Object recognition and computer vision 2022

Jean Ponce, Ivan Laptev, Cordelia Schmid and Josef Sivic

Final projects

Description:

The final project amounts to 50% of the final grade. You will have the opportunity to choose your own research topic and to work on a method recently published at a top-quality computer vision conference (ECCV, ICCV, CVPR) or journal (IJCV, TPAMI). We also provide a list of interesting topics / papers below. If you would like to work on another topic (not from the list below), which you may have seen during the class or elsewhere, please consult the topic with the class instructor (Ivan Laptev <ivan.laptev@ens.fr>) before submitting the project proposal. You may work alone or in a group of maximum 3 people. If working in a group, we expect a more substantial project, and an equal contribution from each student in the group.

For any chosen project, whenever possible, you should aim to reproduce published results using comparable experimental settings. If you use existing code and pretrained models, you can expect to obtain results similar to the published ones. In other situations, e.g. if you retrain a model using limited training data or train the model for a limited number of iterations, your results might be lower. This is OK, yet it's still important to compare your results using the same test protocol as used by the others. Moreover, it is important to analyze results both quantitatively using metrics adopted for the task by others (accuracy, success rates, detector mAP, …) and qualitatively by visualizing test samples for which the method works well and for which it fails. Visualizing problematic test images/videos and corresponding results is a key to get ideas about further improvements.

Your task will be to:
(i) read and understand a research paper,
(ii) implement (a part of) the paper, and
(iii) perform qualitative/quantitative experimental evaluation.

Evaluation and due dates:

- **Project proposal (due on Dec 6).** You will submit a 1-page project proposal indicating (i) your chosen topic, (ii) the plan of work, i.e. what are you going to implement, what data you are going to use, what experiments you are going to do, (iii) if working in a group, who are the members of the group and how you plan to share the work. The project proposal will represent 10% of the final project grade.
- **Project presentation (Jan 9 and Jan 10)**. You will present your work in the class. The project presentation will represent 20% of the final project grade.
- **Project report (due on Jan 16).** You will write a short report (3 double-column pages in CVPR format -- see below) summarizing your work. The report will represent 70% of the final project grade.

Collaboration policy for final projects

You can discuss the final projects with other students in the class. Discussions are encouraged and are an essential component of the academic environment. We encourage you to work **in a group of 2-3 people (3 people max)**. If working in a group, we expect a more substantial project, and an equal contribution from each student in the group. The final project report needs to explicitly specify the contribution of each student. Both students are expected to present the project at the oral presentation and contribute equally to writing the report. *The final projects will be checked to contain original material. Any uncredited reuse of material (text, code, results) will be considered as plagiarism and will result in zero points for the assignment / final project. If plagiarism is detected, the student will be reported to MVA. See below the policy on re-using other people's code.*

Re-using other people code:

You can re-use other people's code. However, **you must clearly indicate in your report/presentation, what is your own code and what was provided by others (don't forget to indicate the source)**. We expect projects balanced between implementation / experimental evaluation. For example, if you implement a difficult algorithm from scratch, only a few qualitative experimental results may suffice. On the other hand, if you completely use someone else's implementation, we expect a strong quantitative experimental evaluation with analysis of the obtained results and comparison with baseline methods.

Instructions for writing and submitting the project proposal.

- You will submit a **1-page project proposal** indicating (i) your chosen topic, (ii) the plan of work, i.e. what are you going to implement, what data you are going to use, what experiments you are going to do, (iii) who are the members of the group and how you plan to share the work. The due date for the proposal is given at the beginning of this page. The project proposal should be a single 1-page pdf file.
- The proposal pdf should be named using the following format: FPP_lastname1[_lastname2 [_lastname3].pdf, where you replace "lastname*" with last names of all members of your group in an alphabetical order, e.g. for a group consisting of 2 people: I. Laptev, J. Sivic, the file name should be **FPP_Laptev_Sivic.pdf**.
- **Upload your proposal to Google Classroom before the due date.**

Instructions for writing and submitting the final project report

- You will hand-in a **3-page report** in the format of the submission to the IEEE Computer Vision and Pattern Recognition conference (CVPR) . Use the latex or word templates provided at the CVPR Author Guidelines webpage. Note, that you are asked to produce **only a 3-page double-column report** (in contrast, a standard CVPR submission is up-to 8 pages). Figures with qualitative results such as visualization of image classification or object detection results as well as references will not count to the 3-page limit, we encourage you to show visual results.

- At the top of the first page of your report should include (i) names of all members of your group, (ii) the date, and (iii) the title of your final project.

- The report should be a single pdf file and should be named using the following format: FPR_lastname1[_lastname2[_lastname3]].pdf, where you replace "lastname*" with last names of all members of your group in alphabetical order, e.g. for a group consisting of 2 people: I. Laptev and J.

Sivic, the file name should be **FPR_Laptev_Sivic.pdf**.

- **Upload your report to Google Classroom before the due date.**

Instructions on preparing the project presentation.

- Each group will present their final project work in the class.
- **Timing.** The exact timing and schedule of the presentations will be determined during the course.
- **Who should speak?** You can have one person presenting for the whole group, but it is preferable that all members of the group get to present a part of the project.
- **Content.** You should introduce the topic, clearly state what the goal of the project is. Show the work you have done. Compare your results to results of previous methods obtained in comparable settings when possible. When reproducing previous methods, make sure to compare to original results. When describing results, please show both qualitative and quantitative results you have obtained and any interesting observations / findings you have made. Your audience are the other students in the class and the class instructors. You want to show us that you have done interesting work. Remember, it is good to illustrate your findings with images.
- **Reusing material / figures / slides from other people.** You can take figures from papers or other people's slides to illustrate an algorithm or explain a method. However, always properly acknowledge the source if you do so.
- **Submitting slides.** Exact instructions for submitting your presentation will be given later.

Suggested final project topics:

Topic A - Zero-Shot Video Question Answering via Frozen Bidirectional Language Models

Topic B - Object goal navigation in unseen environments

Topic C - Video outpainting

Topic D - Self-Supervised Learning of Visual Representations

Topic E - 3D Hand Reconstruction from Monocular Images

Topic F - Visually-Guided Robotic Manipulation

Topic G - Object detection and tracking with DiffusionDet

Topic H - Test-Time Training with Masked Autoencoders

Topic I - Trajectory Generation with Diffusion Models

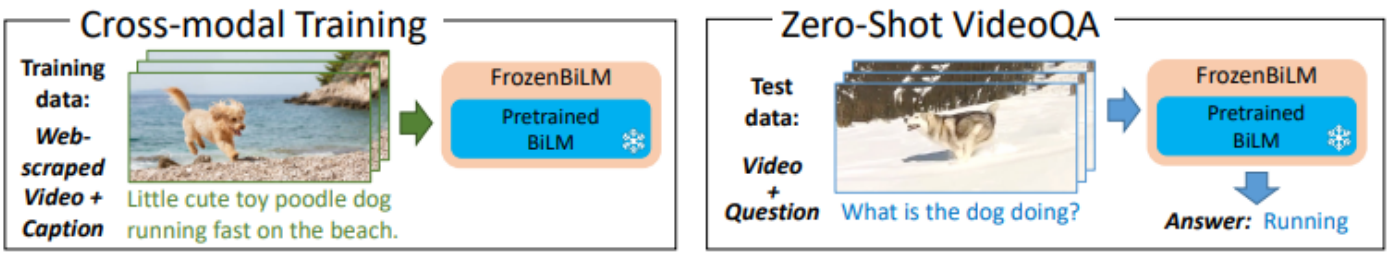
Topic J - Solving Hard Exploration Problems with RL

Topic K – Aligning Text to Sign Language Video

Topic L - Text-conditioned 3D human motion synthesis

Topic X - Your own chosen topic

Topic A - Zero-Shot Video Question Answering via Frozen Bidirectional Language Models



GPU usage: Requires a GPU.

