nimslo project: actionable study guide & implementation plan

path A → path C → path B progression with optional path D

phase 1: foundation - classical stereo vision (path A)

timeline: weeks 8-9 (sept 8-21)

week 8: fundamentals & setup

learning objectives:

- understand stereo vision principles
- grasp epipolar geometry basics
- set up FOSS development environment

actionable tasks:

1. **environment setup** (day 1-2)
    - install OpenCV, numpy, matplotlib, scikit-image
    - test installation with basic image loading
    - create project directory structure
2. **theory deep-dive** (day 3-4)
    - read: "Multiple View Geometry" ch. 9-11 (hartley & zisserman) - available free online
    - watch: cyrill stachniss stereo vision lectures on youtube
    - understand: camera models, calibration, rectification
3. **nimslo analysis** (day 5-6)
    - capture/acquire sample 4-image sets
    - analyze lens spacing and baseline geometry
    - measure approximate focal lengths and distortion
4. **basic implementation** (day 7)
    - load and display all 4 nimslo images
    - implement simple feature detection (harris corners)
    - visualize detected features across all frames

week 9: core stereo algorithms

learning objectives:

- implement feature matching between image pairs
- understand stereo rectification process
- generate basic disparity maps

actionable tasks:

1. **feature matching** (day 1-3)
   - implement SIFT/ORB feature extraction
   - create feature matching pipeline between adjacent pairs
   - filter matches using RANSAC for outlier removal
   - visualize good matches overlaid on image pairs
2. **stereo rectification** (day 4-5)
   - calculate fundamental matrix from matched features
   - implement stereo rectification algorithm
   - verify rectification quality (horizontal epipolar lines)
3. **disparity estimation** (day 6-7)
   - implement block matching algorithm
   - try semi-global block matching (SGBM) from OpenCV
   - generate and visualize disparity maps
   - convert disparity to rough depth estimates

deliverable: basic stereo depth estimation working between at least one image pair

---

phase 2: enhancement - optical flow integration (path C)

timeline: weeks 10-11 (sept 22 - oct 5)

week 10: optical flow fundamentals

learning objectives:

- understand dense vs sparse optical flow
- grasp lucas-kanade and farneback methods
- connect flow to stereo disparity

actionable tasks:

1. **flow theory study** (day 1-2)

- read opencv optical flow tutorials
- understand brightness constancy assumption
- study horn-schunck vs lucas-kanade approaches

2. **dense flow implementation** (day 3-4)
   - implement farneback optical flow between adjacent nimslo frames
   - visualize flow fields as color-coded vectors
   - experiment with flow parameters for best results
3. **flow-stereo connection** (day 5-7)
   - compare optical flow with stereo disparity
   - use flow for stereo correspondence refinement
   - implement flow-guided disparity smoothing

week 11: motion-based depth & animation

learning objectives:

- use optical flow for depth estimation
- create smooth multi-frame animations
- temporal interpolation techniques

actionable tasks:

1. **motion depth cues** (day 1-2)
   - analyze flow magnitude patterns for depth inference
   - combine flow-based depth with stereo depth
   - weight combination based on confidence measures
2. **animation generation** (day 3-5)
   - create traditional 4-frame wigglegram
   - implement flow-based frame interpolation
   - generate smoother 8+ frame animations
   - experiment with different playback speeds/orders
3. **quality assessment** (day 6-7)
   - develop quantitative metrics (alignment error, smoothness)
   - compare flow-enhanced vs basic stereo results
   - identify failure cases and limitations

deliverable: smooth multi-frame animation with improved depth estimation

---

phase 3: innovation - neural network integration (path B)

timeline: weeks 12-13 (oct 6-19)

week 12: CNN depth estimation

learning objectives:

- understand deep stereo networks
- implement transfer learning approach
- compare ML vs classical results

actionable tasks:

1. **literature review** (day 1-2)
    - study key papers: DPSNet, DeepStereo, StereoNet
    - understand network architectures for stereo
    - identify suitable pretrained models
2. **dataset preparation** (day 3-4)
    - augment nimslo image pairs for training
    - create ground truth depth from classical pipeline
    - split data for training/validation
    - implement data loading pipeline
3. **model implementation** (day 5-7)
    - start with simple CNN architecture
    - implement disparity regression network
    - train on nimslo-specific data
    - fine-tune pretrained stereo networks if available

week 13: attention & comparison

learning objectives:

- implement attention mechanisms
- quantitative comparison of all methods
- identify best hybrid approach

actionable tasks:

1. **attention mechanisms** (day 1-3)
   - implement spatial attention for correspondence
   - add attention visualization
   - compare attention maps with classical features
2. **comprehensive evaluation** (day 4-5)
   - test all methods on same image sets
   - measure: accuracy, speed, visual quality
   - create comparison tables and visualizations
3. **hybrid optimization** (day 6-7)
   - combine best aspects of all approaches
   - classical initialization + CNN refinement
   - flow-guided attention mechanisms
   - optimize for nimslo-specific characteristics

deliverable: comprehensive comparison and hybrid method implementation

---

phase 4: documentation & presentation

timeline: weeks 14-16 (oct 20 - dec 10)

week 14: analysis & documentation

actionable tasks:

1. quantitative analysis of all methods
2. identify novel contributions beyond standard approaches
3. document failure cases and future improvements
4. prepare technical documentation

weeks 15-16: presentation & refinement

actionable tasks:

1. create compelling visual demonstrations
2. prepare academic presentation materials
3. finalize code documentation and README

4. optional: implement path D elements if time permits

"dessert" phase: structure from motion (path D)

optional enhancement - implement if ahead of schedule

quick wins for path D:

1. **camera calibration:** use OpenCV calibration on nimslo system
2. **pose estimation:** calculate relative camera positions
3. **sparse 3D:** triangulate matched features into 3D points
4. **visualization:** create point cloud viewer with Open3D

advanced path D:

- dense multi-view stereo reconstruction
- mesh generation and texturing
- interactive 3D model export

---

resource recommendations

FOSS tools priority:

- **OpenCV:** core computer vision algorithms
- **scikit-image:** image processing utilities
- **PyTorch:** neural network implementation
- **Open3D:** 3D visualization (for path D)
- **matplotlib/plotly:** visualization and analysis

key learning resources:

- computer vision: foundations and applications (szeliski) - free online
- opencv-python tutorials - comprehensive and practical
- cyrill stachniss computer vision lectures - excellent theory
- first principles of computer vision youtube channel

evaluation metrics to track:

- alignment accuracy (pixel-level error)

- depth estimation quality (where ground truth available)
- processing time per image set
- visual smoothness of animations
- novelty of approach vs existing methods

---

project log template

```
## [date] - progress update
### completed:
-
### challenges:
-
### next steps:
-
### insights:
-
```