🚀Stereo DSM Dockerization Workflow — Complete Guide

This document summarizes the **end-to-end process** to dockerize a stereo DSM processing project (using Pleiades/SPOT imagery and Ames Stereo Pipeline), validate it in WSL, and deploy it on Windows PowerShell using Docker Desktop.

Git Repository Structure

```
optical_stereo_dsm_docker/
├── data/                          # Input DEM and satellite images (left/right
image + RPCs)
├── src/                           # Stereo helper functions
│   └── stereo.py
├── run_spot_pipeline.py          # Main script to run stereo DSM workflow
├── config_spot.json              # Configuration file for stereo parameters
├── asp_environment.yml           # Conda environment file
├── Dockerfile                    # Final working Docker build file
├── .dockerignore                 # To skip large folders in Docker build
├── run_docker.sh                 # Shell script to build + run Docker from WSL
├── run_stereo_dsm.ps1            # PowerShell script to run Docker in Windows
├── spot_results/                 # Output folder (created manually if needed)
└── README.md
```

🐳Final Working Dockerfile

```dockerfile
# Use official Miniconda base image
FROM continuumio/miniconda3

# Set working directory
WORKDIR /app

# Install system dependencies
RUN apt-get update && apt-get install -y libgl1

# Copy and create Conda environment
COPY asp_environment.yml .
RUN conda env create -f asp_environment.yml
```

```
# Use the conda environment by default
SHELL ["conda", "run", "-n", "asp", "/bin/bash", "-c"]

# Copy entire repo contents
COPY . .

# Default command (overridden by entry args)
ENTRYPOINT ["conda", "run", "-n", "asp", "python"]
```

.dockerignore

```
# Prevent large files from being included in Docker build
__pycache__
*.tif
*.tar
*.zip
data/
spot_results/
pleiades_results/
```

---

run_docker.sh (For WSL/Linux)

```
#!/bin/bash

# Variables
IMAGE_NAME="optical-stereo-dsm"
SCRIPT_NAME=$1        # e.g. run_spot_pipeline.py
CONFIG_FILE=$2        # e.g. config_spot.json

# Usage help
if [ -z "$SCRIPT_NAME" ] || [ -z "$CONFIG_FILE" ]; then
  echo "Usage: ./run_docker.sh <script_name.py> <config_file.json>"
  echo "Example: ./run_docker.sh run_spot_pipeline.py config_spot.json"
  exit 1
fi

# Build Docker image
echo "🔨 Building Docker image..."
docker build -t $IMAGE_NAME .

# Run Docker container
```

```
echo "🗜️ Running Docker container..."
docker run --rm \
  -v "$(pwd)":/app \
  $IMAGE_NAME \
  $SCRIPT_NAME \
  $CONFIG_FILE
```

🛖run_stereo_dsm.ps1 (For Windows PowerShell)

```
# Run this script from inside your project folder
$projectPath = Get-Location
$imageName = "optical-stereo-dsm"

if (-not (Test-Path "$projectPath\spot_results")) {
    mkdir "$projectPath\spot_results" | Out-Null
    Write-Host "📁 Created 'spot_results' directory."
}

docker run --rm `
  -v "$projectPath\data:/app/data" `
  -v "$projectPath\run_spot_pipeline.py:/app/run_spot_pipeline.py" `
  -v "$projectPath\config_spot.json:/app/config_spot.json" `
  -v "$projectPath\spot_results:/app/spot_results" `
  $imageName run_spot_pipeline.py config_spot.json
```

Step-by-Step Process

1. Prepare Project Repo

- Clone/pull or create your project folder with structure as above
- Ensure `asp_environment.yml` has all required packages (including ASP binaries)
- Do **not** include `spot_results` or `data` in the Docker image itself

2. Build Docker Image in WSL

```
cd optical_stereo_dsm_docker
./run_docker.sh run_spot_pipeline.py config_spot.json
```

3. Test in Interactive Docker (optional)

```
docker run -it --rm \
  --entrypoint /bin/bash \
  -v $(pwd):/app \
  optical-stereo-dsm

# Then inside:
conda activate asp
python run_spot_pipeline.py config_spot.json
```

4. Save Image as TAR (for sharing)

```
docker save -o optical-stereo-dsm.tar optical-stereo-dsm
```

5. Load and Run on Windows

```
# PowerShell
cd Downloads
docker load -i optical-stereo-dsm.tar

# Then use run_stereo_dsm.ps1 to run
./run_stereo_dsm.ps1
```

6. Push to Docker Hub (optional)

```
docker tag optical-stereo-dsm yourusername/optical-stereo-dsm:latest
docker push yourusername/optical-stereo-dsm
```

Then others can:

```
docker pull yourusername/optical-stereo-dsm
```

Summary

- Docker image contains environment + src only
- User mounts `data/`, config, and `.py` from host
- Cross-platform (Linux ↔ WSL ↔ Windows)
- Easy to test, share, and deploy

Let me know when you want to export this as a PDF!