Motivation: Video question answering (VideoQA) is a complex task that requires diverse multi-modal data for training. Manual annotation of questions and answers for videos, however, is tedious and prohibits scalability. To tackle this problem, recent methods [4, 5] consider zero-shot/few-shot settings with no/few manual annotation of visual question-answer. In this setting, state-of-the-art approaches [1, 2, 3] build on language models pretrained on Web text which have inner text-only question-answering ability, augment them with light layers to enable joint vision and language reasoning, and train these light layers on Web-scrapped pairs of image-caption data. During this training phase, the language model is kept frozen which preserves its text-only knowledge and also makes training computationally light.

Description: The goal of this project is to reproduce some zero/few-shot video question answering results from [1] using the open-sourced codebase and extend it to other datasets and/or tasks. In detail, we will rely on a pretrained checkpoint; in particular, you may use the BERT-Base one instead of the DeBERTa-V2-XLarge one to save compute. Pre-extracted visual features and pre-processed data are already provided for supported datasets; for extending the approach, pre-extracted visual features for all tasks and datasets will be provided.

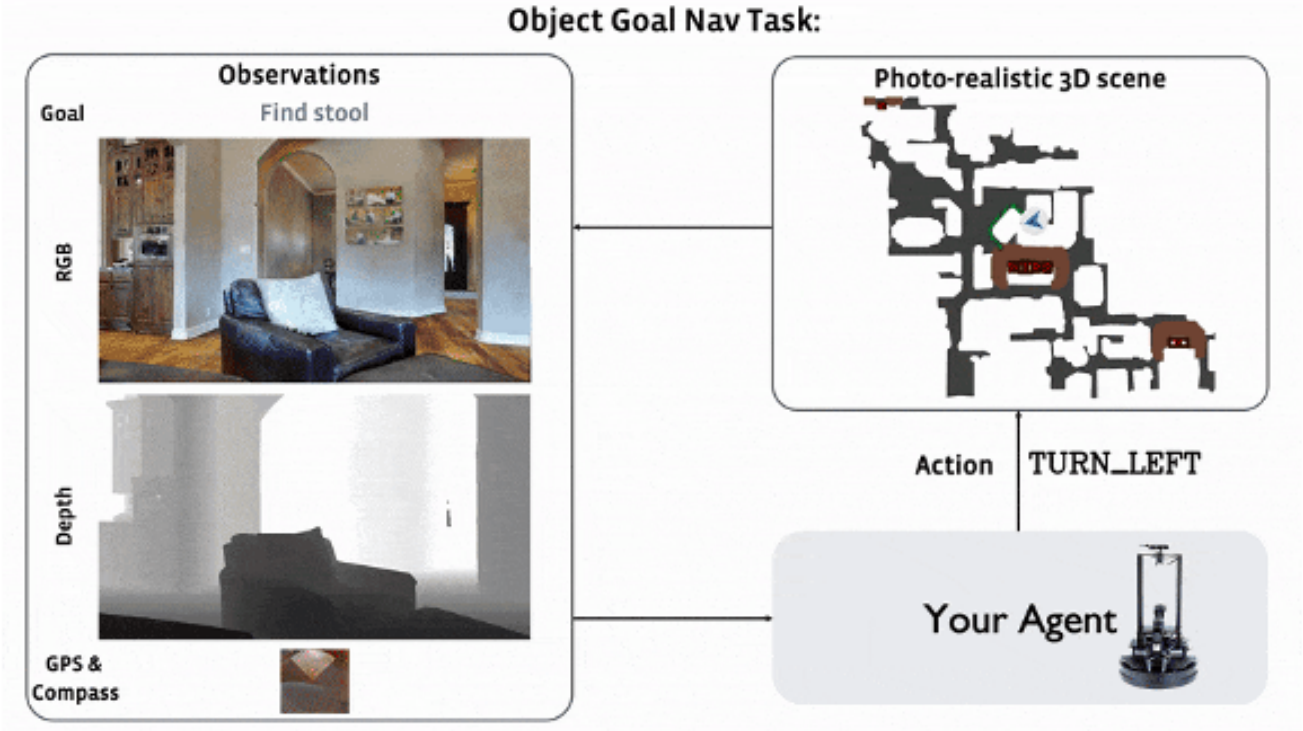
Project Plan:

1) Read [1, 2, 3] to get familiar with frozen language models for vision and language tasks
2) Reproduce the zero-shot and few-shot (1% or 10% of downstream training data) from [1] for the supported dataset of your choice (e.g. iVQA or MSVD-QA).
3) **a) Extension Option 1 (Video Question Answering):** Extend [1] for zero-shot and few-shot video question answering on one/multiple of the following datasets: STAR [6], KnowIT-VQA [7].
   b) Extension Option 2 (Video Captioning): Extend [1] for zero/few-shot video captioning on the MSR-VTT and/or MSVD dataset. You may reuse code from [8]. This will require more coding than Option 1, but will be rated accordingly.
4) **Qualitative analysis (optional):** Qualitatively analyze the failure cases of the zero-shot model vs one of your finetuned model, and the failure cases of the finetuned model, on the dataset of your choice.

References:

[1] Zero-Shot Video Question Answering via Frozen Bidirectional Language Models, Antoine Yang et al, NeurIPS 2022. Code: https://github.com/antoyang/FrozenBiLM
[2] Flamingo: a Visual Language Model for Few-Shot Learning, Jean-Baptiste Alayrac et al, NeurIPS 2022 Oral.
[3] Multi-Modal Few-Shot Learning with Frozen Language Models, Maria Tsimpoukelli et al, NeurIPS 2021.
[4] Just Ask: Learning to Answer Questions from Millions of Narrated Videos, Antoine Yang et al, ICCV 2021 Oral.
[5] MERLOT Reserve: Neural Script Knowledge through Vision and Language and Sound, Rowan Zellers et al, CVPR 2022 Oral.
[6] STAR: A Benchmark for Situated Reasoning in Real-World Videos, Bo Wu et al, NeurIPS 2021 Datasets and Benchmarks Track. Data: http://star.csail.mit.edu/
[7] KnowIT VQA: Answering Knowledge-Based Questions about Videos, Noa Garcia et al, AAAI 2020. Data: https://knowit-vqa.github.io/
[8] SwinBERT: End-to-End Transformers with Sparse Attention for Video Captioning, Kevin Lin et al, CVPR 2022. Code: https://github.com/microsoft/SwinBERT

Topic B - Object goal navigation in unseen environments



GPU: requires roughly 50h GPU.

Dataset: requires students to sign an agreement to obtain a copy of the dataset.

