Lab CudaVision
Learning Vision Systems on Graphics Cards (MA-INF 4308)

CudaLab Project

09.07.2025

PROF. SVEN BEHNKE, ANGEL VILLAR-CORRALES

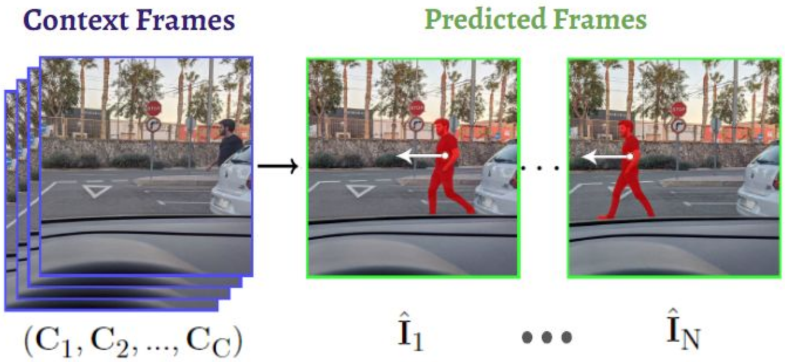
Contact: villar@ais.uni-bonn.de

Evaluating Image Representations for Video Prediction

Future Frame Video Prediction

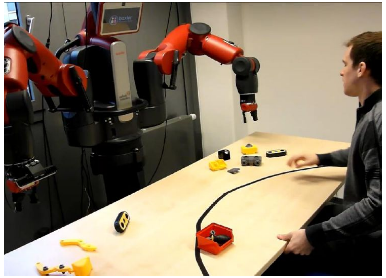
Video Prediction Problem:

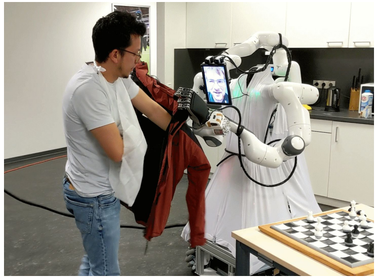
Given **C** consecutive seed/context frames of a video, generate next plausible **N** frames.



Motivation:

- Human-robot collaboration
- Action planning
- Representation learning

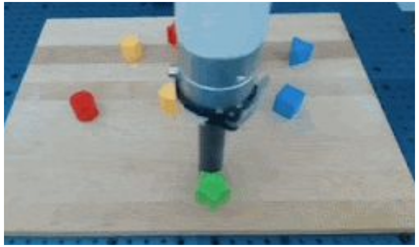


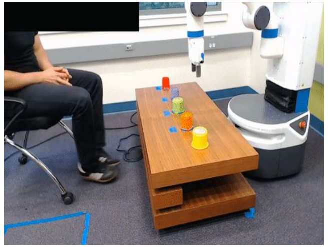


Object-Centric Representation Learning

- Scenes contain moving & interacting objects
- Humans reason over objects — not pixels

➢Can we process images as objects instead of pixels?





Object-Centric Representations:

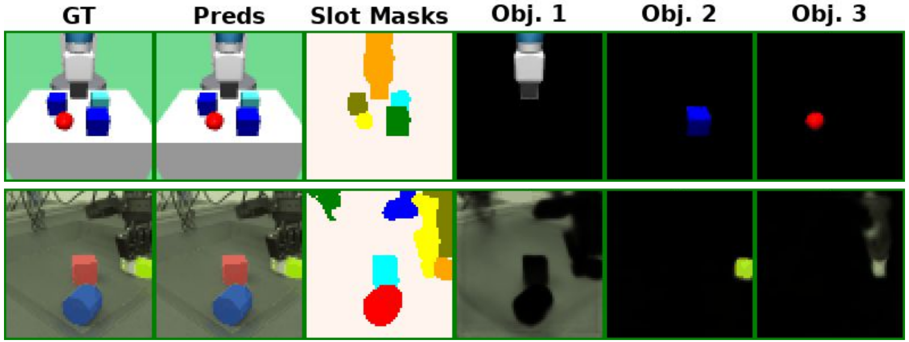
- Encode scene objects
- Robust, interpretable and generalizable
- Well aligned with human perception

➢Can serve as basis for planning and reasoning!

