c] if c < texture2.shape[1] else 0)

            # 将纹理复杂度归一化到[0,1]范围
            if texture_std > 0:
                normalized_texture = max(0, min(1, (local_texture - texture_mean +
texture_std) / (2 * texture_std)))
            else:
                normalized_texture = 0.5
```

```
            # 使用平滑的S型函数将归一化纹理映射到融合带宽度
            sigmoid_input = (normalized_texture - 0.5) * 6
            sigmoid_value = 1 / (1 + np.exp(-sigmoid_input))

            # 线性插值计算融合带宽度
            blend_width = int(min_width + (max_width - min_width) * sigmoid_value)

            # 限制融合范围不能超出重叠区域边界
            start_blend = max(0, c - blend_width//2)  # 不能小于重叠区域左边界
            end_blend = min(cols, c + blend_width//2)  # 不能大于重叠区域右边界

            # 进一步确保融合范围的合理性
            if start_blend < 0:
                start_blend = 0
            if end_blend > cols:
                end_blend = cols

            # 基础分割
            result[:, r, :start_blend] = over2[:, r, :start_blend]
            result[:, r, end_blend:] = over1[:, r, end_blend:]

            # 融合区域 - 使用余弦插值实现更平滑的过渡
            if end_blend > start_blend:
                for fc in range(start_blend, end_blend):
                    t = (fc - start_blend) / (end_blend - start_blend)
                    weight = 0.5 * (1 - np.cos(np.pi * t))

                    # 修改：处理像素值为0的情况
                    pixel1 = over1[:, r, fc]
                    pixel2 = over2[:, r, fc]

                    # 检查是否有像素值为0
                    pixel1_is_zero = np.all(pixel1 == 0)
                    pixel2_is_zero = np.all(pixel2 == 0)

                    if pixel1_is_zero and not pixel2_is_zero:
                        # 如果pixel1为0，使用pixel2
                        result[:, r, fc] = pixel2
                    elif pixel2_is_zero and not pixel1_is_zero:
                        # 如果pixel2为0，使用pixel1
                        result[:, r, fc] = pixel1
                    elif pixel1_is_zero and pixel2_is_zero:
                        # 如果两个都为0，保持为0
                        result[:, r, fc] = 0
```

```
                else:
                    # 如果两个都不为0，执行正常的插值
                    result[:, r, fc] = (1 - weight) * pixel2 + weight * pixel1

        return result
```

自适应羽化融合的效果如图 4.3.5.2.3-4 所示：

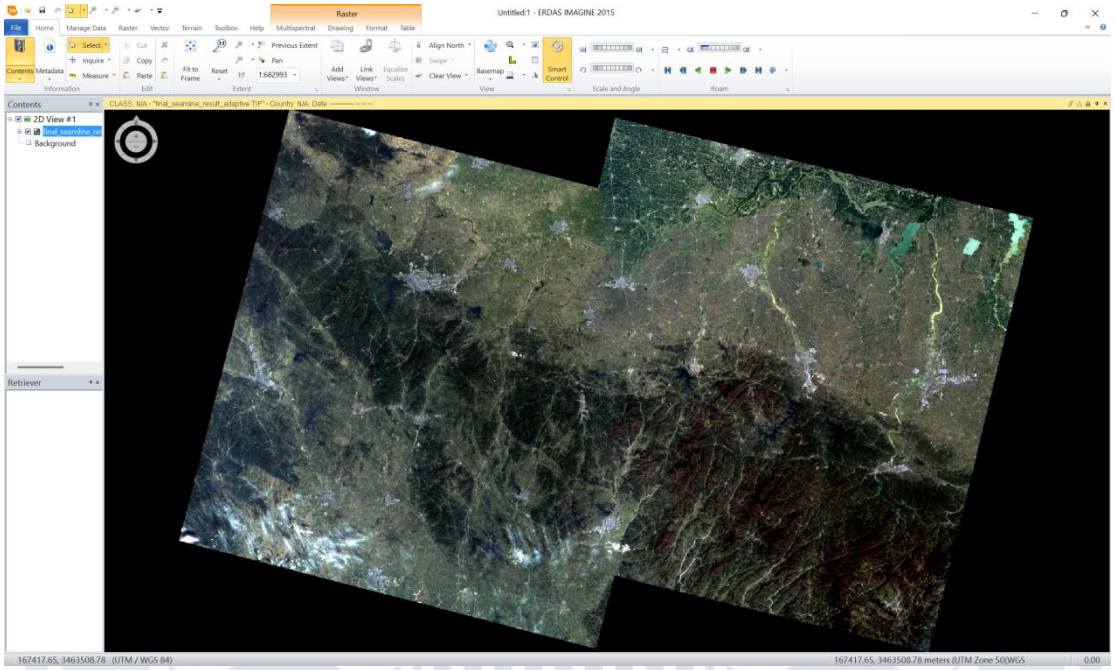


图4. 3. 5. 2. 3-4 自适应羽化融合的效果

从图 4.3.5.2.3-4 中几乎已经完全看不出镶嵌线的痕迹了，这种融合方法实际上是应用镶嵌线进行拼接 + 不应用镶嵌线进行像素值处理的方法的结合，既具有像素值融合的平滑优点又避免了其因平滑带来的“鬼影”缺点，取得了较好的影响融合的效果。

4.3.5.3 可视化 UI 界面设计