Motivation: Searching for objects in realistic environments is a critical ability of embodied agents, which

serves as a precursor to many object manipulation tasks. To this end, the goal navigation task (ObjectNav) [1] has recently received growing research attention. In this task, an agent is provided with an object category and should navigate to a location next to the target object from which the object is visible.

Description: The goal of this project is to read and investigate state-of-the-art methods [2,3] and train an object-goal navigation model. The project will proceed along the following steps.

1. Read [1,2,3] and make sure you understand the object-goal navigation task, evaluation metrics and the models.
2. Setup simulator environment [2]; download the MP3D dataset and Habitat-web human demonstrations; reproduce results on the MP3D validation split using the released model in [2].
3. Train object-goal navigation models (RNN-based models) using behavior cloning [2] with more advanced RGB features such as CLIP resnet, CLIP ViT [3, 4].
4. Compare the navigation performance of different models in seen and unseen environments; visualize the navigation results; find patterns among failure cases.
5. **a) Extension Option 1:** Explore other model architectures such as transformers [5,6] or map-based models [7].
   b) Extension Option 2: Generalize the model to find novel objects not in the training object categories.

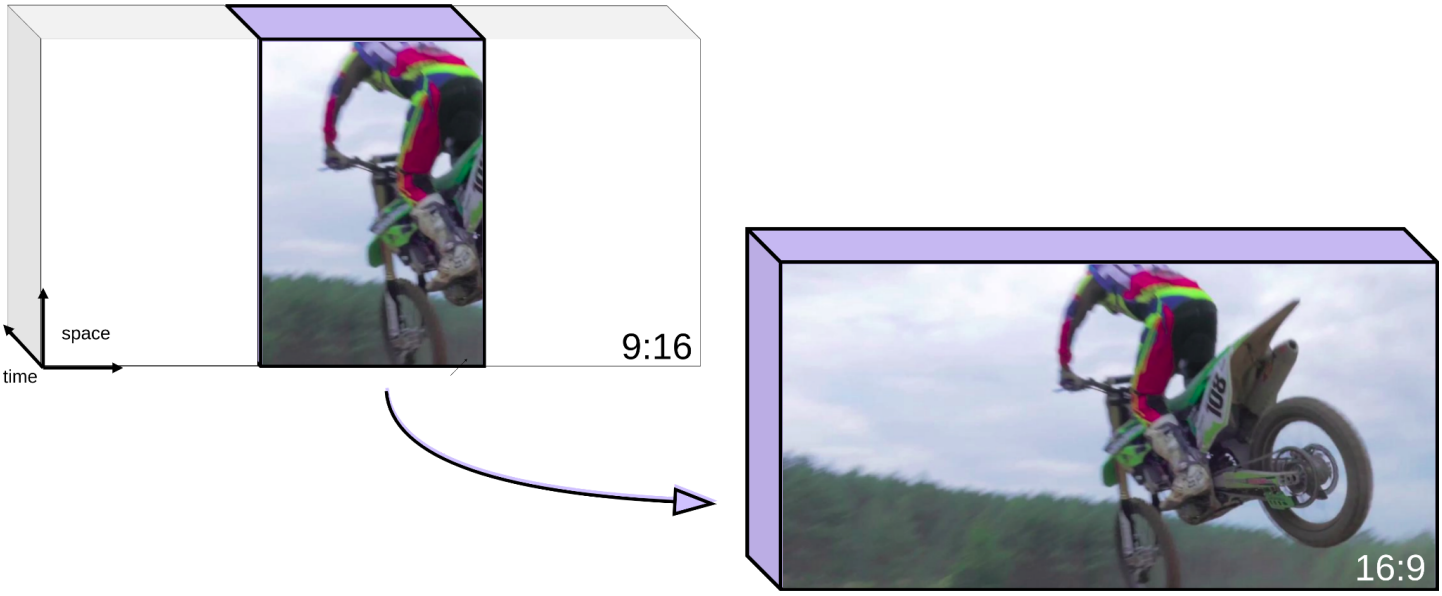
Hints:

1. To reduce resources required for training and evaluation, we suggest using the following reduced subsets of training and validation environments (more environments are encouraged to use if the resource permits):
   Training (8 environments): 17DRP5sb8fy, gZ6f7yhEvPG, dhjEzFoUFzH, PuKPg4mmafe, D7G3Y4RVNrH, Pm6F8kyY3z2, GdvgFV5R1Z5, YmJkqBEsHnH
   Validation (2 environments): pLe4wQe7qrG, oLBMNvg9in8
2. Pre-compute RGB and depth features of human demonstrations for training. You could skip the semantic segmentation features in [2] to reduce training resources.

References:

[1] Batra D, Gokaslan A, Kembhavi A, et al. Objectnav revisited: On evaluation of embodied agents navigating to objects[J]. arXiv preprint arXiv:2006.13171, 2020.
[2] Ramrakhya R, Undersander E, Batra D, et al. Habitat-Web: Learning Embodied Object-Search Strategies from Human Demonstrations at Scale, CVPR 2022. [Code]
[3] Khandelwal A, Weihs L, Mottaghi R, et al. Simple but effective: Clip embeddings for embodied ai, CVPR 2022. [Code]
[4] Radford A, Kim J W, Hallacy C, et al. Learning transferable visual models from natural language supervision, ICML 2021. [Code]
[5] Chen S, Guhur P L, Schmid C, et al. History aware multimodal transformer for vision-and-language navigation, NeurIPS 2021. [Code]
[6] Pashevich A, Schmid C, Sun C. Episodic transformer for vision-and-language navigation, ICCV 2021. [Code]
[7] Ramakrishnan S K, Chaplot D S, Al-Halah Z, et al. PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning, CVPR 2022. [Code]

Topic C - Video outpainting



GPU usage: Requires a GPU.

Motivation: Video outpainting aims at extending the spatial boundaries of a sequence of video frames. Applications include the restoration of old movies originally filmed in 4:3 to the now more standard 16:9 aspect ratio, or the seamless integration of portrait content into landscape videos. Different to image outpainting techniques which only have to fill missing regions with hallucinated content, video outpainting must preserve temporal consistency. This requires understanding of the scene dynamics (camera and object motion) and the propagation of information from neighboring frames to the corresponding extrapolated regions in the current frame.

Description: In this project, we will focus on converting portrait (9:16) to landscape (16:9) videos. Our evaluation protocol will consist in cropping the central region of videos from the DAVIS dataset [1] and then trying to reconstruct at best the original videos.

