UNIVERSITÄT BERN

Lecture 1 - Introduction to Image Analysis

Course Overview & Python for Image Analysis

Amith Kamath

ARTORG Center for Biomedical Engineering Research
University of Bern

Spring/Frühling/Printemps 2026

Lecture Overview & Timing

- **Block 1 (13:15-14:00, 45 minutes):** Course Overview & Semester Roadmap
- **BREAK (14:00-14:15, 15 minutes)**
- **Block 2 (14:15-15:00, 45 minutes):** Code environment setup & Tools

What This Lecture Covers

UNIVERSITÄT BERN

Block 1: Course Foundation & Semester Preview (45 min)

- Website, course structure, grading, and support (10 min)
- Introducing the teaching staff (3 min)
- Biomedical application domains (12 min)
- Mini-walkthrough of the 12-week course content (20 min)

Block 2: Practical Tools & Hands-on Introduction (45 min)

- Python ecosystem: libraries we'll use (5 min)
- Elevation analysis walkthrough (10 min)
- Basic images and histograms (10 min)
- Environment setup with `uv` (5 min)
- Git, GitHub Classroom, Codespaces, and workflows (10 min)
- Assignment 0 instructions & Streamlit demo preview (5 min)

About This Course — Key Resources

Find information about the course here:

`https://ubern-image-analysis.github.io`

On this website, you will find:

- Course schedule and important dates
- All lecture slides and code examples
- Links to GitHub Classroom for assignment submission
- Assignment descriptions and starter templates
- Discussion forum and Q&A resources

Teaching Team & How to Reach Us

UNIVERSITÄT BERN

Instructors:

- **Amith Kamath** — amith.kamath@unibe.ch
- **Dr. Hugo Guillen Ramirez** — hugo.guillenramirez@unibe.ch
- **Prof. Dr. Mauricio Reyes** — mauricio.reyes3@unibe.ch
- **Dr. Pablo Marquez Neila** — pablo.marquez@unibe.ch

Teaching Assistants:

- **Davide Scandella** — davide.scandella@unibe.ch
- **Seyedeh Mirzagar** — seyedeh.mirzagar@unibe.ch

Best Way to Reach Us:

- **Course Questions:** ILIAS Discussion Board (quick community help)
- **Assignment Issues:** GitHub Issues (code-related discussions)
- **Private Matters:** Email directly (24-48 hour response)

This Course in the Master Programs

UNIVERSITÄT BERN

This course is core to two Master programs at the University of Bern:

1. **Master in Biomedical Engineering (BME)**
   - *Particularly relevant for the Image-Guided Therapy (IGT) track*
   - Foundation for medical image processing and surgical guidance
   - Prerequisite understanding for advanced imaging modules
2. **Master in Artificial Intelligence in Medicine (AIM)**
   - Core skill for AI-driven medical image analysis
   - Bridges computer vision and clinical applications
   - Gateway to deep learning for medical imaging

Course Logistics at a Glance

UNIVERSITÄT BERN

Time & Location:

- **Weekly:** Wednesdays, 13:15–15:00
- **Location:** Hörsaal S481, DCBP
- Freiestrasse 3, 3012 Bern
- Duration: 13 lectures + 2 exams

Course Structure:

- 4 major topics (Weeks 1–10)
- 4 assignments (40% grade)
- Midterm + Final exam (60% grade)
- Hands-on programming required

Grading Breakdown

Component	Weight
Assignment 0: Environment	5%
Assignment 1: Filtering	10%
Assignment 2: Interactive Demo	10%
Assignment 3: Deep Learning	15%
Assignments Total	**40%**
Mid-term Exam	30%
Final Exam	30%

Assignments 0, 1, and 3 submitted via GitHub Classroom with automated grading.

Assessment Methods

UNIVERSITÄT BERN

- **Assignments:** GitHub Classroom with automated unit tests (CI/CD via GitHub Actions)
- **Feedback Loop:** Test results and code reviews via GitHub + instructor/TA office hours
- **Exams:** Written exams (likely MCQs)

Image Analysis Applications: Medical & Beyond

Image analysis is everywhere!