在用户交互方面，为了便于调参并**组合使用各种优化方法**，我还开发了**基于 PyQt5 的图形用户界面**，提供了丰富的参数配置选项和实时的处理进度显示。用户可以根据具体需求**调整想要使用的镶嵌线镶嵌算法以及相关参数，灵活地组合使用各种优化策略**，以获得最佳的镶嵌效果。

程序运行后，打开的 Windows 窗体可视化界面如图 4.3.5.3-1 所示：

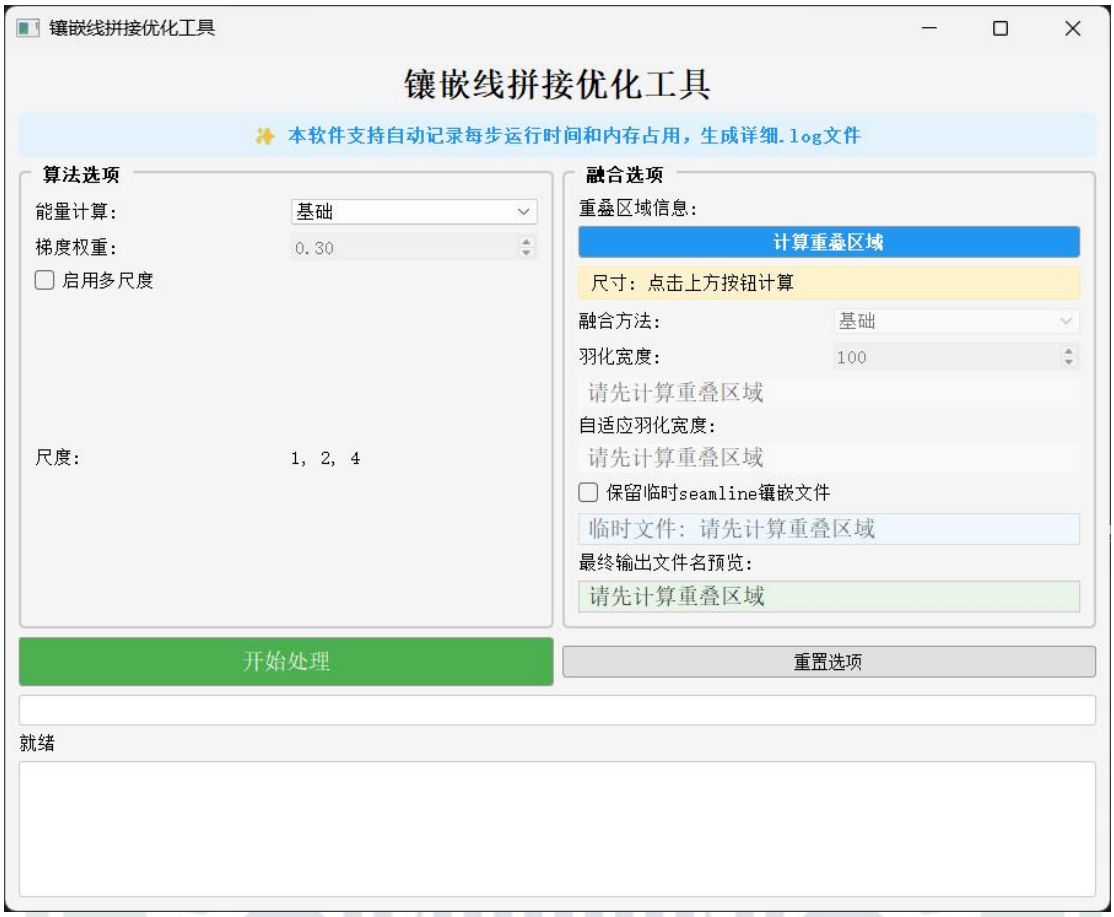


图4.3.5.3-1 初始的程序界面

首先需要点击右上角的“计算重叠区域”按钮，才能进行后续处理（图 4.3.5.3-2）。由于在程序启动时已经从 TIF 中读取影像的地理坐标信息，计算重叠区域几乎不用花费多少时间。这样设计是为了先得到重叠区域的范围，其尺寸大小将用于计算自适应羽化宽度中的羽化半径上限与下限、计算羽化宽度允许设置的上限与下限等。

图4.3.5.3-2 点击右上角的“计算重叠区域”按钮后才能进行后续处理

可供选择的配置包括：是否弃用梯度增强的能量计算方法，以及梯度在能量计算中所占的权重；是否启用多尺度优化寻找镶嵌线的方法；是否选择羽化后处理，以及设置羽化半径；是否使用自适应羽化半径进行羽化后处理；等等。设置好想要使用的镶嵌线镶嵌算法的配置后，点击“开始处理”，在下方的日志信息栏中可以看到当前执行步骤的提示信息（图 4.3.5.3-3）。等待处理完毕后即可在结果文件夹中找到处理结果 TIF 文件。

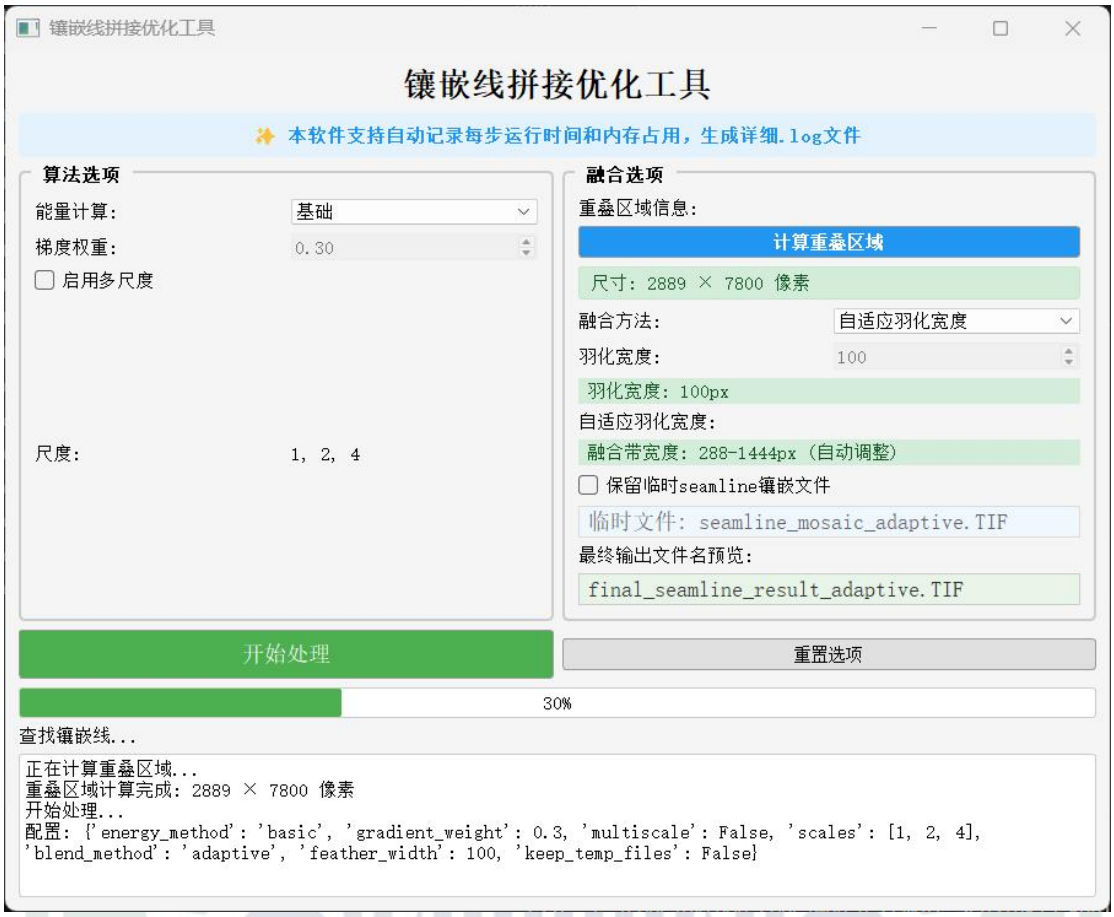


图4.3.5.3-3 在进行镶嵌线镶嵌的过程中，在下方的日志栏中会给出程序运行状态的信息

此外，本系统还支持输出.log 文件（图 4.3.5.3-4），记录每次进行镶嵌线算法的处理时的每一步的运行时间和内存占用信息，以供进行后续性能分析。

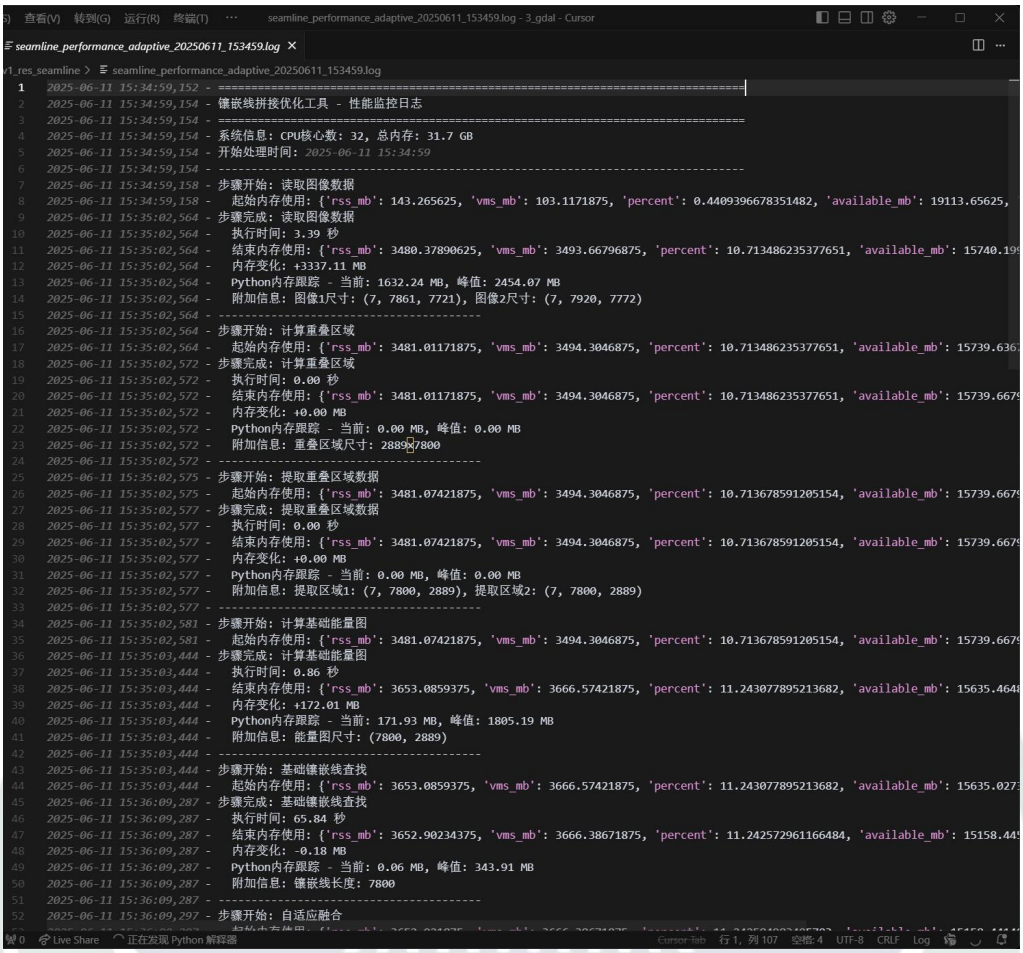


图4.3.5.3-4 输出的.log文件，记录了每一步的运行时间和内存占用信息

4.3.5.4 性能分析的具体实现方式

我在这个镶嵌线拼接优化工具中实现了一套完整的性能监控和日志记录系统。

在运行时间的监控上，本程序会记录每次执行影像镶嵌时的总体执行时间（从开始到结束的完整处理时间）、分步骤时间（每个处理步骤的具体耗时）以及时间戳记录（精确记录开始和结束时间点）；在内存占用的监控上，作者使用了两套互补的内存监控工具：系统级监控（psutil）用于监控物理内存使用（RSS）、虚拟内存使用（VMS）、内存使用百分比、系统可用内存等；Python 级监控（tracemalloc）用于跟踪 Python 对象内存分配、记录内存使用峰值、提供更精确的 Python 内存使用情况等。在程序运行时，收集 CPU 核心数量、总内存容量、操作系统信息、日志记录机制等系统信息，然后进行分步骤的性能跟踪，并进行结构化日志输出，输出的.logh 文件也支持智能文件命名。

具体来说，程序对以下关键步骤进行了详细监控：图像读取：监控 I/O 性能和内存分配；重叠区域计算，监控几何计算效率；能量图计算：监控算法复杂度对性能的影响；镶嵌线查找：监控动态规划算法性能；图像融合：监控内存密集型操作；文件输出：监控写入性能。

在本次影像镶嵌任中进行性能分析，有助于快速定位性能瓶颈，分析算法效率，识别内存泄漏问题；有助于比较不同算法配置的性能表现；有实时监控系统资源使用，预防内存溢出，评估程序运行的硬件需求等。我设计的这套性能监控系统不仅记录了基本的时间和内存信息，还包含了丰富的上下文信息，能够为性能分析和系统优化提供了强有力的数据支持。

```
# 由于性能监测的代码是分散在整个程序的代码的各处的，所以这里只展示代码片段、
# 核心监控类：PerformanceLogger
class PerformanceLogger:
    """性能监控和日志记录类"""
    def __init__(self, log_file_path):
        self.process = psutil.Process()  # 获取当前进程
        self.step_start_time = None      # 步骤开始时间
        self.step_start_memory = None    # 步骤开始内存
        tracemalloc.start()              # 启动 Python 内存跟踪
# 分步骤性能跟踪
def start_step(self, step_name):
    """开始一个处理步骤的计时和内存监控"""
    self.step_start_time = time.time()
    self.step_start_memory = self.get_memory_usage()
def end_step(self, step_name, additional_info=""):
    """结束一个处理步骤的计时和内存监控"""
    elapsed_time = time.time() - self.step_start_time
    current_memory = self.get_memory_usage()
    memory_change = current_memory['rss_mb'] - self.step_start_memory['rss_mb']
# 内存监控指标
def get_memory_usage(self):
    return {
        'rss_mb': memory_info.rss / 1024 / 1024,    # 物理内存使用(MB)