Project Plan:

1) Our first baseline will be the recent video inpainting method E2FGVI [2]. Read the paper to get familiar with the method. Reproduce the results from the paper on the video inpainting task (in terms of PSNR, SSIM, VFID) using the implementation from the authors and their pretrained checkpoints.
2) Apply (without retraining) the method from [2] to our outpainting task by carefully setting the cropped region accordingly (i.e., mask out the sides of the videos such that the central region has a 9:16 ratio, see above illustration). Compare the results quantitatively to the most simple baseline that comes to mind (e.g., padding the 9:16 frames with border pixel values to match the 16:9 ratio, …) by following the same experimental setup as before (report PSNR, SSIM, VFID). Compare qualitatively the outpainted videos to the ground-truth videos. Describe the failure cases of [2] for the outpainting task.
3) **a) Extension Option 1 (Finetuning E2FGVI):** Finetune [2] on the Youtube-VOS dataset [3] on the outpainting task (instead of the inpainting one). Try to improve results on DAVIS with finetuning on

Youtube-VOS. DAVIS and Youtube-VOS datasets also include ground-truth segmentations indicating salient objects (those of greatest motion). These can be used as an extra supervision signal for helping the model to learn long-range dependencies. For example, you can use them to train the model to outpaint not only the input frames, but also the corresponding segmentations. Other approaches are possible, but please be very clear about whether you use segmentations at train/test time or both, and keep in mind that if you use segmentations at test time, you also have to first crop them to the 9:16 format. Compare (in terms of PSNR, SSIM, VFID and qualitatively) to the outpainting results without finetuning obtained in step 2). Can you draw some conclusions?

b) Extension Option 2 (Building on image inpainting): An alternative is to use an off-the-shelf image inpainting model like [4] and adapt it to the video inpainting task. An example of how to proceed is presented in [5]. Read and make sure you understand this approach. You can start from their code and try one or multiple of the following ideas: use the ground-truth segmentations instead of precomputed ones, use the image inpainting method from [4], complete not only the background flow but also the foreground object flow. You can of course propose your own ideas. Train your proposed model on Youtube-VOS [3] and then test on DAVIS. How does it compare to the baselines obtained in 2) and to [5] in terms of PSNR, SSIM, VFID and qualitatively?

References:

[1] A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation, Perazzi et al, CVPR, 2016 [Paper] [Data]

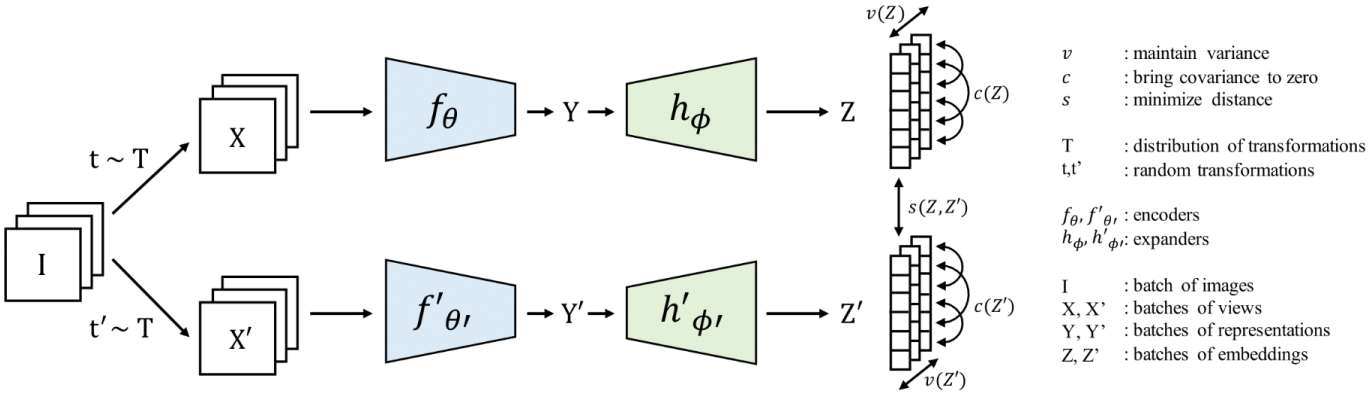
[2] Towards An End-to-End Framework for Flow-Guided Video Inpainting, Li et al, CVPR 2022 [Paper] [Code]

[3] YouTube-VOS: Sequence-to-Sequence Video Object Segmentation, Xu et al, ECCV 2018 [Paper] [Data]

[4] MAT: Mask-Aware Transformer for Large Hole Image Inpainting, Li et al, CVPR 2022 [Paper] [Code]

[5] Complete and temporally consistent video outpainting, Dehan et al, CVPR workshop 2022 [Paper] [Code] [Samples]

Topic D - Self-Supervised Learning of Visual Representations



GPU usage: Requires a GPU.

Motivation: The last decade of progress in artificial intelligence and computer vision has been largely driven by the paradigm of supervised learning, which requires large amounts of labeled data. However the labeling process is costly both in terms of time and money, and is prone to the biases of human data labeling. These reasons motivate the development of self-supervised learning, a new paradigm which enables learning rich visual representations from large collections of unlabeled images. To tackle this problem, contrastive methods [1,2] propose to make embeddings from generated views of a single image closer in the embedding space, while pushing away embeddings from dissimilar images. Non-contrastive methods [3,4,5,6] propose to get rid of the negative term by regularizing embeddings using various strategies.

Description: The goal of this project is to explore VICReg, the recent self-supervised learning method introduced in [3], and (a) pretrain a visual backbone on smaller datasets than what is proposed in the paper, as well as (b) extend the evaluation of an ImageNet pretrained model to new downstream tasks.

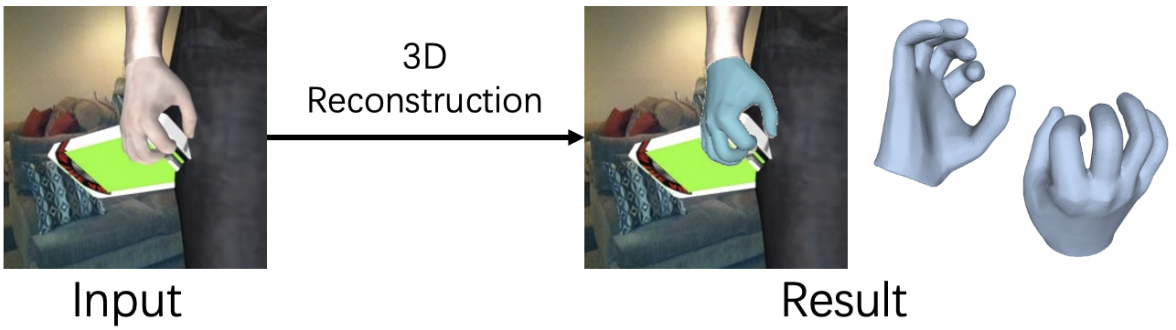
Project Plan:

1) Read [1, 3] to get familiar with current self-supervised learning approaches for learning visual representations
2) Use the VICReg method described in [3] and pretrain backbones on the CIFAR-10, CIFAR-100 or/and ImageNet-100 datasets.
3) Use your pretrained models and perform a *linear evaluation* on their respective test sets.
4) **a) Extension Option 1 (Generalization evaluation).** Use the pretrained ResNet-50 model available to download in the VICReg codebase and evaluate it on CIFAR-10, CIFAR-100 and ImageNet-100, how does it compare to your models directly pretrained on these datasets ?
   b) Extension Option 2 (Fine-grain classification). Pretrain VICReg on the CUB-200-2011 dataset and evaluate the representations on the fine-grain classification task. Play with the data augmentation parameters, how does it influence the performance ?

References:

[1] A Simple Framework for Contrastive Learning of Visual Representations, Chen et al, ICML 2020.
[2] Momentum Contrast for Unsupervised Visual Representation Learning, He et al, CVPR 2020.
[3] VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning, Bardes et al, ICLR 2022. Code: https://github.com/facebookresearch/vicreg
[4] Barlow Twins: Self-Supervised Learning via Redundancy Reduction, Zbontar et al, ICML 2021.
[5] Exploring Simple Siamese Representation Learning, Chen et al, CVPR 2021.
[6] Bootstrap your own latent: A new approach to self-supervised Learning, Grill et al, NeurIPS 2021

Topic E - 3D Hand Reconstruction from Monocular Images



GPU: requires roughly 50h GPU.