Medical/Biomedical Applications:

- Surgical tool tracking and intraoperative guidance
- Medical image segmentation (CT, MRI, ultrasound)
- Automated disease detection and diagnosis
- Microscopy and cell analysis

Other Real-World Applications (same techniques!):

- Autonomous vehicles (road scene understanding)
- Manufacturing (quality control, defect detection)
- Satellite imagery (crop monitoring, disaster response)
- Environmental science (species detection, ecosystem monitoring)

In this course, we emphasize the medical context, but concepts are broader.

Image Analysis: Real-World Applications

UNIVERSITÄT BERN

Image analysis is the extraction of *meaningful information* from images, primarily **digital** images, through techniques of digital image processing to enable *quantitative or qualitative characterization* and understanding of their content.

This course focuses on **applications in medicine and life sciences**. Here are five examples of techniques we will explore:

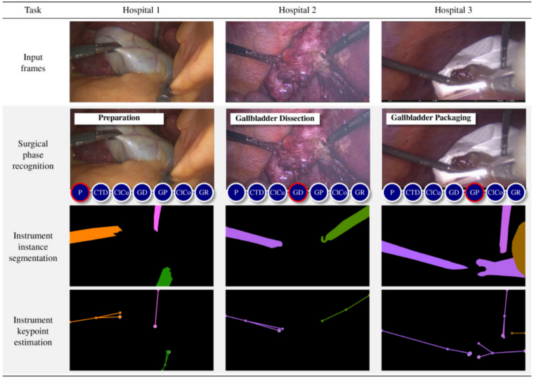
1. Surgical Imaging

Key Applications:

- Surgical tool detection & tracking
- Phase recognition (what step of surgery?)
- Intraoperative navigation
- Hemorrhage detection & blood vessel identification

Challenges:

- Real-time processing required
- Lighting variations
- Occluded instruments



Source: PhaKIR 2024 challenge

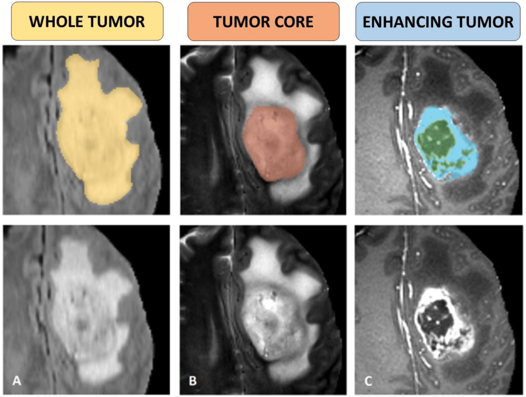
2. Radiology & Medical Imaging

Key Applications:

- Organ & tumor segmentation (liver, lungs, brain, prostate)
- Disease detection & classification
- Registration across different time points or modalities
- Radiomics: quantitative biomarker extraction

Modalities:

- CT (Computed Tomography)
- MRI (Magnetic Resonance Imaging)
- X-ray
- Ultrasound



Source: The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)

3. Ophthalmology

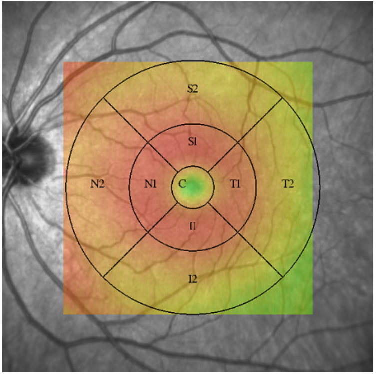
u^b UNIVERSITÄT BERN

Key Applications:

- Fundus image analysis (retinal imaging)
- Diabetic retinopathy detection
- Biomarker extraction (vessel diameter, cup/disc ratio)
- Optical coherence tomography (OCT) segmentation

Clinical Value:

- Non-invasive screening
- Early disease detection
- Population-level studies



Source: ARTORG Center AIMI Lab

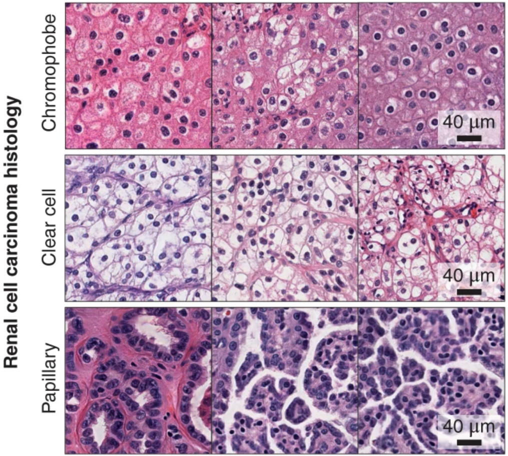
4. Microscopy & Cell Analysis

Key Applications:

- Cell segmentation & tracking
- Calcium signal analysis
- Tissue classification & quantification
- 3D image stacks (z-stacks, confocal)

Biological Questions:

- How many cells? What are their shapes?
- How do cells interact over time?
- What proteins are expressed where?



Source: Mahmood Lab, Harvard

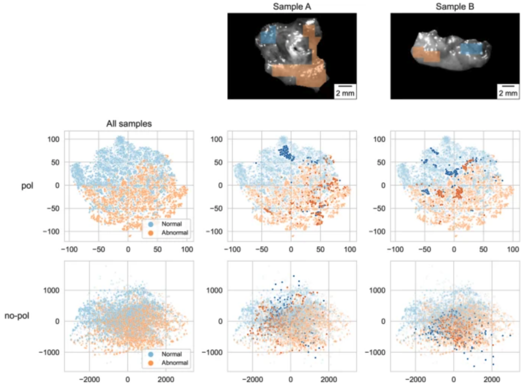
5. Interventional & Intraoperative Imaging

Key Applications:

- Real-time decision support during surgery
- Tumor margin visualization (fluorescence, spectroscopy)
- Resection completeness (R0/R1 prediction)
- Image-guided interventions

Requirements:

- Ultra-low latency processing
- Robust to noise & artifacts
- Interpretable results for clinicians



Source: Sznitmann Lab, ARTORG

Your Journey Through Image Analysis: 12 Weeks

b UNIVERSITÄT BERN

This course is divided into 4 major sections:

1. **Weeks 1-3:** Fundamentals (Image formation, sampling, point operations)
2. **Weeks 4-6:** Classical Methods (Filtering, edges, segmentation)
3. **Weeks 7-9:** Registration & Features (Aligning images, extracting patterns)
4. **Weeks 10-12:** Deep Learning (CNNs, modern architectures, applications)

Let's preview what each week covers so you know what's ahead!

Week 1: Course Overview & Python Setup (Today!)

Topics:

- Course logistics, grading, and resources
- Python ecosystem for image analysis
- Basic image operations (loading, histograms)
- Git workflow and GitHub Classroom

Assignment 0 Released: Environment setup & Python basics

Week 2: Digital Image Formation & Representation

Instructor: Amith Kamath

Key Questions:

- How do cameras capture light and create digital images?
- What is a pixel? How is color represented?
- What are the limitations of digital imaging?

Concepts:

- Camera models and projection geometry
- Sensor types (CCD, CMOS)
- Color models (RGB, HSV, YCbCr)
- Image file formats (PNG, JPEG, TIFF, DICOM)

Assignment 1 Released: Filtering and edge detection

Week 3: Sampling, Quantization, Resolution

Instructor: Dr. Hugo Guillen Ramirez

Key Questions:

- How many pixels do we need? What is Nyquist sampling?
- What happens when we undersample or oversample?
- How do we represent continuous signals digitally?

Concepts:

- Sampling theory and aliasing
- Quantization and bit depth
- Spatial and intensity resolution trade-offs
- Fourier analysis of signals

Week 4: Point Operations & Filtering

UNIVERSITÄT BERN

Instructor: Dr. Hugo Guillen Ramirez

Key Questions:

- How can we enhance image contrast?
- What is convolution and why is it everywhere?
- How do we remove noise while preserving edges?