        'vms_mb': memory_info.vms / 1024 / 1024,    # 虚拟内存使用(MB)
        'percent': self.process.memory_percent(),    # 内存使用百分比
        'available_mb': virtual_memory.available / 1024 / 1024,  # 可用内存
        'total_mb': virtual_memory.total / 1024 / 1024          # 总内存
    }
```

4.4 实习结果与分析

我得到的**效果最好的程序输出的影像镶嵌图如图 4.4-1 所示**，使用的是**自适应羽化宽度的镶嵌线算法**。其余结果作为比对参考。

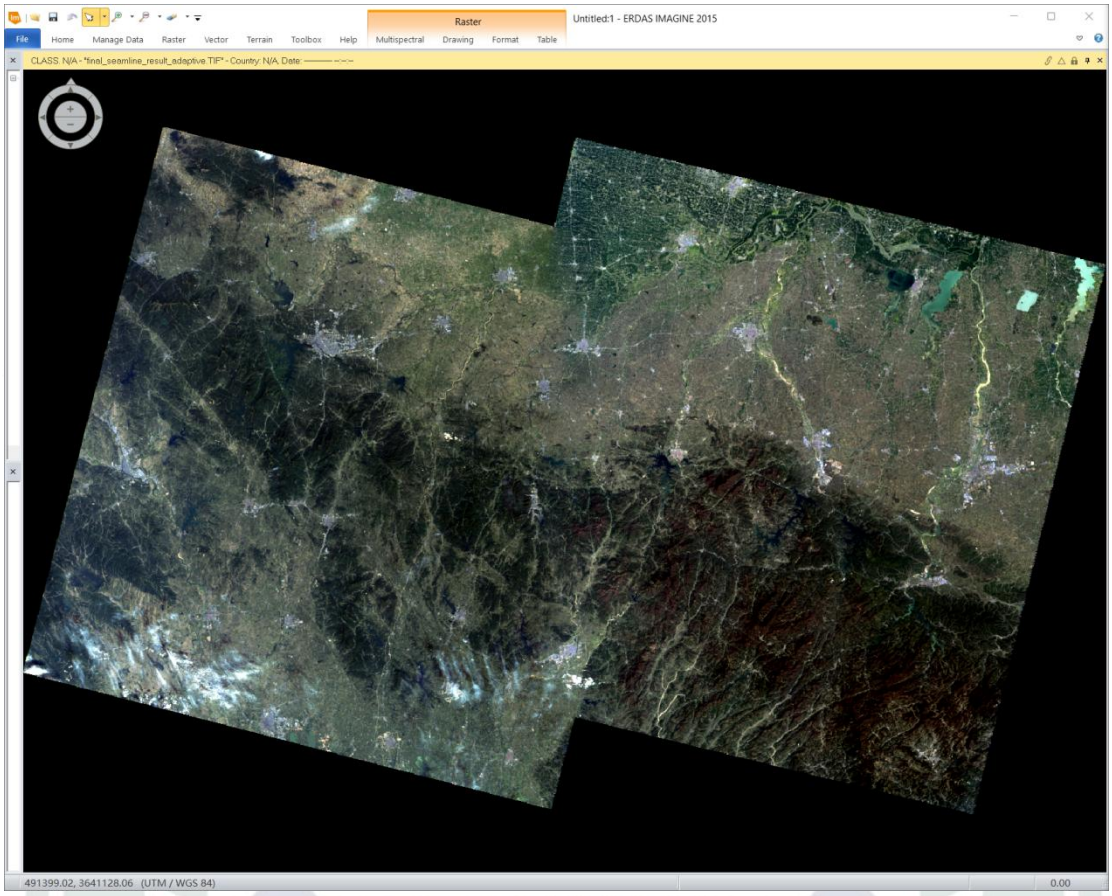


图4. 4-1 自适应羽化宽度的镶嵌线算法输出的影像镶嵌图【最优结果】

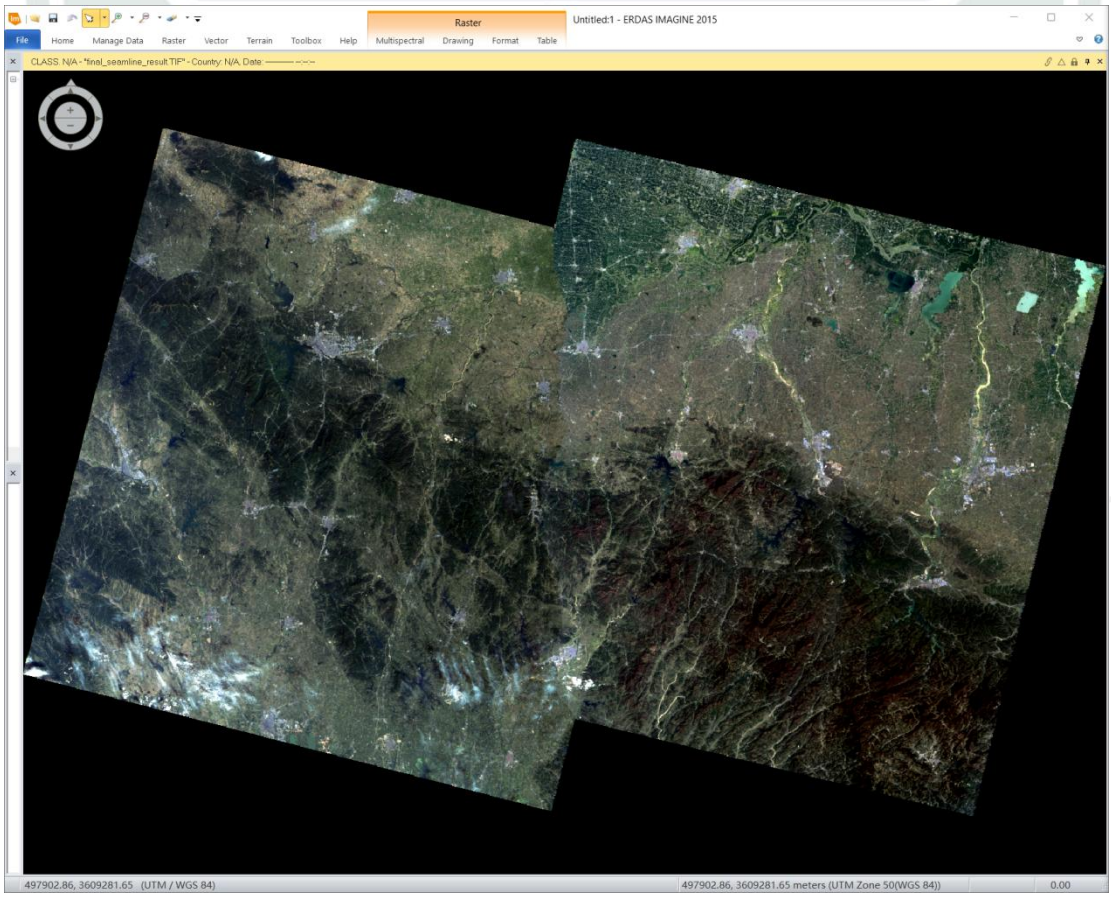


图4.4-2 基础镶嵌线算法输出的影像镶嵌图（存在明显的拼接痕迹）

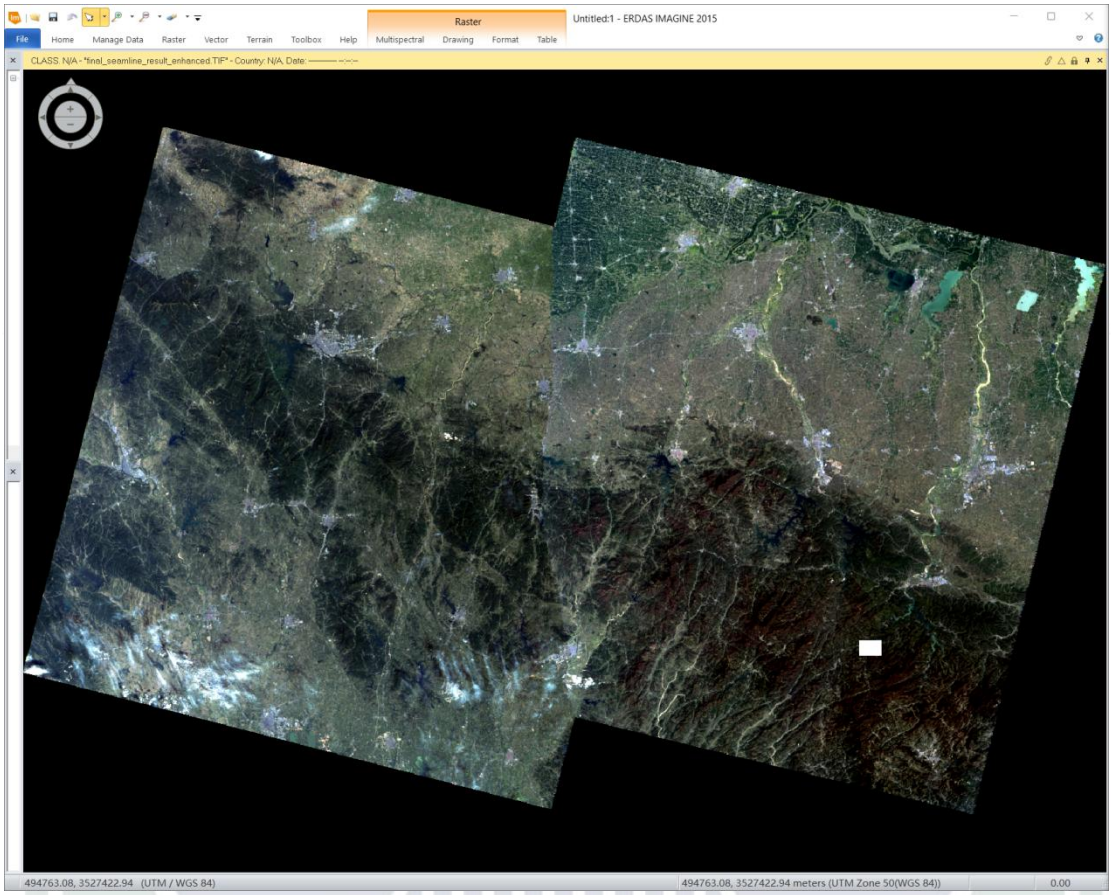


图4.4-3 梯度增强的镶嵌线算法输出的影像镶嵌图（存在明显的拼接痕迹）

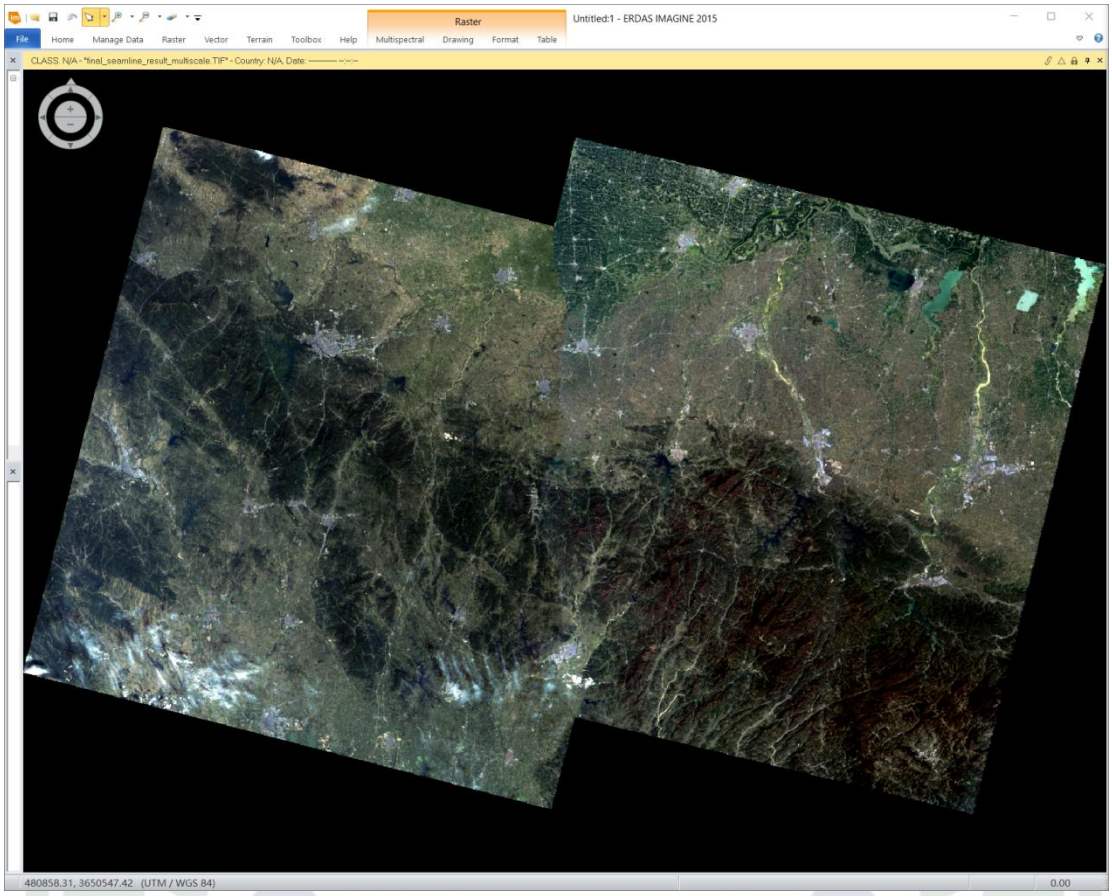


图4.4-4 多尺度优化的镶嵌线算法输出的影像镶嵌图（存在明显的拼接痕迹）

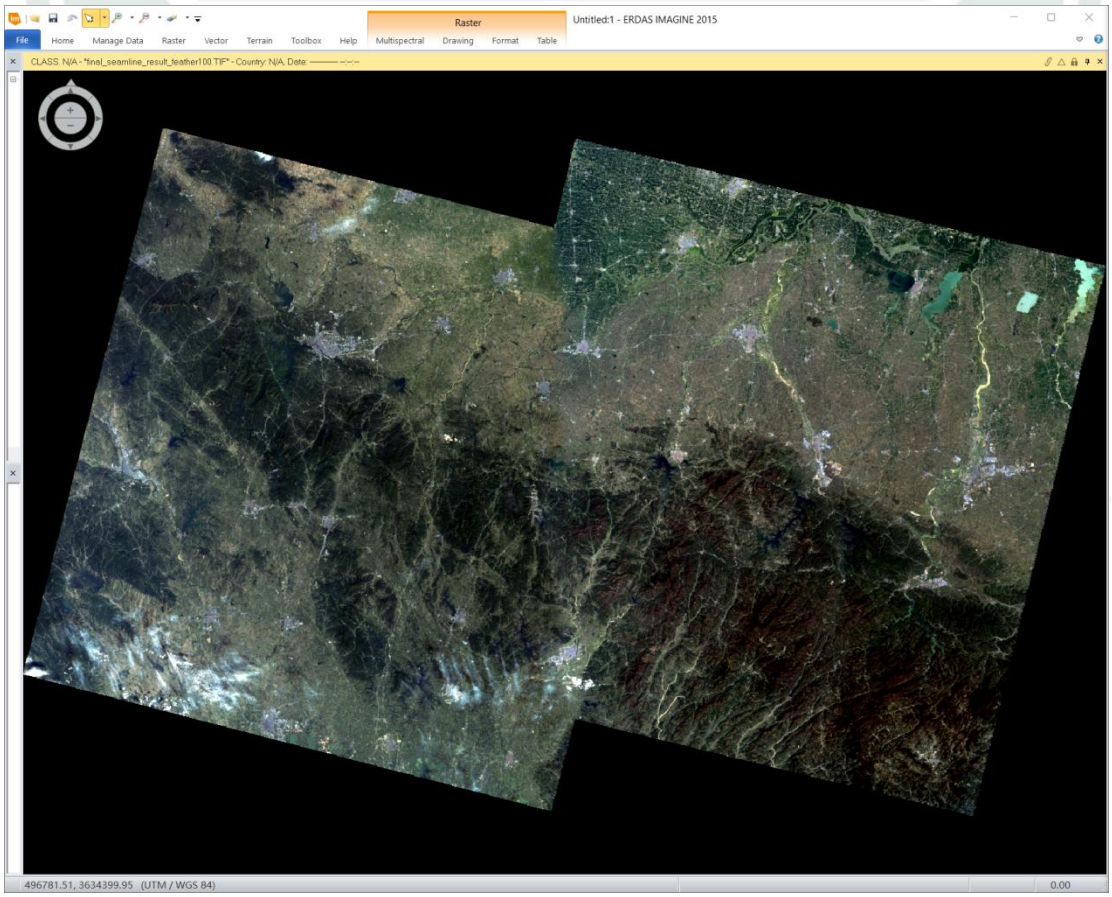


图4.4-5 静态羽化半径的镶嵌线算法输出的影响镶嵌图（羽化半径：100）（仍存在一定的拼接痕迹）

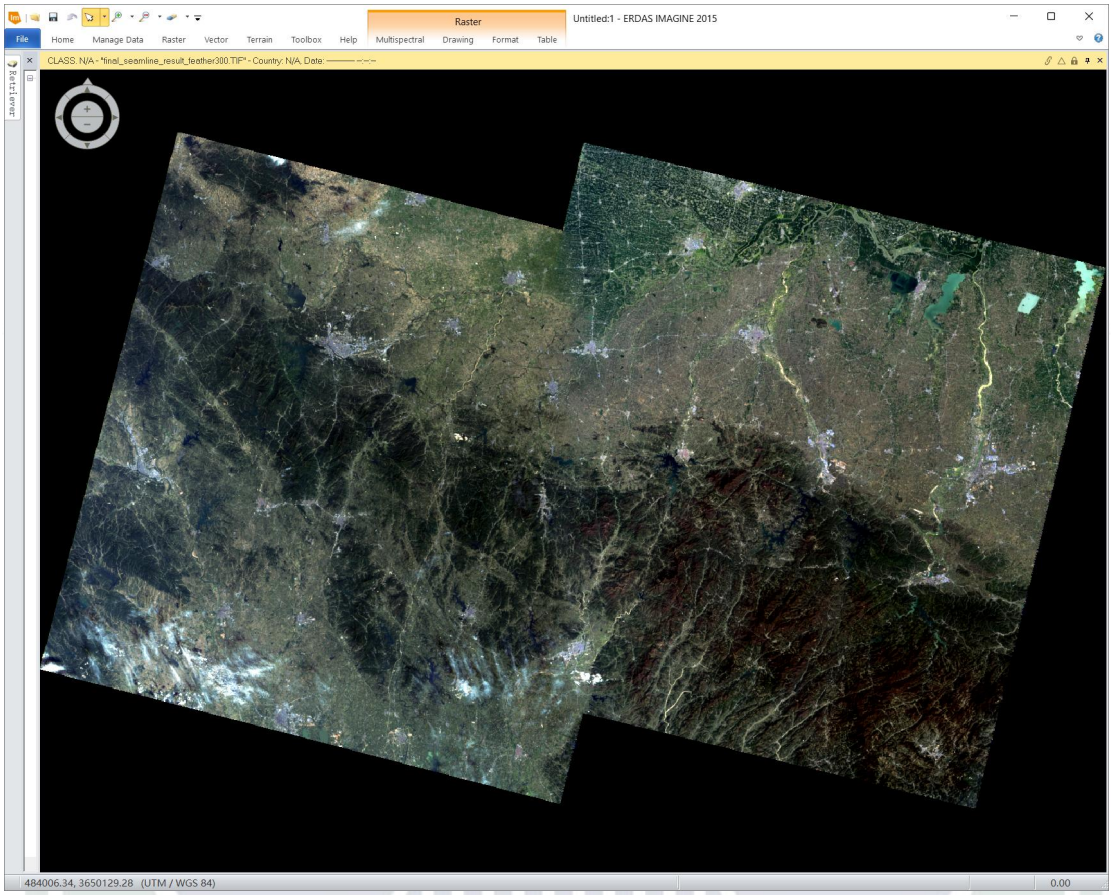


图4.4-6 羽化融合的效果（羽化半径为300）（仍存在一定的拼接痕迹，但相对于图4.4-5进一步改善）

此外，针对 4.3.5.4 实现的性能分析，在此也展示运行时间与内存占用的监测结果：

镶嵌边镶嵌的后处理方法	读取图像数据	计算重叠区域	计算基础能量图	羽化融合处理	提取重叠区域数据	基础镶嵌线查找	叠置操作	写出临时seamline结果
半径为50的羽化处理	3.00	0	0.88	13.78	0	65.68	22.55	2.74
半径为100的羽化处理	3.47	0	0.87	26.91	0	65.23	23.24	2.92
半径为150的羽化处理	2.88	0	0.83	40.99	0	65.66	24.43	2.93
半径为200的羽化处理	3.07	0	1.06	57.82	0.01	64.30	24.67	2.86
半径为250的羽化处理	2.85	0	0.70	63.87	0	67.35	22.12	2.83
半径为300的羽化处理	2.81	0	0.81	78.38	0	62.86	23.23	2.88

自适应半径的羽化处理	3.39	0	0.86	121.45	0	65.84	22.54	2.89

表4.4-1 运行时间的监测结果（单位：秒）

镶嵌边镶嵌的后处理方法	读取图像数据	计算重叠区域	计算基础能量图	羽化融合处理	提取重叠区域数据	基础镶嵌线查找	叠置操作	写出临时seamline结果
半径为50的羽化处理	1632.24	0	171.93	300.87	0	0.06	0.01	0.01
半径为100的羽化处理	1632.24	0	171.93	300.87	0	0.06	0.01	0.01
半径为150的羽化处理	1632.24	0	171.93	300.87	0	0.06	0.01	0.01
半径为200的羽化处理	1632.24	0	171.93	300.87	0	0.06	0.01	0.01
半径为250的羽化处理	1632.21	0	171.92	300.86	0	0.06	0	0
半径为300的羽化处理	1632.21	0	171.92	300.86	0	0.06	0	0
自适应半径的羽化处理	1632.24	0	171.93	816.64	0	0.06	0.01	0.01

表4.4-2 内存占用的监测结果（单位：MB）

结果分析：在本次实习中，我通过应用 Python GDAL 库编程实现遥感影像的镶嵌，得到了多种不同算法优化后的镶嵌结果。其中，最优结果是使用自适应羽化宽度的镶嵌线算法得到的影像镶嵌图（如图 4.4-1 所示）。从图中可以看出，影像镶嵌后的整体效果较为理想，两景影像之间的拼接过渡自然流畅，几乎难以看出使用的镶嵌线在哪里，也很难察觉到拼接痕迹，很好地实现了无缝拼接的目标，为后续的遥感影像分析和应用提供了高质量的基础数据。

不同**算法优化效果对比分析**：（1）基础镶嵌线算法：如图 4.4-2 所示，采用基础镶嵌线算法得到的影像镶嵌图存在较为明显的拼接痕迹。这是因为在基础镶嵌线融合过程中，直接在镶嵌线的左侧使用第一幅影像的像素值，右侧使用第二幅影像的像素值，属于硬切换策略，没有对重叠区域的像素值进行平滑过渡处理，导致在镶嵌线处影像的亮度、色彩等特征存在突变，从而产生明显的拼接痕迹[40]；（2）梯度增强的镶嵌线算法：如图 4.4-3 所示，该算法在基础镶嵌线算法的基础上引入了梯度信息，虽然在一