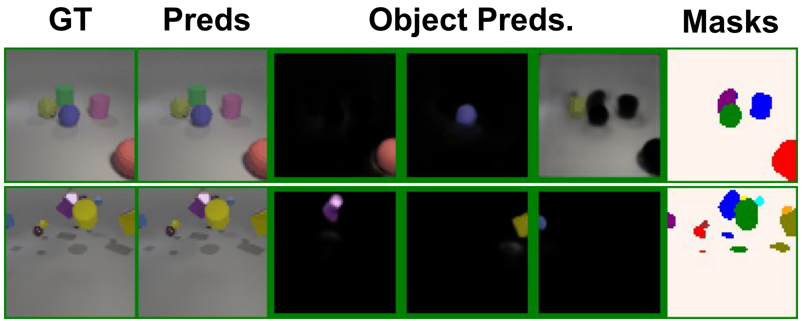
Object-Centric Video Prediction

- Practical approach to video prediction
  1. Discretize frames into objects
  2. Predict future object states
  3. Reconstruct frames from object states

Learning robot behaviors



Learning object dynamics and interactions



Original Caption

'the **large purple rubber cone** is picked up and placed to (2 , 3) . the **small gold metal snitch** is picked up and placed to (-1 , 1) .'

Changed Objects

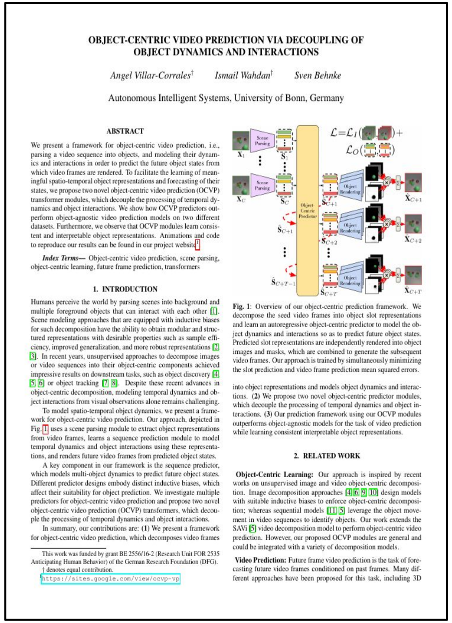
'the **medium cyan rubber sphere** is picked up and placed to (-2 , -2) . the **medium purple metal cone** is sliding to (2 , 3)'

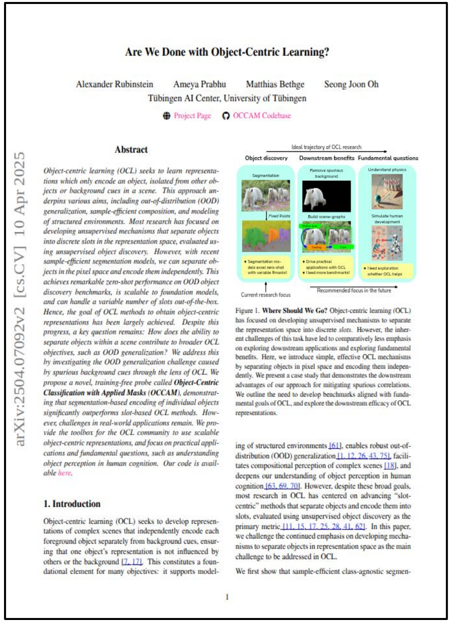
In this Project…

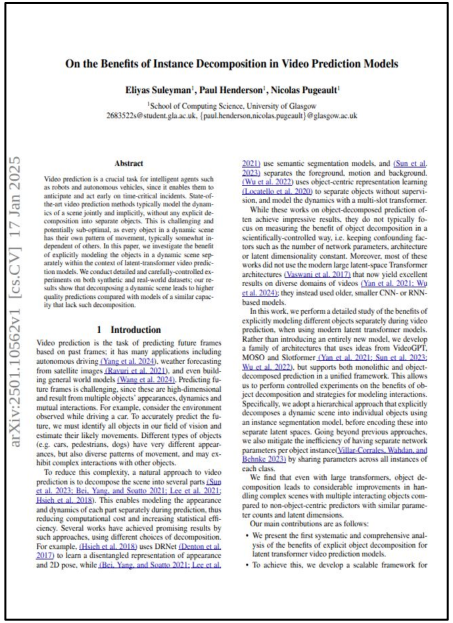
- Train transformer-based video prediction models
- Evaluate different scene representations, including holistic and object-based
- Analyze the effect of different representations