Concepts:

- Histogram operations (equalization, stretching)
- Linear filtering (box, Gaussian, median)
- Convolution in spatial vs. frequency domain
- Separable filters and computational efficiency

Week 5: Edge Detection & Morphological Operations

Instructor: Dr. Hugo Guillen Ramirez

Key Questions:

- How do we find object boundaries?
- What is an "edge" mathematically?
- How can we clean up binary masks?

Concepts:

- Gradient operators (Sobel, Prewitt, Canny)
- Corner detection (Harris, Shi-Tomasi)
- Morphological operations (erosion, dilation, opening, closing)
- Hough transform for line/circle detection

Week 6: Image Segmentation

Instructor: Prof. Dr. Mauricio Reyes

Key Questions:

- How do we divide an image into meaningful regions?
- What is the difference between thresholding and clustering?
- How can we segment medical images (organs, tumors)?

Concepts:

- Thresholding methods (Otsu, adaptive)
- Region growing and watershed
- Graph-based segmentation (GrabCut, graph cuts)
- Active contours and level sets

Week 7: MID-TERM EXAM

Covers: Lectures 1-6

Topics tested:

- Image formation and sampling theory
- Point operations and filtering
- Edge detection and morphological operations
- Segmentation algorithms

Format: Written exam with conceptual and computational questions

Week 8: Feature Extraction

Instructor: Prof. Dr. Mauricio Reyes

Key Questions:

- How do we describe an image numerically?
- What features are robust to rotation, scaling, and lighting?
- How can we match images or find objects?

Concepts:

- Classical features: SIFT, SURF, HOG
- Texture features (LBP, Gabor filters)
- Shape descriptors (moments, contour analysis)
- Feature matching and RANSAC

Assignment 2 Released: HuggingFace Streamlit demo (group work)

Week 9: Image Registration & 2D vs. 3D Analysis

UNIVERSITÄT BERN

Instructor: Prof. Dr. Mauricio Reyes

Key Questions:

- How do we align images from different times or modalities?
- What transformations preserve anatomy?
- How does 3D imaging differ from 2D?

Concepts:

- Rigid vs. non-rigid registration
- Similarity metrics (SSD, MI, NCC)
- Optimization methods (gradient descent, ...)
- 3D image representation (volumes, slices, projections)

Week 10: Introduction to Deep Learning for Images

UNIVERSITÄT BERN

Instructor: Dr. Pablo Marquez Neila

Key Questions:

- Why are CNNs so effective for images?
- How do we train a neural network?
- What are loss functions?
- How is data split into training/validation/test and why do we need augmentation?

Concepts:

- Convolutional layers, pooling, activation functions
- Backpropagation and gradient descent
- Stochastic Gradient Descent + Adam et al.
- Overfitting and bias variance tradeoffs

Assignment 3 Released: Deep learning for medical image segmentation

Week 11: Introduction to Deep Learning Part 2

Instructor: Dr. Pablo Marquez Neila

Key Questions:

- Classic architectures (AlexNet, VGG, ResNet)
- What are task-specific architectures (ResNet, U-Net)?
- What is the state-of-the-art in medical AI?

Concepts:

- Medical imaging architectures (U-Net, nnU-Net)
- Transformers and Vision Transformers (ViT, self DIstillation, NO labels)
- Foundation models (SAM, MedSAM)

Week 12: TBD: likely best practices

UNIVERSITÄT BERN

Instructor: Amith Kamath

Key Questions:

- How do we ensure our research is reproducible?
- What are good practices to follow when training models?
- How do we validate algorithms for clinical use?

Concepts:

- Version control, documentation
- Evaluation metrics (accuracy, sensitivity, specificity, AUC)
- Clinical validation and regulatory approval (FDA, CE)
- Real-world deployment challenges

Week 13: Assignment 2 Flash Presentations

UNIVERSITÄT
BERN

Instructor: Amith Kamath

What happens:

- Each individual/group presents their HuggingFace Streamlit demo
- up to 5-minute presentation
- Show the live app and explain your approach
- Q&A from instructors and classmates

Goal: Learn from each other and practice communicating technical work!