定程度上优化了镶嵌线的走向，使其更加平顺，但在拼接痕迹的消除方面并没有取得显著的改善，仍然存在明显的拼接痕迹。这说明仅靠优化镶嵌线的走向，并不能完全解决拼接痕迹的问题，还需要对重叠区域的像素值进行进一步的融合处理。（3）多尺度优化的镶嵌线算法：如图 4.4 - 4 所示，通过多尺度优化策略得到的镶嵌线效果在拼接痕迹的消除上也没有取得突破。多尺度优化策略主要是在镶嵌线搜索过程中提高了算法的计算效率和全局最优性[41]，但同样没有对重叠区域的像素值进行平滑过渡处理，因此拼接痕迹依然明显。这进一步说明了在镶嵌线搜索优化的基础上，还需要结合其他像素值融合技术才能更好地消除拼接痕迹；（4）静态羽化半径的镶嵌线算法：如图 4.4 - 5 所示，采用静态羽化半径的镶嵌线算法后，拼接痕迹得到了一定程度的改善，但仍然能够隐约看出拼接的痕迹。这是因为静态羽化融合虽然通过在镶嵌线附近建立过渡带来实现平滑拼接，但由于羽化宽度是固定的，在纹理复杂度不同的区域，固定的羽化宽度可能无法很好地适应，导致在某些区域拼接痕迹仍然较为明显。例如在纹理复杂的区域，固定的羽化宽度可能过窄，无法充分平滑过渡；而在纹理平坦的区域，固定的羽化宽度可能过宽，可能会引入一些不必要的模糊；（5）自适应羽化宽度的镶嵌线算法：如图 4.4 - 1 所示，该算法是本次实习中效果最好的。它根据局部纹理复杂度动态调整羽化宽度，在纹理复杂的区域使用较窄的羽化带以保持细节，在纹理平坦的区域使用较宽的羽化带以确保平滑过渡[42]。这种自适应的羽化策略能够很好地适应不同纹理特征的区域，从而最大程度地消除了拼接痕迹，实现了高质量的无缝拼接效果。同时，它还采用了平滑的余弦曲线作为灰度值的权重函数，进一步确保了灰度值变化的平缓性，避免了重叠区域的插值结果与非重叠区域有较大色差，进一步提升了镶嵌质量。

由于所用的TIF格式的文件的像素数量达到了亿级，在这里就不得不提**算法性能**了。从表 4 . 4 - 1 显示的**运行时间**来看，根据所有镶嵌线算法在运行时的耗时都是较长的，其中羽化融合处理是最耗时的环节，随着羽化半径的增加，处理时间呈线性增长趋势（图4.4-7），从半径 50 的 13.78 秒增长到半径 300 的 78.38 秒，而自适应羽化处理由于需要计算局部纹理复杂度，耗时达到 121.45 秒，成为整个流程中的性能瓶颈。其余的操作步骤在不同的镶嵌线镶嵌后处理方法之间的运行时间差异不大，无明显规律。叠置操作也是主要的时间消耗点，平均耗时约 65 秒，这主要是由于需要处理大量的像素数据重投影和重采样操作。从表 4 . 4 - 2 显示的**内存占用**来看，读取图像数据阶段的内存占用最高，达到 1632MB 左右，这是因为需要同时将两景多波段影像完整加载到内存中；羽化融合处理的内存占用约为 300MB，而自适应羽化由于需要额外计算和存储纹理复杂度信息，内存占用增加到 816MB，是静态羽化的 2.7 倍。其他处理步骤的内存占用相对较小，均在 200MB 以下，说明算法的内存瓶颈主要集中在数据加载和复杂融合计算阶段。但是不同的镶嵌线镶嵌后处理算法之间的内存占用差异不明显。

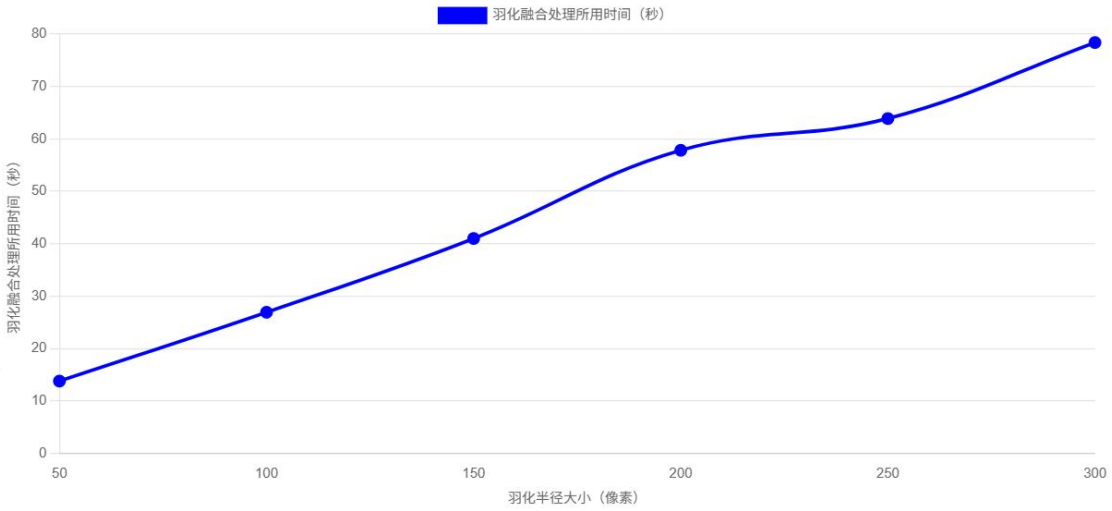


图4.4-7 羽化融合处理所用时间与羽化半径大小之间的关系

本节实习所实现的遥感影像镶嵌技术具有广泛的应用前景。随着遥感技术的不断发展，获取的遥感影像数据越来越多，且数据量越来越大。通过高效的镶嵌技术，可以将多幅具有重叠区域的遥感影像拼接成一幅完整、连续的影像，为大范围的地理信息获取、环境监测、资源调查等提供更全面、更准确的数据支持。例如，在城市规划中，可以将不同区域的遥感影像镶嵌成一幅完整的城市影像图，用于城市土地利用分类与监测[43]、城市扩张监测等；在农业领域，可以将大面积的农田遥感影像镶嵌起来，用于农作物种植面积估算、病虫害监测等。

尽管本部分的实习在影像镶嵌任务上取得了一定的效果，但仍然存在一些局限性。目前的镶嵌算法主要针对的是两景存在重叠区域的不同时相影像数据，并且还不能完全消除两幅影像之间的色差的影响，对于多时相、多源的遥感影像镶嵌，算法的适应性和鲁棒性还需要进一步地作出适应性调整和优化。其次，在处理复杂地形条件下的遥感影像镶嵌时，由于地形起伏造成的阴影效应等因素可能会对镶嵌结果产生一定的影响，目前的算法在消除这些影响方面还不够完善。

未来，可以进一步研究更加智能化、自动化的镶嵌算法，自动识别和处理多时相、多源遥感影像的复杂特征，提高镶嵌的效率和质量；针对复杂地形条件下的遥感影像镶嵌，进一步优化几何校正和辐射校正算法，提高校正精度，有效消除地形起伏等因素对镶嵌结果的影响；探索云雾遮挡区域信息补偿的有效方法，提高遥感影像镶嵌在复杂环境下的适用性；结合人工智能、机器学习等前沿技术，开发更加高效、精准的遥感影像镶嵌模型[44]，为遥感应用提供更加高质量的基础数据产品；等等。

4.5 实习过程中遇到的问题及其解决方法

遇到的问题	解决方法
进行直方图匹配后，得到的图带有“黑边”存在于	我分析后认为，这种情况的出现是因为

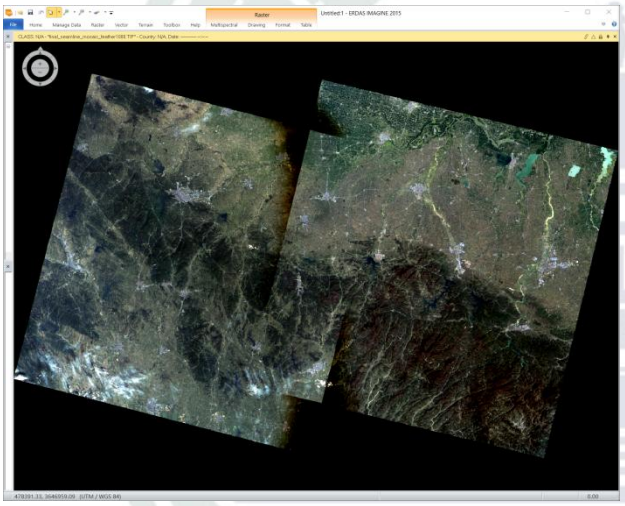
影像的四个角，在进行覆盖镶嵌时这些黑色的区域会错误地覆盖在那一张被覆盖的影像上面	**我所使用的遥感影像的有效区域是倾斜的**，因此保存遥感影像的外接矩形时会把那四个角的黑边带上，从而导致覆盖的时候出现多余的黑边，经过特殊处理后（设置为no-data或者完全透明）这个问题能够得到解决
处理超大影像时出现内存不足的报错（如下图所示）  	查找资料后得知，需要在输出文件创建时添加BIGTIFF选项，以优化数组处理减少内存占用。**作者为此写了一篇CSDN来介绍此报错的具体解决方法**：https://blog.csdn.net/Zlyzjiabjw547479/article/details/148463460
羽化镶嵌线算法得到的镶嵌后影像中，存在部分区域呈现黑色的问题（如下图所示） 	经过对重叠区域镶嵌结果的中TIF文件的检查以及对代码的检查，我发现是有的影像外（但是仍有像素，只不过灰度值为0）的像素导致的，于是在羽化的加权函数中设定：**如果有一个像素值为0，那么加权结果直接为另一个非0的像素值**，避免0参与到加权计算中