Motivation: 3D reconstruction of hands is required for a wide range of applications in AR/VR, robotic manipulation and human-computer interaction. Here, we focus on reconstructing human hands from monocular images, which is a more practical and user-friendly setting compared to reconstruction from depth sensor outputs or multi-view images. Given the ill-posed nature of the problem, the use of prior knowledge is particularly important for successful reconstruction. Existing approaches to single-vew hand reconstruction include (a) parameter estimation for the mesh-based hand model MANO [1] and learning Signed Distance Functions (SDFs) for hand reconstruction [2]. MANO contains strong hand prior knowledge and produces anthropomorphically valid hand meshes, but the generated hand mesh is of limited resolution. Standard SDFs can generate high-resolution meshes but do not incorporate explicit geometric constraints and hand priors. In this project we aim to combine the advantages of both approaches and embed hand pose priors into the learning process of SDF to make more accurate and physically-plausible reconstructions.

Description: The main goal of this project is to explore AlignSDF, the recent method introduced in [5], to incorporate hand pose priors for SDF-based shape reconstruction.

Project Plan:

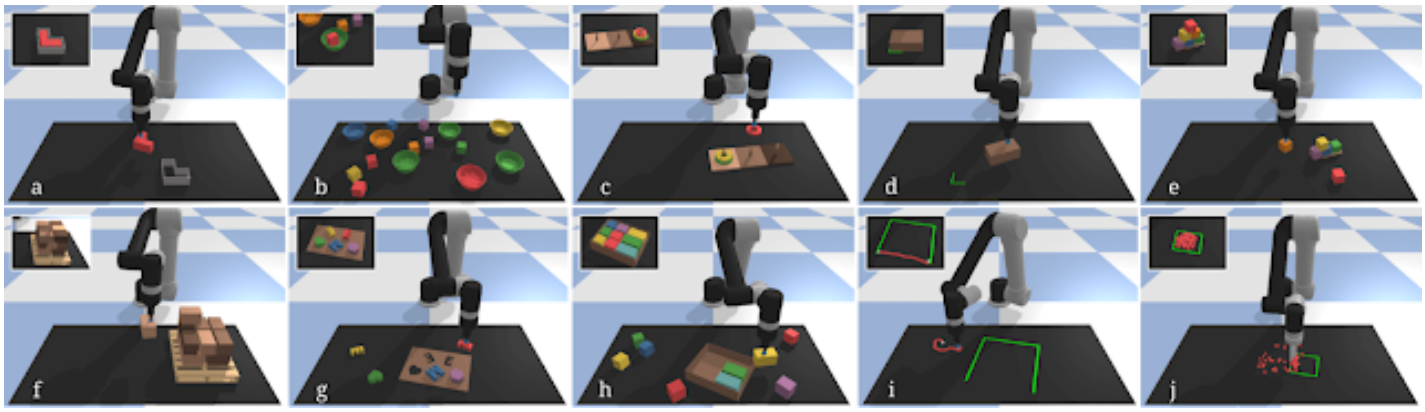
1) Read [1, 2] to get familiar with MANO and Signed Distance Functions (SDFs) for 3D reconstruction.
2) Use the method in [3, 4] to reconstruct hand meshes on the ObMan dataset [3] using MANO and SDFs, respectively.
3) Try AlignSDF [5] to reconstruct hand meshes and compare these three methods [3, 4, 5] quantitatively.
4) **a) Extension Option 1:** Generalize the AlignSDF model to in-the-wild samples (e.g., CORe50).
   b) Extension Option 2: Explore other backbone architectures (e.g., Vision transformer [6]) or other SDF decoder architectures (e.g., [7]).

References:

[1] Embodied Hands: Modeling and Capturing Hands and Bodies Together, Romero et al, TOG, 2017.
[2] DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al, CVPR, 2019.
[3] Learning joint reconstruction of hands and manipulated objects, Hasson et al, CVPR, 2019.
[4] Grasping Field: Learning Implicit Representations for Human Grasps, Karunratanakul et al, 3DV, 2020.
[5] AlignSDF: Pose-Aligned Signed Distance Fields for Hand-Object Reconstruction, Chen et al, ECCV, 2022.
[6] An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, Dosovitskiy et al, ICLR, 2020.

[7] Implicit Neural Representations with Periodic Activation Functions, Sitzmann et al, NeurIPS, 2020.

Topic F - Visually-Guided Robotic Manipulation



References:
[1] Transporter Networks: Rearranging the Visual World for Robotic Manipulation, Zeng et al., CoRL 2020 | Blog article on Transporters
[2] CLIPort: What and Where Pathways for Robotic Manipulation, Shridhar et al., CoRL 2021
[3] AdaBins: Depth Estimation using Adaptive Bins, Bhat et al., CVPR 2021 | Code
Code for [1]: original Tensorflow repository or adapted PyTorch version

GPU: Requires a GPU, Google Colab's GPUs should be enough

Motivation: Pick-and-place actions, the task of grasping an object with a robotic gripper and placing it at a desired position, is central in robotics. It finds applications in plenty of industrial fields, for instance for packing kits, palletizing boxes and managing storage in warehouses. Yet, having robots perform such tasks requires an excellent perception and understanding of the real world as well as a versatile robot, able to adapt to new environments and new tasks. Fullfilling these conditions is a current research topic.

Description: Zeng et al. [1] introduced a simple neural architecture that processes visual observations to accomplish a large diversity of pick-and-place tasks while requiring very few demonstrations to be trained on. This powerful method implicitly solves two problems that are central in robotics: object pose estimation and task planning. The goal of this project is to explore the capabilities of these Transporter Networks and enlarge their skills to multi-task agents or introduce 6D object pose detection in the process.

Project plan:

[A] Get familiar with ideas and details in [1] and make sure you understand the method, how it works and how it implicitly performs pose estimation and task planning

[B] Select 1 or 2 tasks addressed by [1] and reproduce results reported in the paper for these tasks

You can then explore one or multiple options below (students working in a group are expected to work

on two or more options).

[C1 - Option 1: Multi-task network]
Zeng et al. [1] trained one agent per task without sharing experience across tasks. It is, hence, interesting to modify the architecture in [1] to share features and to solve several tasks with the same agent. Modify the method in [1] to learn learn an agent that can solve several tasks with a single model. You may consider the following steps:

- Pick 4 or 5 tasks among those addressed by [1] and create a large dataset with demonstrations for each of these tasks
- Define a random vector for each task, which should encode that task
- Concatenate the output of each decoder network from the Transporters with the vector corresponding to the right task and process this with a small neural network, with the architecture of your choice, in order to get new pick and place heatmaps
- Train the agent on the large dataset and compare performance to the original method.