15-Minute Break

Grab coffee, stretch, chat with classmates!

We'll reconvene at 14:15 for the second (probably more fun) block.

Python for Image Analysis

- Offers an extensive ecosystem of specialized libraries
- Efficient, well-documented tools across basic image manipulation, to advanced algorithms
- Clean(ish) syntax, rapid prototyping capabilities, strong integration with machine learning frameworks
- Therefore, dominant programming language in both academic and industrial computer vision
- See Learn Python With Jupyter for interactive tutorials

Extensive Libraries

Core Scientific Computing:

- **NumPy:** Multi-dimensional arrays, mathematical operations
- **Pandas:** Data tables, statistics, CSV handling
- **SciPy:** Advanced algorithms (interpolation, optimization, signal processing)

Image Processing & Visualization:

- **Pillow (PIL):** Image I/O, basic operations
- **Matplotlib:** Plotting, visualization, displaying images
- **Seaborn:** Statistical visualizations, heatmaps
- **scikit-image:** Classical image processing algorithms

Medical Imaging & Deep Learning:

- **SimpleITK:** Medical image formats (DICOM, NIfTI), registration
- **MONAI:** Medical imaging deep learning framework
- **PyTorch:** Deep learning and neural networks

Example Task: Swiss Canton Elevation Analysis

Let's do our first image analysis together!

```
https://github.com/ubern-image-analysis/lecture-1-elevation-
    analysis
```

What we'll do:

- Load elevation data as images (PNG files)
- Use image masks to extract canton-specific data
- Calculate statistics (min, max, mean elevation)
- Visualize with histograms and plots

Skills you'll practice:

- Loading images with Pillow
- Working with NumPy arrays
- Matplotlib visualization

Elevation Analysis Walkthrough

b UNIVERSITÄT BERN

Use codespaces or, clone/fork the repository:

- Make sure your environment is setup (needed for assignments + rest of the course)
- Fill out code sections. Start with ‘make help‘
- Did you find something surprising?

We’ll work on this until 14:30, then move to basic image operations!

What is an Image (Computationally)?

An image is a 2D (or 3D) array of numbers.

```
import numpy as np
from PIL import Image


img = Image.open(f'images/elevation_map_1km.png')
img_array = np.array(img)
print(img_array.shape)
print(img_array.dtype)
print(img_array[100, 200])
```

Key properties:

- **Shape**: (height, width) for grayscale, (height, width, channels) for color
- **Data type**: `uint8` (0–255) is standard for 8-bit images
- **Indexing**: Images use (row, column) indexing!

Image Histograms: Understanding Intensity Distribution

A histogram shows how often each pixel value appears

```
import matplotlib.pyplot as plt

img_gray = np.array(Image.open(f'images/elevation_map_1km.png')
    .convert('L'))

plt.hist(img_gray.ravel(), bins=256, range=[0,256])
plt.xlabel('Pixel Intensity')
plt.ylabel('Frequency')
plt.title('Grayscale Histogram')
plt.show()
```

Why histograms matter:

- Reveals image contrast and brightness
- Helps detect over/underexposure in medical images
- Used in image matching and retrieval

Git & GitHub Classroom: Version Control Basics

What is Git? A version control system, think of it as "track changes" for code.

Key concepts:

- **Repository** A folder that tracks all file changes over time
- **Commit** A snapshot of your code at a point in time (with a message explaining changes)
- **Push** Upload your local commits to a remote server (GitHub)
- **Clone** Download a repository from GitHub to your computer
- **Fork** Make your own copy of someone else's repository

Quick Git Workflow for This Course

UNIVERSITÄT BERN

For this course:

- We will extensively use **Git and GitHub**
- You **clone/fork** repositories, make changes, and **push** commits
- Assignments are auto-graded; runs on each push (instant feedback!)
- Final submission at deadline is auto-graded

Two ways to work:

1. **Local development:** Clone repo → edit on your machine → push
2. **GitHub Codespaces:** Cloud-based VS Code (no local setup!)