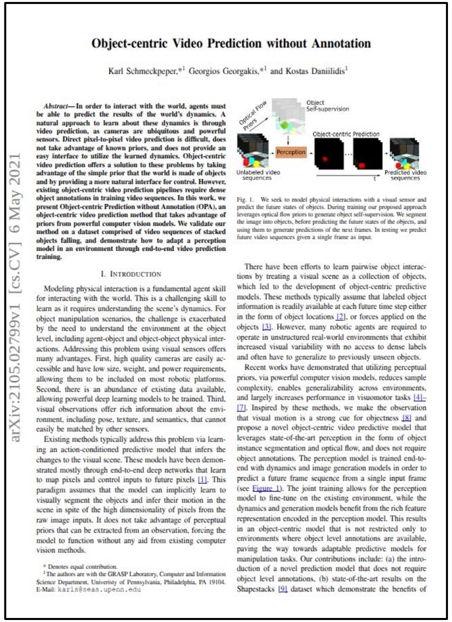
Model

Model Inspiration

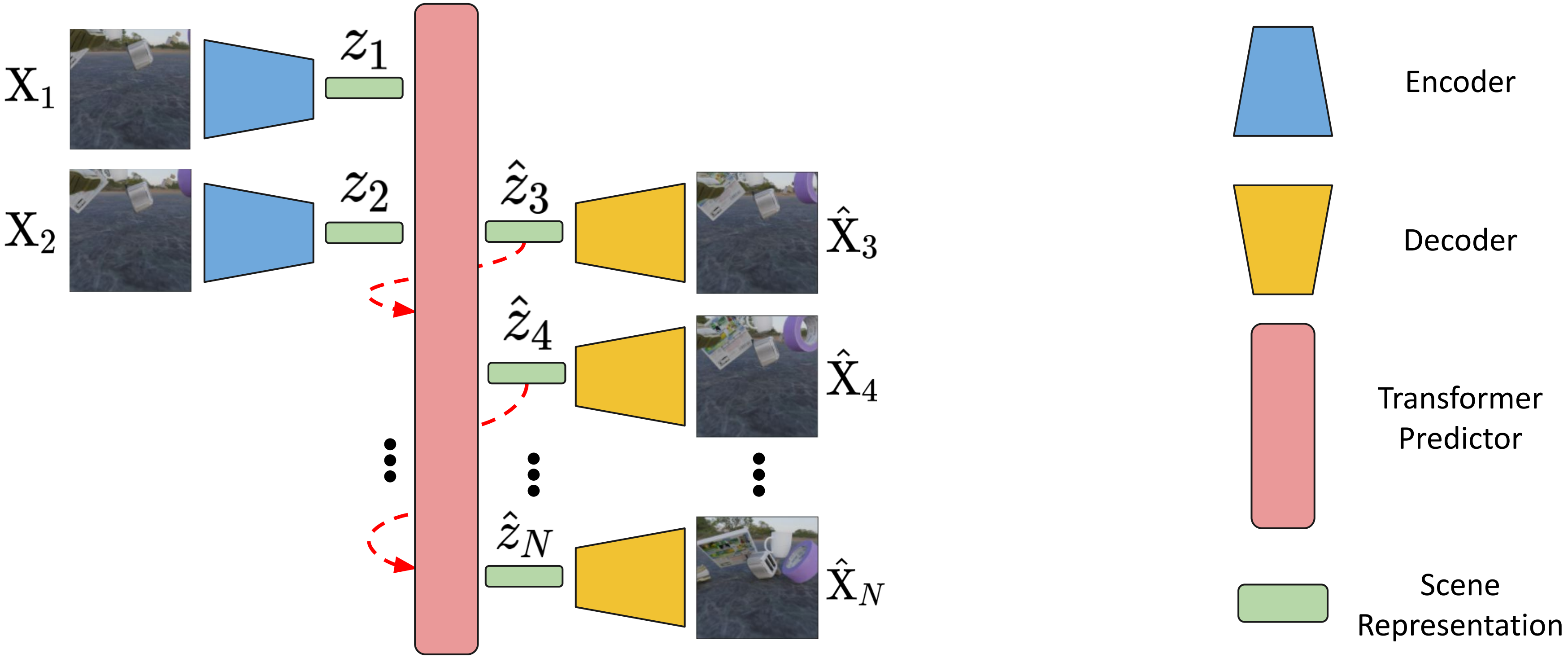








Model



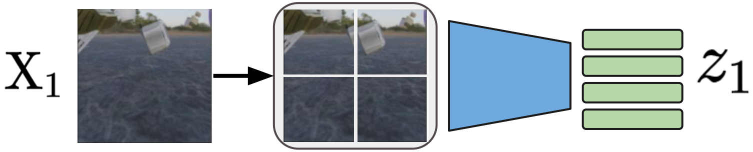
X_1
X_2
z_1
z_2
$\hat{z}_3$
$\hat{z}_4$
$\hat{z}_N$
$\hat{X}_3$
$\hat{X}_4$
$\hat{X}_N$
Encoder
Decoder
Transformer Predictor
Scene Representation

Encoding

- We will evaluate different representations

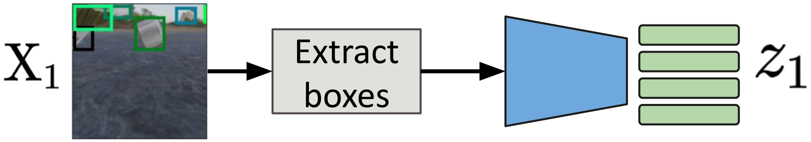
ViT-like Encoder

Encode image patches with self-attention



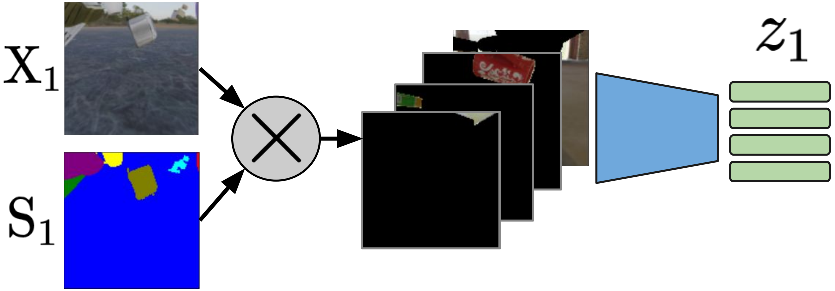
BBox Encoding

Encoder processes each BBox separately



