1 Software Architecture

This section describes the modular software architecture developed to validate the photometric stereo pipeline and Poisson solvers.

1.1 Package Structure

The implementation is organized as a Python package with the following directory structure. Each module is self-contained with a single responsibility.

```
python_code/
|
|-- config.py                   # Shared constants and parameters
|-- runner.py                   # Main experiment orchestrator
|
|-- surfaces/                   # Test surface generators (8 files)
|   |-- __init__.py             # Exports all create_* functions
|   |-- gaussian.py             # Gaussian bump surface
|   |-- sphere.py               # Hemispherical surface
|   |-- ellipsoid.py            # Ellipsoidal surface
|   |-- cone.py                 # Conical surface with apex
|   |-- cube.py                 # Flat-top cube surface
|   |-- saddle.py               # Hyperbolic paraboloid
|   |-- peaks.py                # MATLAB peaks function
|   +-- sinusoid.py             # 2D sinusoidal surface
|
|-- photometric/                # Photometric stereo pipeline (4 files)
|   |-- __init__.py             # Exports all PS functions
|   |-- lighting.py             # make_rotating_lights()
|   |-- rendering.py            # render_photometric_images()
|   |-- stereo.py               # photometric_stereo()
|   +-- gradient.py             # gradients_from_normals(),
   compute_divergence()
|
|-- solvers/                    # Poisson equation solvers (4 files)
|   |-- __init__.py             # Exports all solve_* functions
|   |-- fft_periodic.py         # FFT solver (periodic BC)
|   |-- fd_dirichlet.py         # Finite difference (Dirichlet BC)
|   |-- dct_neumann.py          # DCT solver (Neumann BC)
|   +-- tikhonov.py             # Tikhonov regularization
|
|-- visualization/              # Plotting utilities (6 files)
|   |-- __init__.py             # Exports all save_* functions
|   |-- surfaces_3d.py          # 3D mesh plots
|   |-- heatmaps.py             # 2D depth/error maps
|   |-- profiles.py             # Cross-section line plots
|   |-- histograms.py           # Error distribution histograms
|   |-- normals.py              # RGB normal map visualization
|   +-- composites.py           # Multi-panel figure generation
|
|-- experiments/                # Experiment definitions (2 files)
|   |-- __init__.py
|   |-- exp_solver_compare.py   # 8 shapes x 3 solvers comparison
|   +-- exp_ablation.py         # Light sweep, noise, Tikhonov
```

```
|
+-- output/                     # Generated results
    |-- figures/                # 216 PNG images (8x3x9)
    |   |-- gaussian/
    |   |   |-- fft/
    |   |   |-- fd_dirichlet/
    |   |   +-- dct_neumann/
    |   |-- sphere/
    |   |-- ellipsoid/
    |   |-- cone/
    |   |-- cube/
    |   |-- saddle/
    |   |-- peaks/
    |   +-- sinusoid/
    +-- solver_comparison_results.json
```

1.2 Core Modules

1.2.1 Surface Generation (surfaces/)

Each surface module exports a function returning the height field and grid:

```
def create_gaussian_surface(nx=256, ny=256):
    """Returns: X, Y, Z, dx, dy"""
    # 8 surfaces: gaussian, sphere, ellipsoid,
    #             cone, cube, saddle, peaks, sinusoid
```

1.2.2 Photometric Stereo (photometric/)

The photometric module handles the complete PS pipeline:

```
lights = make_rotating_lights(32, elevation=45)
images = render_photometric_images(N_true, lights)
N_est = photometric_stereo(images, lights)
p, q = gradients_from_normals(N_est)
f = compute_divergence(p, q, dx, dy)
```

1.2.3 Poisson Solvers (solvers/)

Three solvers with identical interface:

```
Z = solve_poisson_fft(f, dx, dy)          # Periodic BC
Z = solve_poisson_fd_dirichlet(f, dx, dy) # Zero BC
Z = solve_poisson_dct_neumann(f, dx, dy)  # Zero-flux BC
```

1.3 Running Experiments

```
cd python_code
pip install numpy scipy matplotlib
python runner.py
```

This generates 216 figures (8 shapes $\times$ 3 solvers $\times$ 9 figure types) and a JSON results file with all RMSE values.