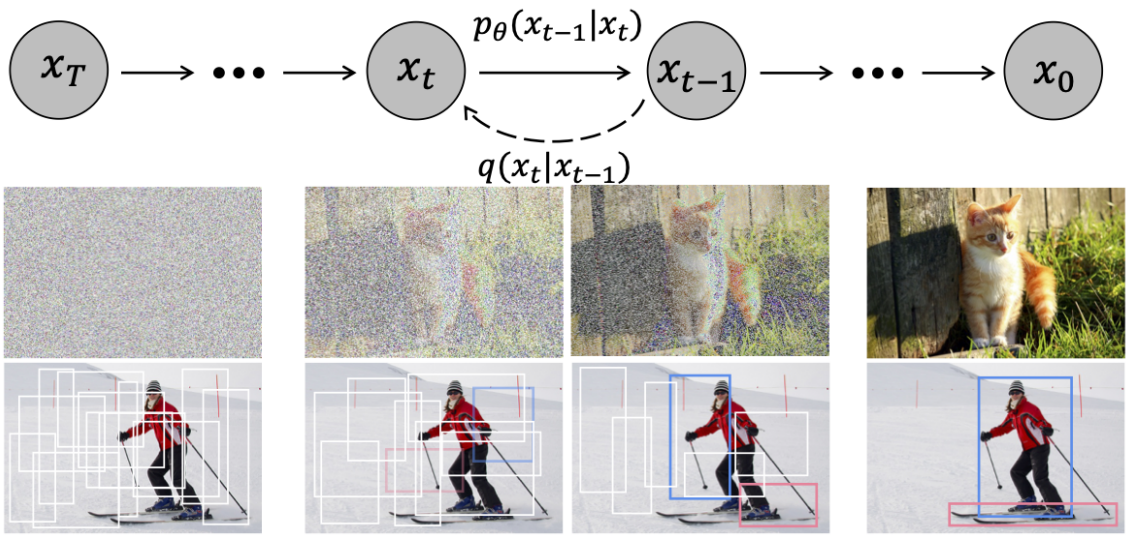
You can also explore other methods and architectures of your choice. You may draw inspiration from [2].

[C2 - Option 2: Feature sharing between pick and place networks]
Transporter's pick and place encoder-decoders are 2 different networks with their own weights. Yet, they take as input the same observation and could share common features. Adapt the Transporter architecture to have a common backbone between pick and place models and keep small specific layers for each step of the pipeline. Study the impact of these changes on the success rate, the number of demonstrations required to train the model, the speed at inference, the computational resources required or any metric that seems relevant to you.

[C3 - Option 3: Remove Ground Truth Depth]
Transporter takes as input RGB-D images. Is the depth channel really useful? To answer that question, explore two tracks:

1. Simply remove the Depth channel, change the input shape of all the networks, retrain them on a few tasks and compare the performance you obtain with those reported in [1].
2. Reuse an existing depth-estimation network to predict the depth channel and give it as input to the original Transporter Network. Compare with the ground truth depth. You may use [3] or any model of your choice.

Topic G - Object detection and tracking with DiffusionDet



GPU: requires GPU
Code: DiffusionDet

Motivation: Diffusion models have recently shown remarkable performance for image generation [5,6,7]. Other recent work extends diffusion models to text-based image editing [2,3] but also other tasks such as motion planning [4] and object detection [1]. This project will explore DiffusionDet [1], the very recent diffusion-based approach to object detection.

Description: Object detection is a set prediction problem where both the number and the classes of objects in the image is unknown a-priori. DiffusionDet [1] poses object detection as a problem of "denoising" noisy object hypotheses. Such hypotheses are simply generated at test time by sampling from a random distribution and are iteratively refined given the image input. We note that if initial object hypotheses are good, DiffusionDet can have a computational advantage compared to previous methods as only a few refinement steps might be needed. Moreover, if object hypotheses come from object detection at previous video frames, refinement of such hypotheses will automatically provide temporal association of object detections, which is a common problem in object tracking [8]. The goal of this project is, hence, to extend DiffusionDet to the problem of object tracking and to compare such an extension to state-of-the-art tracking methods.

Project plan:

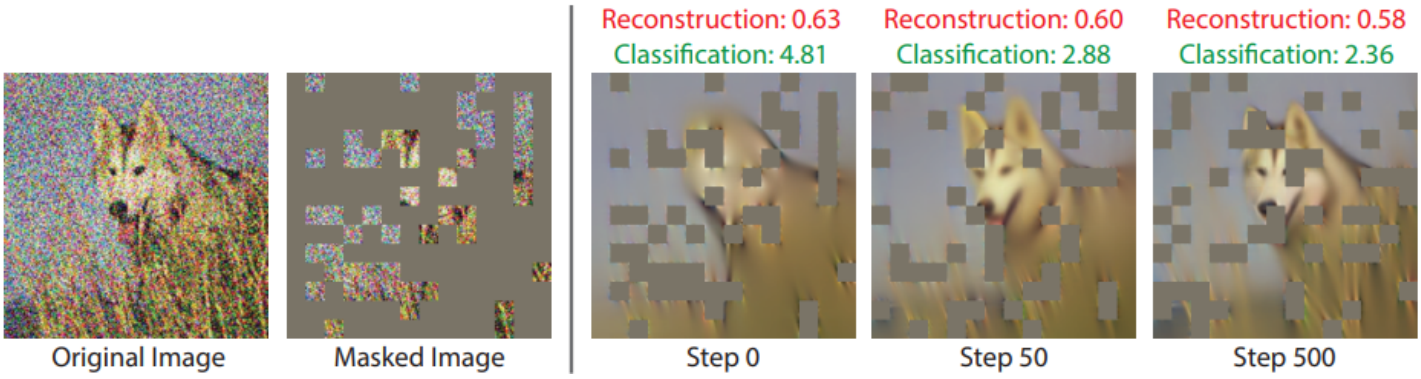
1. Get familiar with DiffusionDet [1] and related diffusion models [5-7].
2. Reproduce some of the detection results in [1] using pre-trained models and code from the authors
3. **a) Extension Option 1:** Design a multi-object tracking approach based on DiffusionDet and compare your approach to SOTA tracking methods on the MOT17 [9] test set. At each new frame your method should (a) associate object detections to object tracks obtained at previous frames and (b) enable initialization of new tracks for new objects.
   b) Extension Option 2: In addition to Option 1, improve tracking results by finetuning DiffusionDet for the object tracking task on video frames. Advanced extensions may consider training DiffusionDet video inputs (in contrast to single-image inputs) such as 2+ video frames, current video frame + optical flow,

etc.

References

[1] DiffusionDet: Diffusion Model for Object Detection (2022) Chen et al., arXiv:2211.09788 [Code].
[2] Imagic: Text-based real image editing with diffusion models (2022) Kawar et al., arXiv:2210.09276
[3] Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding (2022) Saharia et al.
[4] Planning with Diffusion for Flexible Behavior Synthesis (2022), Janner et al., ICML'22.
[5] Denoising diffusion probabilistic models (2020) Ho et al., NeurIPS'20
[6] Diffusion models beat gans on image synthesis (2021) Dhariwal et al., NeurIPS'21
[7] High-Resolution Image Synthesis with Latent Diffusion Models (2022) Rombach et al., CVPR'22
[8] TrackFormer: Multi-Object Tracking with Transformers (2022) Meinhardt et al., CVPR'22
[9] MOT16: A benchmark for multi-object Tracking (2016) Milan et al arXiv:1603.00831

Topic H - Test-Time Training with Masked Autoencoders



GPU: requires GPU
Code: TTT-MAE

Motivation: Masked autoencoders (MAE) have been shown very successful for unsupervised pretraining of large language models [3], and more recently for training computer vision models [2]. The use of unlabeled data in MAE enables its application to very large datasets during training. Instead of using MAE during training, [1] proposes to apply MAE to test images with the aim of adapting existing network to the target image distribution. The goal of this project is to explore and analyze this Test Time Training (TTT) technique.