Mask-Encoding

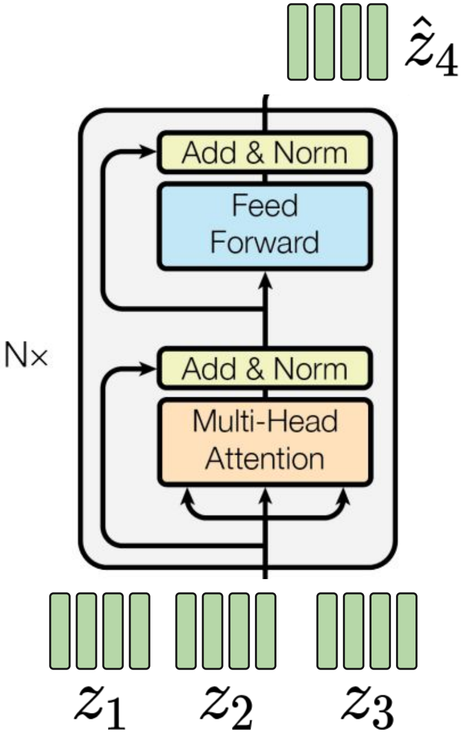
Encode each masked object separately



- Try your own representation/encoding:
  - Concatenate image with segmentation
  - Concatenate image with depth
  - Image + Binary masks
  - …

Predictor

- Autoregressive transformer encoder predictor model
- Process scene representations via self-attention
- Process scene representations via self-attention
- Key modules and variants:
  - Positional encoding
  - Pre-Norm vs. Post-Norm
  - Full Attention vs. Space-time attention (similar to ViViT)



