Planetary image processing research must be reproducible at the binary, numerical, and geometric-kernel level. The JunoCam haze-layer pipeline presented in this paper is executed entirely within a controlled Linux environment using the USGS Integrated Software for Imagers and Spectrometers (ISIS3). To achieve true reproducibility, all software, all SPICE kernels, and all environment variables are explicitly defined here.

ISIS does not run natively on Windows, and attempting to do so leads to inconsistent geometry, missing libraries, and broken path resolution. Therefore, all Windows systems must run ISIS inside a **WSL2** (Windows Subsystem for Linux 2) Ubuntu environment. Native Linux installations are fully supported.

The steps below describe the exact procedure used in this study to create a stable, deterministic environment for all analysis stages — from raw IMG/LBL ingestion to limb tracing, normal construction, brightness-profile extraction, and haze-layer rectification.

0.1 Operating System Requirements

This pipeline requires:

- Linux distributions. We use Debian 13, Ubuntu 22.04 and 24.04 LTS, running either:
    - natively on Linux, or
    - inside WSL2 on Windows 11
- WSL1 is not supported (no FUSE, no Linux kernel, broken symlink semantics, and missing system calls)
- At the time of publication, Windows-native execution of ISIS3 is impossible (ISIS relies on GLIBC, POSIX filesystem semantics, symbolic links, and Linux-style shared objects)

This paper assumes the user is operating inside a WSL2 Ubuntu terminal or native Linux shell.

0.2 Installing Miniforge and Mamba

ISIS builds distributed by USGS are optimized for the **conda-forge** ecosystem. The reproducible environment begins by installing *Miniforge*, a minimal conda-forge-only distribution.

# Download Miniforge (Linux x86_64):
```
wget https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-
x86_64.sh
```

# Install interactively:
```
bash Miniforge3-Linux-x86_64.sh
```

# Load conda into the current shell:
```
source ~/.bashrc
```

The environment must use **mamba**, the accelerated package solver:

```
mamba --version
```

If mamba is not available, install it:

```
conda install mamba -n base -c conda-forge
```

0.3 Creating the ISIS Environment

A dedicated environment isolates all versions and prevents conflicts downstream:

```
mamba create -n JHC python=3.11
mamba activate JHC
```

After successfully creating your ISIS environment, install ISIS3 from the official USGS channel:

```
mamba install -c usgs-astrogeology isis
```

This places ISIS under:

```
$CONDA_PREFIX/isis
```

and automatically defines the required environment variables (ISISROOT, ISISDATA) on activation.

> ***Note:*** *Some ISIS3 builds do not ship the legacy isisversion binary.*
> *This is expected and not an error. Validation is instead performed via the command spiceinit -help and the presence of the full ISISROOT tree.*

0.4 Ensuring Correct Environment Variables

On each activation use the following commands:

```
conda activate JHC
echo $ISISROOT
echo $ISISDATA
```

Expected:

```
ISISROOT = /home/<user>/miniforge3/envs/JHC
ISISDATA = /home/<user>/.Isis/isis_data
```

To verify the successful installation run:

```
ls "$ISISROOT/scripts" | head
ls "$ISISROOT/lib" | head
```

Presence of downloadIsisData, ISIS plugin libraries, and instrument models confirms a correct install.

0.5 Downloading Offline SPICE Kernels (Required for Reproducibility)

This pipeline intentionally avoids reliance on remote NAIF servers or the USGS ALE SPICE service. Instead, all kernels are downloaded locally using the official ISIS data downloader. To install these kernels, run the following commands in your activated conda environment:

```
downloadIsisData base "$ISISDATA"
downloadIsisData juno "$ISISDATA"
```

Approximate sizes at the time of publication:

- **base:** ~27 GB
- **juno:** ~46 GB

After downloading the kernels, these values can be verified with:

```
du -sh "$ISISDATA/base"
du -sh "$ISISDATA/juno"
```

Directory structure should contain:

```
~/.Isis/isis_data/base/kernels/
~/.Isis/isis_data/juno/kernels/
```

While you can download these kernels using NAIF’s online server, using local kernels guarantees:

- deterministic geometry
- identical limb solutions
- reproducible normals
- stable haze-layer sampling
- replicable results years later regardless of NAIF server changes

These are the core, central reproducibility features of our classification system.

0.6 Verification Tests

Downloading these kernels will take time. After successfully downloading the required kernels, run the following commands to verify that ISIS can initialize geometry and resolve kernels:

```
spiceinit -help
```

Expected output includes the full set of SPICE parameters and server URL:

```
URL = https://astrogeology.usgs.gov/apis/ale/v0.9.1/spiceserver/
```

Next, test the file tree:

```
ls "$ISISDATA"
```

Expected:

```
base
juno
```

Test the ISISROOT path:

```
ls "$ISISROOT"
```

Expected: full ISIS tree (bin, lib, scripts, plugins, etc.)

> ***Note:*** *As mentioned above, some distributions do not include isisversion. This does not affect the correctness or completeness of the installation.*

0.7 Environment Lock for Archival Reproduction

To guarantee future identical reconstruction, you can lock your environment with the following command:

```
conda env export --no-builds > isis_environment.lock.yml
```

This environment lockfile is included in the project repository and ensures that any researcher can reproduce:

- identical ISIS binaries
- identical library versions
- identical SPICE geometry
- identical haze-layer extraction behavior

With the operating system, conda environment, ISIS installation, and SPICE kernel archive now fully configured, the computational environment is complete and reproducible. Every remaining stage of the pipeline—image ingestion, limb tracing, geometric rectification, normal construction, and haze-layer analysis—depends on this foundation. From this point forward, all commands operate inside the activated JHC environment, guaranteeing that every result presented in this paper can be reproduced exactly, on any machine, using the same software versions and the same local kernel set. Having established this controlled environment, we now proceed to Stage 1: ingesting raw JunoCam images and attaching the full SPICE geometry required for scientific analysis.