GitHub Classroom: Accepting Your First Assignment

UNIVERSITÄT BERN

Step 1: Accept the assignment via GitHub Classroom

1. Open link that your instructor shares
2. Sign in with your GitHub account
3. Click **"Accept this assignment"**
4. GitHub creates your personal fork of the assignment repository

You now own a private repository with your name.

Example URL: `github.com/ubern-image-analysis/homework-0-yourname`

Option 1: Clone Locally (standard workflow)

UNIVERSITÄT BERN

Clone the repository to your machine:

```
git clone https://github.com/ubern-image-analysis/...
cd homework-0-yourname
make help
```

Work on your code, then commit and push:

```
git add python_basics.py
git commit -m "Implement histogram analysis"
git push origin master
```

Every push triggers automated tests on GitHub Actions!

Option 2: GitHub Codespaces (Cloud Development)

UNIVERSITÄT BERN

No local setup needed—everything runs in your browser!

1. Click the green **“Code”** button on your GitHub repo
2. Select **“Codespaces”** tab → **“Create codespace on main”**
3. Full VS Code environment opens in browser
4. Run: `make setup` and start coding!

Advantages:

- No installation required
- Same environment for everyone
- Works on any device (even tablets!)

Codespaces: Great if your laptop is slow or you want cloud dev. Free limits: 120 core-hours per month, or 60 hours of runtime on a standard 2-core machine.

Git Workflow Tips

Best practices:

- **Commit frequently** with clear messages (helps debugging and shows progress)
- **Push early, push often** (get feedback from automated tests)
- **Don't wait until the deadline** to test your code
- **Use meaningful commit messages:** "Fix histogram bug" not "update"

GitHub Actions Auto-grading:

- Every push triggers automated tests
- Check the **"Actions"** tab to see test results
- Green checkmark = passing, red X = failing
- Click on failed tests to see error messages

Tip: Failing tests? Review the error messages and iterate!

Why We Use `uv` for Python

We're using `uv` (ultraviolet) instead of pip, conda, or venv. Here's why:

- **Speed:** 10-100x faster than pip (written in Rust)
- **Simplicity:** Single tool for dependency management + virtual envs (no venv + pip confusion)
- **Reproducibility:** Automatic `uv.lock` file ensures exact versions across all machines
- **Isolation:** Project-based environments prevent dependency conflicts
- **Modern:** Built for Python 3.10+, designed for today's workflows

Step 1: Install uv and Create Environment

UNIVERSITÄT BERN

Install uv (one-time):

- macOS/Linux: Run the official installer script from `astral.sh/uv`
- Windows: Use PowerShell installer
- Or use Homebrew: `brew install uv`

Create your project environment: Navigate to the course folder and run: `make setup`

(Creates .venv and installs all dependencies from requirements.txt)

Step 2: Activate Environment and Work

UNIVERSITÄT BERN

Activate the virtual environment:

```
source .venv/bin/activate
.venv/Scripts/activate
```

Or run scripts directly without activation:

```
uv run pytest tests/test_python_basics.py
```

Step 3: Verify Your Installation

Test that everything is working:

```
uv run python -c "import numpy, PIL, SimpleITK, matplotlib;
    print('All libraries loaded!')"
```

Or run the Assignment 0 test script:

```
uv run pytest tests/test_python_basics.py
```

Common issues and solutions:

- *uv not found*: Restart your terminal or check installation
- *ModuleNotFoundError*: Run `make setup` again
- *Stuck?* Post in ILIAS or email instructors/TAs!

Assignment 0: Environment Setup & Python Basics

UNIVERSITÄT BERN

What: Verify your Python environment and implement basic functions.