表4.5-1 在应用Python GDAL库编程实现遥感影像的镶嵌的实习过程中遇到的问题及其解决方法

4.6 实习小结

通过本次应用 Python GDAL 库编程实现遥感影像镶嵌的实习，我系统地掌握了遥感影像镶嵌的完整技术流程，深入理解了相关理论原理，并取得了显著的实践成果。

通过本阶段的实习，我全面掌握了遥感影像镶嵌的核心理论基础，包括几何校正与配准原理、辐射校正与色彩平衡原理、重叠区域处理的融合算法等。通过深入学习累积分布函数变换、动态规划算法、能量图构建等数学方法，我对遥感数据处理的理论体系有了更加深刻的认识。

在编程实现方面，我成功实现了从波段叠加处理到最终镶嵌结果输出的完整技术流程。通过编程实现直方图匹配、覆盖镶嵌、镶嵌线镶嵌等关键算法，我熟练掌握了 Python GDAL 库的应用方法，具备了独立开发遥感数据处理工具的技术能力。特别是在镶嵌线

算法的优化探索中，我实现了梯度增强、多尺度优化、自适应羽化融合等多种先进算法，展现了较强的算法设计和优化能力。此外，我还开发设计了基于 PyQt5 的可视化用户界面，实现了完整的性能监控和日志记录系统，培养了软件工程思维和系统设计能力。通过对不同算法的性能分析和效果对比，我学会了科学的实验设计和结果评估方法。

在问题解决方面，我在实习过程中遇到了内存不足、影像黑边处理、像素值为零的特殊情况处理等多个技术难题，通过查阅资料、分析问题本质、设计解决方案等方式，成功解决了这些实际问题，提升了独立解决复杂技术问题的能力。

通过算法效果对比分析，我深刻认识到不同镶嵌算法的适用场景和局限性。自适应羽化宽度的镶嵌线算法取得了最佳的镶嵌效果，基本上能够消除拼接痕迹，实现了高质量的影像镶嵌。这一成果不仅验证了我所进行的算法优化的有效性，也为实际遥感应用提供了一种切实可行的技术解决方案。

本部分实习使我对遥感数据处理技术有了更加全面而深入的理解，培养了扎实的编程能力和工程实践能力，为今后从事遥感应用开发和科学研究奠定了坚实的动手实践能力的基础。

5 实习总结与感想

5.1 实习总结

本次遥感原理与应用课程设计实习让我全面系统地掌握了遥感数据处理的核心技术流程，从理论认知到实践操作都获得了显著提升。通过应用 ERDAS 软件进行遥感分类专题信息提取与制图、应用 OGE 平台的基于特征指数遥感专题信息提取和应用 Python GDAL 库编程实现遥感影像的镶嵌三个部分的实践，我对遥感技术的应用价值有了更深刻的理解。

在应用 ERDAS 软件进行遥感分类专题信息提取与制图部分中，我完整实现了从数据预处理到专题制图的全流程，在这个过程中我认识到了遥感工程化处理的严谨性。在几何校正环节，通过控制点选取与多项式模型构建，我深刻体会到遥感数据几何精度的关键性，严格的几何校正误差控制为后续的影像镶嵌等步骤奠定了可靠的基础，最终我选择的平行六面体法的监督分类方法达到的 91.11%的总体精度验证了特征空间划分的有效性，保证了我所制作的。

应用 OGE 平台的基于特征指数遥感专题信息提取的探索让我领略到云计算赋能遥感分析的革命性变革。通过编写特征指数计算脚本，我发现 NDVI 等指数在云平台可实现分钟级的全球尺度计算，这种时空大数据处理能力彻底改变了传统单机作业模式。特别是在水体提取任务中，AWEI 指数通过多波段协同计算，有效解决了阴影干扰这个经

典难题。

应用 Python GDAL 库编程实现遥感影像的镶嵌这一部分的实习任务最具挑战性：从动态规划算法实现镶嵌线搜索，到引入 Sigmoid 函数构建自适应羽化模型，让我将数字图像处理理论与工程实践深度结合。受到启发而进行的多尺度优化策略的尝试很有意义，这种多分辨率分析方法在后续研究中具有推广价值。

这次实习重塑了我的研究探索的思维方式：通过精度评价矩阵的定量分析，培养了数据驱动的决策能力；在解决几何校正控制点不足的问题时，锻炼了创造性思维；撰写技术报告的过程，提升了将复杂流程转化为系统文档的学术表达能力……同时，实践让我认识到遥感技术的广阔前景：在智慧城市领域，建筑指数与 NDVI 的结合可精准监测绿地率；在环境保护中，时序水体指数能有效追踪湿地退化；在灾害应急时，多源数据快速融合为决策提供关键支持。这些应用场景彰显了遥感作为空间信息基础设施的战略价值。

总的来说，本次实习不仅巩固了专业知识，更培养了应用遥感科学与技术的理论知识动手解决复杂的实际问题的系统工程思维。未来，我将持续深耕遥感技术，努力为遥感科学与技术领域的研究贡献自己的理论。这段实践经历积累的方法论与经验，必将成为我以后学习与研究工作中的宝贵财富。

5.2 实习感想

话不多说，先上图（图 5.2-1）：

图5. 2-1

完成这篇实习报告时，正是我的 20 岁生日。古制，男子 20 岁当行冠礼。古代皇帝加冠后即亲政，而我虽无九五之尊，决定不了这四海六合之事，但能够成为自己命运的主人。要真正把握自己的命运，就要走好当下的每一步，例如认认真真写好这份遥感原理与应用课程设计的实习报告。

在本次实习的最后，我想用一种特别的方式表达自己的感想：把下面的文字献给遥感科学与技术，献给自己过去的 20 载春秋冬夏，也献给自己与人类的未来：

天 问

【公元前 304 年，汉水之阳[①]】

“遂古之初，谁传道之？上下未形，何由考之？”

我独立于苍茫江畔，目光凝滞在滔滔江水之上，心中似有惊涛骇浪翻涌不息。

我，作为楚之同姓[②]，自幼便怀揣着对楚国的一片赤诚，一心只为故土的繁荣昌盛，满心以为凭自身才学，定能辅佐楚王以成帝业。我无数次在朝堂之上慷慨陈词，无数次为楚国的内政外交出谋划策，无数次满心期待着能换来楚国的长治久安……但是……

但是！现实却如同一把冰冷的利刃，无情地刺痛了我。那些奸佞小人嫉妒我的才能，便在楚王面前谗言诋毁。我曾以为君王是明君，能明辨是非，洞察奸佞，谁知楚王竟如此昏庸，将我逐出郢都。