Project plan:

1. Get familiar with the Test Time Training (TTT-MAE) approach in [1].
2. Reproduce ImageNet-C results [1]. You can sample 1000 images out of the 10000 per category to save computing.
1. Investigate how does the TTT-MAE steps number affects the results.
2. Compare results to the online version where at every new test image the model is not reset to the original weights, but continues from the last checkpoint.
3. Investigate images from the Imagenet validation set in which the algorithm does not work, and

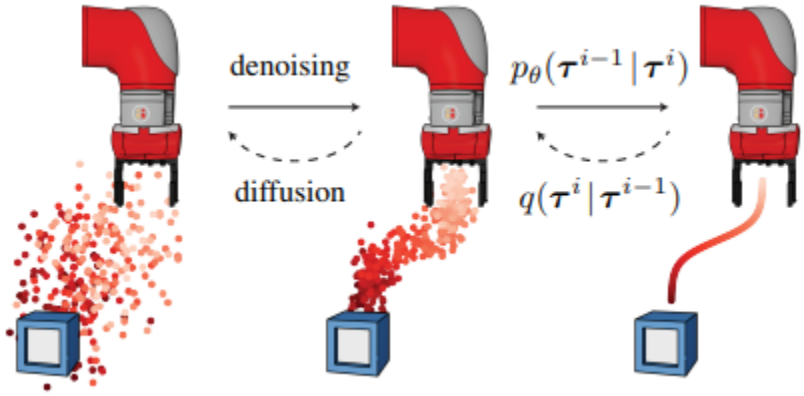
characterizing them.

4. Build an affinity matrix by applying a model train for the test image x_i to all test images x_j=1..N. Cluster images according to such affinity matrix to identify images that belong to the same domain. Analyze and visualize such clusters. Apply TTT-MAE to each cluster using all images within the cluster and see if this improves the performance.

References

[1] Test-Time Training with Masked Autoencoders (2022) Gandelsman et al., NeurIPS'22 [Code, Project page]
[2] Masked autoencoders are scalable vision learners (2022) He et al., CVPR'22.
[3] BERT: Pre-training of deep bidirectional transformers for language understanding (2019), Devlin et al., NAACL'19.

Topic I - Trajectory Generation with Diffusion Models



GPU: requires GPU
Code: Diffuser

Motivation: Diffusion models have shown impressive results for image generation with models such as DALL-E 2 or Imagen. A key property of these generative models is their ability to handle multi-modal data distributions. In this project, we would like to highlight these multi-modal abilities on trajectory generation by building on the work Diffuser [1]. The goal of this project is to compare a Behavioral Cloning baseline to Diffuser [1] on problems with multi-modal solutions, then extend Diffuser from environments with state-based observations to environments with image-based observations.

Project plan:

1. Get familiar with the Diffuser approach in [1] and its codebase. A good reference to build an intuition on diffusion models is the blog post [3], reading it up to the section "Score-based generative modeling with stochastic differential equations" excluded should provide a good introduction to the topic.

2. Reproduce the paper results for Halfcheetah and Walker2d D4RL tasks from Table 2, Medium-Expert dataset in [1].
3. Compare the two following settings:
   A) The default Diffuser model obtained at step 2. that models sequences of state-action pairs [(s_0,a_0),...,(s_T,a_T)].
   B) A model where diffusion is performed only on sequences of actions [a_0, …, a_T].

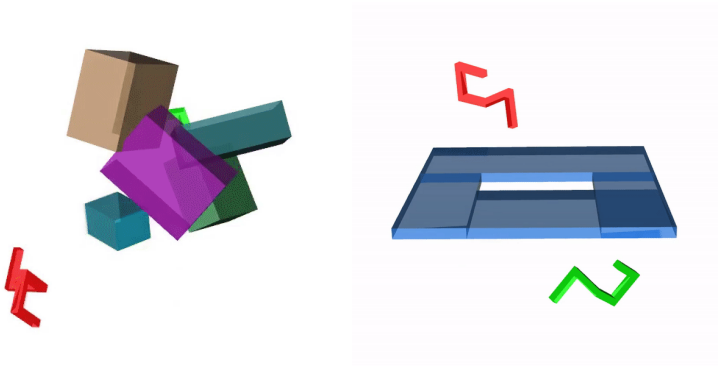
   The model in B) should take the same state inputs as A) but perform diffusion only on the sequence of actions [a_0, …, a_T].
4. Compare 3.A) and 3.B) on the pushing task from [2].
5. Implement a Behavioral Cloning (BC) [5] [6] model on the pushing task from [2].
   Compare BC to 4.B)
6. Extend Step 5. from state-based observations to image-based observations. To activate MuJoCo image rendering, follow instructions in [2]. Then, you can process the images with a ResNet-18 [7] backbone using timm library [8] and use the feature vectors as an input to Diffuser.
   The transformer module performing diffusion should take as input a sequence of tokens composed of the feature vector of the current state output by the vision module x_t and the sequence of future actions [a_t, …, a_T]. The sequence of future actions can be padded with void tokens so that all sequences have the same length.

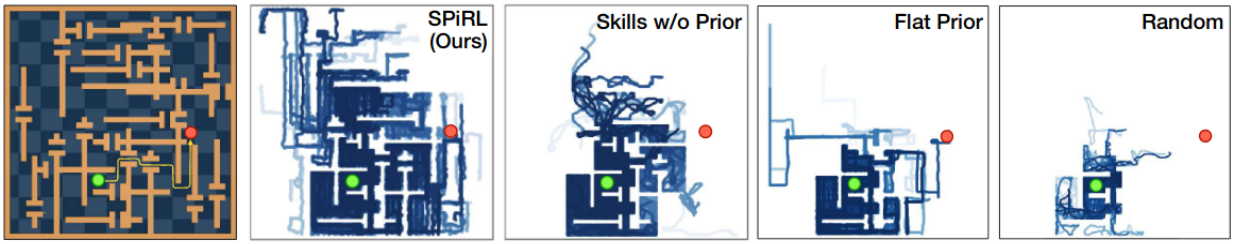
References

[1] Planning with Diffusion for Flexible Behavior Synthesis (2022). Janner et al., ICLR'22 [Code, Project page]
[2] Codebase for the multimodal MuJoCo environments [Code]
[3] Generative Modeling by Estimating Gradients of the Data Distribution. Yang Song.
[4] Denoising diffusion probabilistic models (2020). Ho et al., NeurIPS'20.
[5] An Algorithmic Perspective on Imitation Learning (2018). Takayuki Osa et al.
[6] Imitation Learning Baseline Implementations
[7] Deep Residual Learning for Image Recognition (2016). Kaiming He et al. CVPR'16
[8] PyTorch Image Models. Ross Wrightman

Topic J - Solving Hard Exploration Problems with RL

Requirements: A GPU and familiarity with Deep RL Algorithms.