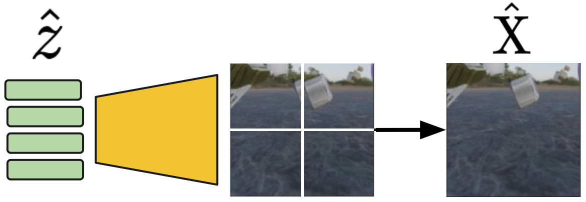
Decoder

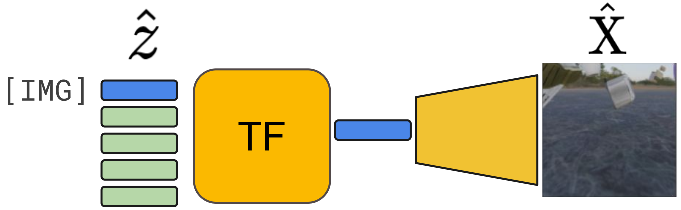
- Map from predicted scene representations to predicted frames

- Different possible architectural designs:

1. Transformer Decoder:
   - Process tokens with self-attention to reconstruct patches
   - Combine patches to render image

2. Conv. Decoder:
   - Process tokens with convolutions and upsampling to directly reconstruct the image
   - Many possible design choices



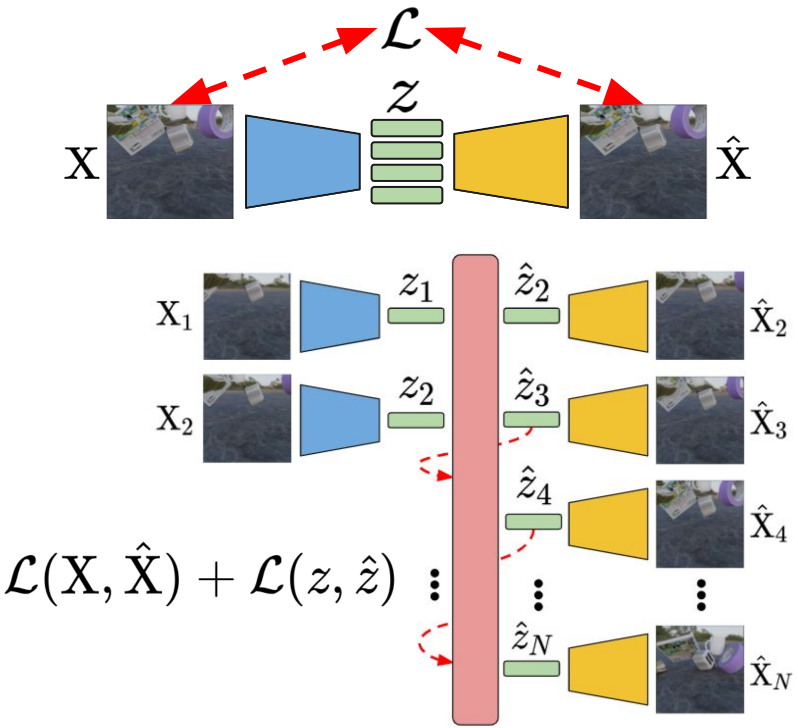


Training

- Two-stage training pipeline

1. Autoencoder Training
   - Train only encoder and decoder
   - Solve image reconstruction task

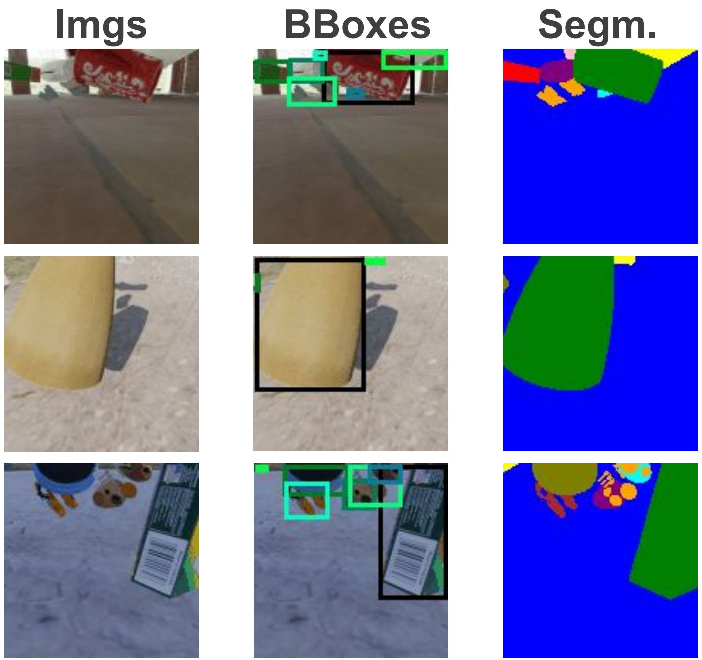
2. Predictor Training
   - Train only the predictor module
   - Encoder and decoder remain frozen
   - Solve image and feature prediction task



Datasets

MOVi-C Dataset

- Synthetic Object-Centric Dataset
  - Rendered realistic objects colliding
  - RGB, Semantics, BBoxes, Flow, …
  - Fixed camera
- Sequences of length 24
  - ~10k training videos
  - ~250 validation videos
- Images are (128, 128)
  - I suggest working with (128, 128) or (64, 64)



Available in: `/home/nfs/inf6/data/datasets/MOVi/movi_c`

Dataset Recommendations

- It usually helps to ‘have more data’
- Apply spatial augmentations (mirror, rotate)
  - Apply the same augmentation to all frames in the sequence!
- Train using randomly sampled subsequences (e.g 1⟶8, 12⟶20, …)
- Use temporal augmentations (e.g. subsampling) if necessary

Training & Evaluation

Training on MOVi

- Training:
  - Train your models using the 2-stage approach described before
  - Train your models using 5 seed frames to predict the next 5
  - I suggest using image resolution of **(64, 64)** or (128, 128)

- You should train **at least two different model variants** using different representations
  - One should be image-level and the other should be object-level
  - I recommend using: ‘Image patches’ and ‘object segmentations’
  - You are encouraged to try out more representations and change the models

Evaluation on MOVi

- Evaluation:
  - Evaluate your models using 5 seed frames to predict the next 15