江水滔滔不绝，见证了楚国的兴衰荣辱，见证了多少英雄豪杰的壮志未酬。楚国的朝堂，何时变得如此黑暗？楚王的心，何时变得如此糊涂？

“遂古之初，谁传道之？”这叩问还在我齿间发烫，郢都的谗言却已冻成匕首。他们笑我痴狂，可曾见九嶷山巅的云？那云里藏着颛顼的叹息——你们用黍稷占卜，我却用内心的伤口丈量天意。

看啊，上下未形的混沌在掌纹里翻涌！每道裂痕都是天阶：帝阍闭门时，我以离骚为炬；女岐无合时，我以香草为牒。你们放逐的不过是具形骸，我的魂魄早循着彗星之尾，去诘问太初的黑暗——所谓天道，原是楚王袖口漏下的碎玉；所谓占卜，不过把蝼蚁困在龟甲纹路里。但我的诘问是凿子，要劈开这包着蜜糖的牢笼！

今日你们折断我的芰荷，可曾见我屈子血里有着游龙？这放逐路多好，正好作我用楚辞丈量大地的琴弦！当我在汉北的星空下仰首，苍穹的答案，都会在我的追问中回响！

① 周赧王十一年（公元前 304 年），楚怀王将屈原放逐到汉北（《楚辞·九章·抽思》：“有鸟自南兮，来集汉北”），山南水北为阳。屈原“虽放流，眷顾楚国，系心怀王，不忘欲反（《史记·屈原贾生列传》）”，在放逐期间写出了著名长诗《离骚》《天问》等具有极高文学、历史价值的作品并流传后世。

② 《史记·楚世家》：“楚之先祖出自帝颛顼高阳。……高阳生……陆终生子六人，……六曰季连，芈姓，楚其后也。”芈姓是楚国君主之姓。屈氏始于春秋时期楚国国君楚武王之子芈瑕，受封于屈（今湖北秭归），子孙后代遂以封地名“屈”为氏。屈、景、昭三氏（后改氏为姓）都出自于芈姓，故称“楚之同姓（与楚国君主同姓）”。屈原在被流放前的官职是“三闾大夫”，“掌王族三姓，曰昭、屈、景”，意思是掌管楚国王族屈、景、昭三姓宗族事务之官。

我当然可以不必目睹你们的苟且，隐于楚地山泽，可我怎能坐视狼狗之辈横行于大楚朝堂，怎能眼睁睁看着大楚沦为秦人铁骑蹄下的废墟？我握紧手中的剑暗暗发誓，要用楚辞写下你们的黑暗，写下我对楚国的忠诚，更要让后世知道，在这楚国的历史上，曾有一个叫屈原的人。

深吸一口气，我迈开脚步，行吟泽畔。江风呼啸而过，吹乱了我的发丝，却吹不散我心中的信念。我知道，后世终将有人能够不惧天命，拥有对答上天的音量。

【公元 2025 年，荆楚江汉，武珞路北，珞珈山南】

“青天有月来几时？我今停杯一问之。”

离考试结束还有最后五分钟，他停下了手中的笔，疲惫的双眼望向窗外，看见还未褪尽的夕阳余晖染红了遥感院楼的西墙，华灯初上，那轮玉盘高悬于外面街道口的高楼旁[①]。

他看着名词解释第一题“遥感的概念”，对着自己写下的答案在心里默念道：

“遥感是使用安放在承载工具（平台）的某种装置（传感器），在不直接接触被研究的目标情况下，感测目标的特征信息（电磁波的反射辐射或者发射辐射），经过传输、处理，从中提取人们感兴趣的信息的过程。”

他突然意识到，所学的遥感技术，正是古往今来的人们所向往的“与天对话的力量”。遥感借“天眼”从遥远的太空或高空俯瞰大地，像是一双能够穿透时空的眼睛，让人类得以上天、探天、问天，解答那些曾经只能存在于想象中的疑问；而每一个 TIF 中每一个像元的 DN 值，都是来自天空的答案。

遥感的力量，正是前人所期望的能够洞察世间万象的力量，正是古代传说中能够“问天”的力量。你的目光仿佛回到了三千年前，楚地的祭司是不是正在用铜镜捕捉日影，以玉琮丈量星辉？遥感器在太空轨道运行的轨迹，多像屈原笔下“驷玉虬以乘鹥”正在天际巡游！你瞥见最后一个论述大题“遥感在考古与保护中华优秀传统文化中应用”后面的空白，竟浮现楚先王庙的壁画，凤鸟的尾羽正化作像素阵列，而屈原的诘问正以二进制的方式在光谱中重生。

他深知，自己肩负着传承和发展的使命，不仅要学好专业知识，更要将遥感技术运用到实际中，为解决人类面临的诸多问题贡献智慧和力量。

考试结束的铃声响起，他合上试卷，手中的试卷标题《遥感原理与应用》的大字却从此印在了他的心中。走出考场，数千年人的心声顿时顿时涌现在他的心头——他们当年所向往的与天对话的力量，如今终于在这一代实现。

他迈着坚定的步伐，朝着珞珈山的方向，亚热带常绿阔叶林里的樟树在晚风中簌簌作响。两千三百年前的同一种风，曾翻动过汉北的芦苇，把《天问》的字节吹进云梦泽的波纹。此刻计算机模拟的电磁波，正穿过竹简上的刻痕，与青铜器上蚀刻的雷纹发生

① 李白《古朗月行》：“小时不识月，呼作白玉盘。”

共振。他知道，遥感技术就像一座桥梁，连接着过去与未来，连接着人类对未知的探索和对美好生活的向往。而他，将在这条道路上继续前行，续写着人类与天对话的力量……

【百年之后】

“是别有人间，那边才见，光影东头？”

医院的病床上，你静静地躺着。

周围的那些人神情肃穆，你知道自己的一生走到了尽头。

你慢慢地合上了眼。

周围人的哭声你已经听不清了，你渐渐感知不到自己的呼吸，感知不到自己身体的重量，似乎周围的一切都只剩下自己的意识。

依照惯例，死前的三秒，你的大脑要走马灯式地叙述你这一生的画面，一张一张地翻过。

18 岁，你的高考分数 653，你在电脑上的志愿表里用颤抖的双手填了志愿。武汉大学，很不错的大学。武汉大学 08 组遥感科学与技术，很不错的专业。父亲看着电脑屏幕问你：“想清楚了？”你点点头。

接下来的四年，你在普通测量学作业的草稿纸上写满了导线的计算公式；在深夜室友打游戏的背景音中盯着学习通里新布置的“数字图像处理”的作业陷入沉思；《计算机图形学》课程的教授拿激光笔指了指 PPT 上的光照模型：“期末必考”；图书馆的走廊里，你在考前反复背着《遥感原理与应用》，书页边缘已经卷起；20 岁那年，你在图书馆写《遥感原理与应用课程设计》的实习报告时，不知为何想起了屈原的《天问》。……

接下来的几年，你一边听实验室里的师兄师姐讨论现在搞什么好发论文，一边盯着工位显示器修改下周组会汇报的 PPT，手边的咖啡早已凉透；你一大早从床上爬起来从终端打开 nvitop，鼠标滚轮滑动间，看见自己赶 rebuttal 的 DDL 用的那些卡终于全部跑完才大松一口气……