Release	Week 1 (today)
Due	Week 7 (Before mid-term)
Deliverables	Updated `python_basics.py` where all tests pass.
Testing	Automated tests in GitHub Actions
Points	5% of grade

What you'll submit:

- Commit to your fork of the repo with implemented functions for six problems.
- Verify that all 30 tests pass.
- Ask us for hidden bonus tests if you're adventurous!

Access via: GitHub Classroom

Assignment 1: Image Operations & Filtering

UNIVERSITÄT BERN

Release: Week 1 (next week) — **Due:** Week 6 (before midterm)

Topics covered:

- Reading and writing images (PNG, JPEG, DICOM)
- Color space conversions
- Basic image operations (cropping, resizing, intensity manipulation)
- Image filtering (Gaussian blur, edge sharpening)
- Bilinear interpolation for resizing
- Image quality metrics (PSNR, SSIM)

Skills you'll gain:

- Core NumPy array operations
- Working with multi-channel images
- Implementing classical image processing algorithms

Assignment 2: Interactive Web Demo with Streamlit

UNIVERSITÄT BERN

Release: Week 7 — **Due:** Week 13 — **Group size:** TBD — **Points:** 10%

What's Streamlit?

- Python framework for building interactive web apps
- Write Python, get a beautiful web interface
- Deploy to HuggingFace Spaces (free hosting!)
- Share your work with anyone—no installation needed

Your task:

1. Build a Streamlit app for an image analysis task
2. Test locally with real images
3. Deploy to HuggingFace Spaces
4. Present your demo in 5 minutes (Week 13)

Why? Learn how to share your research and build your portfolio!

Example Streamlit Demo: Color Space Explorer

b UNIVERSITÄT BERN

Check out this example:

`https://huggingface.co/spaces/ubern-introimage/colorspaces`

What it does:

- Convert between RGB, HSV, LAB, YCbCr color spaces
- Visualize individual channels
- Interactive sliders to modify colors

Your Assignment 2 could be:

- Object detection demo
- Image segmentation tool
- Filter comparison app
- Something else you fancy!

Assignment 3: Deep Learning Classification

Release: Week 7 (after mid-term) — **Due:** Week 14 (end of course)

Topics covered:

- Training a deep learning classifier
- Working with pytorch and large scale data
- Data loading, hyperparameter tuning, tracking progress
- Evaluations, metrics and interpretation

Skills you'll gain:

- Core PyTorch operations
- Working with large data sets
- Implementing deep learning algorithms

Block 1 & Block 2 Recap

Block 1:

- Course logistics, grading, and motivation
- Semester content preview (12 weeks of image analysis!)

Block 2:

- Git, GitHub Classroom, and version control
- Environment setup with `uv`
- Basic images, and histograms
- Assignment 0 instructions & Streamlit demo preview

By now, you should:

- Have a working Python environment
- Understand the course structure and what's ahead
- Know how to submit assignments via GitHub Classroom
- Be ready to dive into image analysis!

What's Next: Lecture 2

Next week (Week 2): Digital Image Formation & Representation

- **Instructor:** Amith Kamath
- **Topics:** Camera models, projection, color representations, file formats
- **Ideally, make strong progress on Assignment 0**, it is due at mid-term
- **Assignment 1 released:** Filtering and edge detection

Before next lecture:

1. Attempt Assignment 0 (environment setup) via GitHub Classroom
2. Install all required packages (see Assignment 0 instructions)
3. Test your environment and be comfortable with it
4. Join the ILIAS course discussion board (link in course info)

Stuck? Post on ILIAS or email the TAs!

Resources for Self-Study

UNIVERSITÄT BERN

Recommended Tutorials:

- Python documentation: NumPy, jupyter, uv docs
- scikit-image tutorials: Classical image processing
- HuggingFace tutorials: Building and deploying Streamlit apps

Course Materials:

- All slides, code examples, and datasets on GitHub
- Assignment templates and autograder feedback
- Office hours schedule on course website

Thank you for attending!

Questions before we wrap up?

Next steps:

1. Get your Python environment running
2. Make progress on Assignment 0
3. Practice with the elevation analysis notebook
4. Don't hesitate to ask for help!

See you next week!