  - Compute the following metrics: PSNR, SSIM, LPIPS, FVD for 5 and 15 predictions
  - Compute qualitative evaluations: GIFs, images, …

- Investigate and answer the following questions:
  - What representation work best?
  - Why do you think so?
  - What are the strengths and weaknesses of each representation and model?
  - What trade-offs do they present?

Project Goals and Deliverables

Passing Requirements

1. Implement the required model, pipelines and utils

2. Train two different model variants using different representations
   - Evaluate your models quantitatively using PSNR, LPIPS, SSIM & FVD
   - Qualitatively evaluate your models via videos and images
   - You must follow the project guidelines up to certain extent
   - ➢ Make changes to the model and try out things to achieve good results

3. Create overview notebook

4. Write project report

Deliverables

- Complete codebase
  - Clean and structured
  - Not just a notebook!
- Trained model checkpoints and (tensorboard, WandB, …) logs
- Overview notebook (.ipynb & .html) showing main functionalities:
  - Load data and display some samples
  - Load pretrained model and display the structure or some stats
  - Display some qualitative results (e.g. results on 5 sequences)
  - Show the quantitative evaluation
- Project report

Grading

- Results and Experiments **55%-60%:**
  - Performing several experiments and obtaining good results
  - **Additional experiments**: more representations, model changes, ablations, …
  - This grade partly depends on how your results compare to the class
- Codebase & Overview Notebook **20-25%:**
  - Implement all functionalities
  - Modularity and structure
- Report **20%-25%**

Project Report

- Document your work in the project report
- Try to be brief, but readable and informative
- Include figures and tables
- Use *BibTex* for the references
- I expect 8-12 pages, but highly depends on number and size of imgs/tables
- Use the following template
    - https://www.overleaf.com/read/tmnvhrsdmjrp

Important Dates

- **09.07**: Starting date
- **05.09-16.09**: Revision session (very flexible dates, just write me an email)
- **22.09**: Draft submission due (optional)
- **30.09**: Final submission

Questions?