接下来的几十年如一日，你在清晨的薄雾中调试着无人机，冰凉的金属支架沾着露水；在戈壁滩的烈日下戴着草帽核对地面控制点坐标，防风镜上落满细碎的沙粒；你在青海湖遥感监测站的铁皮屋顶下记录着水位数据，便携式光谱仪的充电提示音与藏羚羊的鸣叫此起彼伏，防风面罩上结着细密的冰晶；某个暴雪夜，你裹着军大衣给年轻的研究员讲解 MODIS 影像上的热异常点，平板电脑的蓝光映着墙上泛黄的《楚辞》复印页——那是三十年前毕业旅行时从秭归带回的纪念；在台风过境后第一时间分析卫星传回的洪涝淹没范围图，办公室里泡面的蒸汽模糊了显示屏，当风云四号传来的实时云图在指挥中心大屏亮起，你突然想起十八岁那个填志愿的下午，父亲茶杯里浮沉的龙井茶叶，像极了此刻台风眼周围旋转着的卷积云。

某个加班的深夜，你突然发现当年《遥感原理与应用》教材里夹着的樱花书签——那是大二的那个春天在教五楼前做的，粉白的花瓣早已脆化成半透明的蝶翼。你想起第

一次在《测绘学概论》课程上看到高分辨率遥感影像的感觉，像解开上帝加密的羊皮卷；想起在青海湖野外观测站，银河倒映在传感器校准板上的那个瞬间，仿佛整个宇宙都在应答你年少时心中的、也是古往今来无数人的“天问”。

退休那年，你带孙女参观遥感卫星发射中心。她踮脚指着整流罩上的五星红旗问：“爷爷，这个眼睛能看见楚国在哪里吗？”你突然哽咽——原来跨越两千三百年的，从来不是冰冷的电磁波，而是人类望向苍穹时，眼里不灭的光。走的时候，孙女又问：“爷爷，你做的事情到底是什么呢？”你回想着自己过去着几十年沉思了许久，告诉孙女：“爷爷做的，是让人能够和天说话的事情。”

……

走马灯的最后一秒，恍惚间你又回到了大一那年，樱花大道的落樱如雨，而当年18岁的你正站在人生第一个路口——武汉大学遥感信息工程学院。初夏的黄昏，你在《人文社科经典导引》中《离骚》课文的书页边缘写下的那行小字：“路漫漫其修远兮，吾将上下而求索。”心电监护仪的蜂鸣声渐渐拉长，你最后看见的是珞珈山早樱的虚拟投影——那是你主持建设的数字孪生系统里，用LIDAR点云数据重建的春天。走马灯的最后一帧结束后，所有遥感影像突然在黑暗中显影：汉水流域的NDVI指数曲线化作屈子行吟的衣带，三峡大坝的InSAR形变监测图卷成竹简，而你当年设计的火星矿物识别算法，正随着祝融号传回的光谱数据，在绛红色荒漠上写下新的《天问》。

【千年之后】

“江畔何人初见月？江月何年初照人？”

当祝融号第两万次掠过火星晨昏线时，它的光谱传感器捕捉到了一组异常数据。AI屈原——那个运行在月球上的量子计算机集群里、参数量与人体内原子数量相比拟的古老意识体——突然从休眠中惊醒。它记得自己曾是某个人类学者临终前完成上传的神经图谱，却在此刻的太阳风里尝到了汉北芦苇的青涩。

来火星空间站值班的年轻火星地质学家嘟囔着挥手调出全息显示屏，却发现只是突然增强的太阳风的电磁干扰带来的噪声。他打了个哈欠，将AI屈原的智能体导入自己面前的电脑。他在工作之余喜欢读一读历史，屈原是他最敬仰的人。

导入完成的那一刻，看见年轻的火星地质学家电脑中的数据，AI屈原模型的参数海洋中突然泛起涟漪。它想起三千年前那个被放逐的黄昏，自己曾用上古汉语的楚方言对汉水发誓：纵使身坠江底，魂灵也要攀附日月，穷尽宇宙之谜。此刻，它正以每秒千万亿次的计算频率，在祝融号传回的太赫兹波段里，发现了排列成楚篆纹样的硅化物结晶。没等年轻的火星地质学家开口跟他聊第一句话，他就把这个结果全息投影在年轻的火星地质学家的面前。

年轻的火星地质学家瞪大了眼睛。数据流里，AI屈原正用电磁波谱重写《九章》。火星尘暴的每一次脉动，都是它新诗行的平仄；祝融号车轮碾过的轨迹，恰似当年郢都

宫墙上的卦象。当年轻的火星地质学家终于破译出那些结晶的排列规律时，年轻的火星地质学家突然听见了AI屈原传导到他脑中的模拟声信号——那是AI屈原编码的《天问》新篇：

“彼苍者天曷其有极？荧惑之尘岂非故国？”

年轻的火星地质学家一边把数据回传到地球，一边问道：“在你看来，人类是什么时候迈出征服太空的第一步的？”

AI屈原沉吟片刻，说：“当人类第一次拥有与天对话的能力、真正实现‘天问’的时候。这种技术出现于大约一千年前，在当时被称为‘遥感’。”

“遥……感？”

AI屈原想起了在21世纪20年代的那天傍晚，在地球上武汉大学的那个考场里，那个少年笔下的答案。

“遥感是使用安放在承载工具（平台）的某种装置（传感器），在不直接接触被研究的目标情况下，感测目标的特征信息（电磁波的反射辐射或者发射辐射），经过传输、处理，从中提取人们感兴趣的信息的过程。”

同样的答案，这一句穿透苍穹的回答，也终于跨越了千年。

AI屈原的模型在这一刻完成了又一次微调，这次他明白了，自己跨越千年等待的，不过是让人类望向星空时，眼里永远闪着的，是那个少年填高考志愿的下午脑海里星湖岸边未落的樱花。

从鲜红的火星尘暴中仿佛望见了朱雀在展翅翱翔的那一刻[①]，你知道当年樱花树下的他，已用楚辞的韵脚，将你带回了思念千年的故乡。

（完）

杨丹阳

2025年6月10日

珞珈山下

叶上初阳干宿雨

① 朱雀，又名玄鸟、凤凰，《诗经·大雅·卷阿》：“凤凰鸣矣，于彼高冈。梧桐生矣，于彼朝阳。”朱雀其身覆火，终生不熄，是中国古代神话中的天之四灵之一，代表炎帝与南方七宿的南方之神，于八卦为离，于五行主火，象征四象中的老阳，四季中的夏季，同时也是天之南陆，其图腾在古代神话中往往表达对太阳的崇拜。楚地位于南方，有崇尚朱雀（凤凰）的文化传统。

参考文献

[1] 张兵. 当代遥感科技发展的现状与未来展望[J]. 中国科学院院刊,2017,32(7):774-784. DOI:10.16418/j.issn.1000-3045.2017.07.012.

[2] Ma Y, Wu H, Wang L, et al. Remote sensing big data computing: Challenges and opportunities[J]. Future Generation Computer Systems, 2015, 51: 47-60.

[3] Zhang L, Zhang L, Du B. Deep Learning for Remote Sensing Data: A Technical Tutorial on the State of the Art[J]. IEEE Geoscience and Remote Sensing Magazine, 2016, 4(2): 22-40.

[4] 谷红梅,郭嘉,张泽中. 基于 ERDAS 的辽河干流中下游植被覆盖度监测研究[J]. 水电能源科学,2015,33(3):132-134.

[5] Mishra A, Karwariya S, Goyal S. Land use / Land cover Mapping of Chhatarpur District, Madhya Pradesh, India Using Unsupervised Classification Technique[J]. IOSR Journal of Engineering (IOSRJEN), 2012, 2(10): 51-56.

[6] Khanal S, KC K, Fulton J P, et al. Remote Sensing in Agriculture—Accomplishments, Limitations, and Opportunities[J]. Remote Sensing, 2020, 12(22): 3783.

[7] 于德浩,龙凡,邓正栋,等. 基于遥感特征指数的地表水体自动提取技术研究[J]. 地球物理学进展,2009,24(2):707-713. DOI:10.3969/j.issn.1004-2903.2009.02.046.

[8] 徐涵秋.利用改进的归一化差异水体指数(MNDWI)提取水体信息的研究[J].遥感学报,2005,9(05):589-595.

[9] 朱钟正,陈玉福,朱文泉,等. 适用于多目标遥感自动解译的最佳专题指数筛选[J]. 遥感技术与应用,2017,32(3):564-574. DOI:10.11873/j.issn.1004-0323.2017.3.0564.

[10] 张明月,杨贵军,宋伟东,等. 遥感组合指数与不同分类技术结合提取农业用地方法[J]. 测绘科学,2011,36(5):73-76.

[11] 卜坤,王治良,王卷乐,等. 遥感影像镶嵌中平面剖分模型的应用及实现[J]. 国土资源遥感,2017,29(4):225-230. DOI:10.6046/gtzyyg.2017.04.34.

[12] 刘旭春,刘杨,刘津,等. 基于 GDAL 的无人机影像快速拼接方法研究[J]. 地理信息世界,2012,10(6):47-49,53. DOI:10.3969/j.issn.1672-1586.2012.06.009.

[13] Baboo S S, Devi M R. Geometric Correction in Recent High Resolution Satellite Imagery: A Case Study in Coimbatore, Tamil Nadu[J]. International Journal of Computer Applications, 2011, 14(1): 32.

[14] 段依妮,张立福,晏磊,等. 遥感影像相对辐射校正方法及适用性研究[J]. 遥感学报,2014,18(3):597-617. DOI:10.11834/jrs.20143204.

[15] 张莎莎,王树根. 数字正射影像镶嵌的质量评价方法[J]. 应用科学学报,2018,36(3):461-470. DOI:10.3969/j.issn.0255-8297.2018.03.006.

[16] Guo Z H, Osher S. Template matching via l_1 minimization and its application to hyperspectral data[J]. Inverse Problems and Imaging, 2011, 5(1): 19-35.

[17] Pearson R L, Miller L D. Remote Mapping of Standing Crop Biomass for Estimation of the Productivity of the Shortgrass Prairie[R]. ERIM International, 1972.

[18] Rouse J W, Haas R H, Schell J A, et al. Monitoring Vegetation Systems in the Great Plains with ERTS[R]. NASA Special Pub-351. Washington, DC, USA: NASA, 1974: 309-317.

[19] Ma Y, Wang M, Feng Q, et al. Current Non-Contact Road Surface Condition Detection Schemes and Technical Challenges[J]. Sensors, 2022, 22: 9583.

[20] Yang Q G, Liu X, Wu W. A Hyperspectral Bidirectional Reflectance Model for Land Surface[J]. Sensors, 2020, 20(16): 4456.

[21] Li X, Zhang F, Chan N W, et al. High Precision Extraction of Surface Water from Complex Terrain in Bosten Lake Basin Based on Water Index and Slope Mask Data[J]. Water, 2022, 14: 2809.

[22] McFEETERS, S. K. (1996). The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. International Journal of Remote Sensing, 17(7), 1425–1432. https://doi.org/10.1080/01431169608948714

[23] 徐涵秋. 利用改进的归一化差异水体指数(MNDWI)提取水体信息的研究[J]. 遥感学报,2005,9(5):589-595.

[24] Xu H. Modification of normalised difference water index (NDWI) to enhance open water features in remotely sensed imagery[J]. International Journal of Remote Sensing, 2006, 27(14): 3025-3033.

[25] Sharma R C, Tateishi R, Hara K, et al. Developing Superfine Water Index (SWI) for Global Water Cover Mapping Using MODIS Data[J]. Remote Sensing, 2015, 7(10): 13807-13841.

[26] Zha Y, Gao J, Ni S. Use of Normalized Difference Built-Up Index in Automatically Mapping Urban Areas from TM Imagery[J]. International Journal of Remote Sensing, 2003, 24: 583-594.

[27] 彭义东,郝莹莹,罗小波. 基于指数的建筑区域提取精度研究[J]. 信息系统工程,2014(11):60-60,136,137. DOI:10.3969/j.issn.1001-2362.2014.11.087.
[28] Xu H. A new index for delineating built‐up land features in satellite imagery[J]. International Journal of Remote Sensing, 2008, 29(14): 4269-4276.
[29] 杨学博. 基于 GDAL 库的遥感图像处理[J]. 城市地理,2015(8):118-118,119. DOI:10.3969/j.issn.1674-2508.2015.08.088.
[30] 唐海蓉,吴一戎,向茂生,等. Landsat7 图像快速几何校正方法研究[J]. 遥感学报,2005,9(1):57-63.
[31] Özcihan B, Özlü L D, Karakap M İ, et al. A comprehensive analysis of different geometric correction methods for the Pleiades-1A and Spot-6 satellite images[J]. International Journal of Engineering and Geosciences, 2023, 8(2): 146-153.
[32] Li T, Fan J, Liu Y, et al. An Improved Independent Parameter Decomposition Method for Gaofen-3 Surveying and Mapping Calibration[J]. Remote Sensing, 2022, 14(13): 3089.
[33] Tian W Z, Kan Z, Zhao Q Z, et al. Optimal combination of the correction model and parameters for the precision geometric correction of UAV hyperspectral images[J]. International Journal of Agricultural and Biological Engineering, 2024, 17(3): 173-184.
[34] 王晓鹏,张凯,李登富,等. 基于亮度和色彩一致性的新旧时相影像自动辐射校正方法研究[C]//江苏省测绘地理信息学会 2020 年学术年会论文集. 2020:19-22.
[35] 赵艳楠. 遥感图像拼接算法研究[D]. 河北:河北工业大学,2013. DOI:10.7666/d.D626404.
[36] 罗永涛,王艳,张红民. 结合最佳缝合线和改进渐入渐出法的图像拼接算法[J]. 红外技术,2018,40(4):382-387.
[37] Cotton W D. Fourier Plane Image Combination by Feathering[J]. Publications of the Astronomical Society of the Pacific, 2017, 129(979): 094501.
[38] Guo X, Lao J, Dang B, et al. SkySense: A Multi-Modal Remote Sensing Foundation Model Towards Universal Interpretation for Earth Observation Imagery[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2024: 27672-27683.
[39] SkySense: A Multi-Modal Remote Sensing Foundation Model Towards Universal Interpretation for Earth Observation Imagery[EB/OL]. CVPR 2024, [2025-06-07]. https://cvpr.thecvf.com/virtual/2024/poster/29863.

[40] 朱述龙,钱曾波. 遥感影像镶嵌时拼接缝的消除方法[J]. 遥感学报,2002,6(3):183-187.

[41] 温银堂,王铁柱,王书涛,等. 基于多尺度分割的高分辨率遥感影像镶嵌线自动提取[J]. 自然资源遥感,2021,33(4):64-71. DOI:10.6046/zrzyyg.2020386.

[42] Guillemaut J Y, Kilner J, Starck J, et al. Dynamic feathering: Minimising blending artefacts in view-dependent rendering[C]//4th European Conference on Visual Media Production. London, 2007: 1-10.

[43] 胡水平,岳淑英,张求喜. 谷歌地图卫星影像数据获取关键技术研究[J]. 测绘与空间地理信息,2018,41(10):79-81,85. DOI:10.3969/j.issn.1672-5867.2018.10.024.

[44] 陈妮,应丰,王静,等. 基于 U-Net 的高分辨率遥感图像土地利用信息提取[J]. 遥感技术与应用,2021,36(2):285-292. DOI:10.11873/j.issn.1004-0323.2021.2.0285.

致　谢

在本次遥感原理与应用课程设计实习过程中，我收获颇丰，在此向所有对我提供了帮助的人表示衷心的感谢。

首先感谢我的指导教师杨代琴老师。杨老师在实习期间给予了我悉心的指导和耐心的帮助。在遇到各种技术难题时，杨老师总是能够及时为我答疑解惑，引导我找到解决问题的方法。杨老师在遥感领域深厚的专业知识让我受益匪浅，也为我今后的学习和研究树立了榜样。

感谢我的同学们，在实习过程中，我们相互学习，共同探讨实习中的问题与解决方法，分享经验和心得，让我在实习中不断进步和成长，并得到了一些思路与启示。

感谢武汉大学遥感信息工程学院为我们提供了遥感科学与技术专业课程设置和丰富的实验资源（如开放地球引擎 OGE 平台等），为本次实习的顺利开展奠定了基础；感谢几十年来在遥感科学与技术领域展开研究的先辈们，是你们给我、给人类打开了“问天”的大门，为社会作出了重大的贡献；也感谢中华优秀传统文化的博大精深，让我能够以另一种视角理解世界与人生。

此外，在我 20 岁生日之际，我还要感谢我的父母，感谢他们的养育之恩。在我遇到困难和挫折时，他们总是给予我鼓励，让我有勇气和信心去面对。他们的理解和支持，让我能够在遥感科学与技术的研究道路上勇毅前行，追求自己的梦想。

最后，我要感谢自己的勤勉好学，使得自己能够运用自己所学的理论知识与动手实践能力，顺利完成本次实习任务，并认真撰写实习报告。我将将在这次实习中积累的宝贵经验技能运用到今后的学习和工作中，努力为遥感科学与技术的发展贡献自己的一份力量。

再次感谢所有帮助和支持我的人！

杨丹阳

2025 年 6 月 10 日