Motivation: Reinforcement Learning (RL) has been able to solve hard problems such as playing Atari games or solving the game of Go with a unified approach. Yet modern RL approaches struggle to solve hard exploration tasks defined with a sparse reward such as Montezuma's Revenge. We are interested in solving similar problems in Motion Planning, a classical problem in robotics where the goal is to find a collision free path in a cluttered environment. We propose to investigate how to accelerate the learning by either improving the initial policy or using curriculum learning which consists in simplifying the initial problem and increasing the difficulty during learning.

Description: This project proposes to explore enhancements of RL algorithms on hard exploration problems involving perception. We propose to solve a navigation task where a robot has to navigate in a maze-like environment while only getting a sparse reward if it reaches the goal.

Plan:
We will be using Soft Actor Critic [5] with Hindsight Experience Replay [6] (SAC-HER) as a baseline. The problems to solve are maze environments from [1]. To get familiar with the code you should first download [1], make sure you can reproduce results from [1] on Maze-Simple-v0. The goal is then to solve or get improvements over the baseline on Maze-Medium-v0 and Maze-Hard-v0. You can choose to:

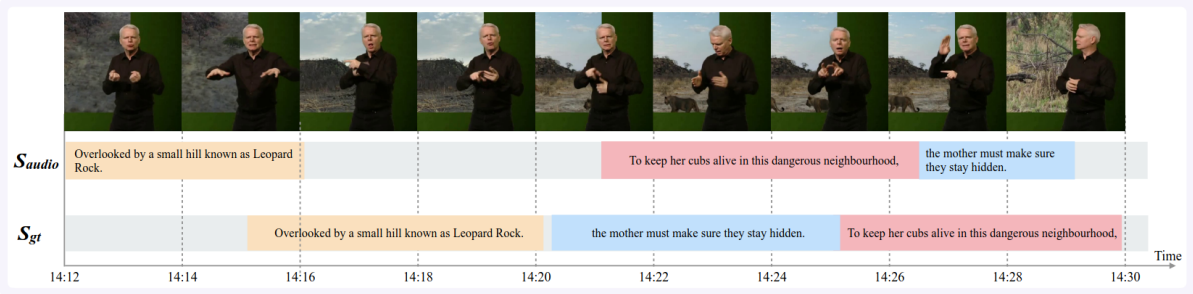
1. Implement curriculum learning [3, 4].
   Initialize with start and goal positions which are at a short distance and increase the distance during learning.
2. Model pre-training on an auxiliary task.
   Pre-train a model on a task such as predicting actions leading to collision,

distance to the goal, distance to obstacles or other meaningful quantities.
Use the pre-trained models to initialize the weights of the RL Q-function and policy.

3. Learn skill priors [2].
   Hard: might require a group of 2.
   Generate a dataset of demonstrations and leverage it to learn skill priors.
   Re-implement SAC modifications of SPiRL [2].
   Use the learned skill prior to solve maze problems faster.
4. Implement enhancements of your choice

References:
[1] Learning Obstacle Representations for Neural Motion Planning. Strudel et al. Code
[2] Accelerating Reinforcement Learning with Learned Skill Priors. Pertsch et al. Code
[3] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. Narvekar et al.
[4] Curriculum Learning. Bengio et al.
[5] Soft Actor Critic. Haarnoja et al.
[6] Hindsight Experience Replay. Andrychowicz et al.

Topic K – Aligning Text to Sign Language Video



Requirements: A GPU and familiarity with PyTorch

Motivation: Sign languages are languages just like French, English or Japanese, but use hands, body orientation and facial expressions as means of communication. British Sign Language (BSL) is the sign language used by the deaf community in the UK. It is a natural language like any other: it was not “invented” but rather evolved over time, it has its own grammar and is different from English, and a native BSL signer can just as easily express concepts in BSL as everyone else in their native language. Although translation systems work quite well for spoken and written languages, translation systems essentially don’t work at all for signed languages, beyond translation of very simple phrases. This project aims to work on a task even simpler than translation, yet still difficult: aligning text sentences to the correct location in sign language video.

Description: The task is to align text sentences (subtitles) to sign language video using the BOBSL dataset. This dataset contains around 13 hours of strongly annotated training data (text sentences which are manually aligned to sign language video), as well as over 1000 hours of weakly annotated training data (text sentences

which are approximately aligned to sign language video). You can look at [1] for an idea upon which to improve.

Dataset: https://www.robots.ox.ac.uk/~vgg/data/bobsl/
You will need to request access.

Example project outline:

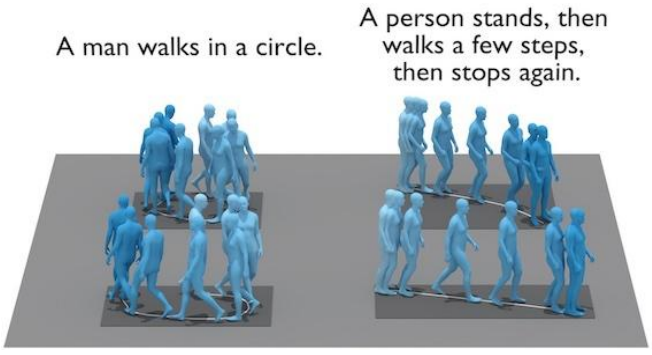
1) Reproduce the baseline results on BOBSL from [2] using the available code here.
2) Try to improve upon these results:
    a) You could try to use additional information from automatic sign spottings, available from the project page of [3]. You could add information on the available sign spottings to the transformer-based architecture, for example by concatenating word embeddings of spottings with the associated video features in the decoder, OR
    b) You could try to take into account more context by inputting 3 consecutive subtitles instead of 1 subtitle, and then try to align all three subtitles at once.
3) Evaluate results to see whether or not there is an improvement
4) Visualise error cases

[1] Bull, H., Afouras, T., Varol, G., Albanie, S., Momeni, L., & Zisserman, A. (2021). Aligning subtitles in sign language videos. In ICCV 2021.
Paper | Project page | Code

[2] Albanie, S., Varol, G., Momeni, L., Bull, H., Afouras, T., Chowdhury, H., Fox, N., Woll, B., Cooper, R., McParland, A. and Zisserman, A., 2021. BBC-Oxford British Sign Language Dataset. *arXiv preprint arXiv:2111.03635*.
Paper | Project page

[3] Momeni, L., Bull, H., Prajwal, K. R., Albanie, S., Varol, G., & Zisserman, A. (2022). Automatic dense annotation of large-vocabulary sign language videos. In ECCV 2022.
Paper | Project page

Topic L - Text-conditioned 3D human motion synthesis



GPU usage: Requires a GPU, but lightweight.

Motivation: Text-conditioned human motion generation has many applications for both the virtual world (e.g., the gaming industry) and the real world (e.g., controlling a robot by speech for personal physical assistance). For example, in the film and gaming industries, motion capture is often used to create special effects involving humans. Since motion capture is expensive, technologies that automatically synthesize new motion data could save time and money. The goal of this project is to train a generative model capable of synthesizing 3D human motion. Since language represents a natural interface for people interacting with computers, it is natural to give a textual description as input.

Description: The goal of the project is to synthesize 3D human motion sequences conditioned by a textual description. You will first read and understand [1], which learns a Variational Auto-Encoder (VAE) to generate human motion from a set of action labels, and then focus on [2] which uses a similar architecture to [1] but produces human motion from a textual description. The project will follow the steps below, but you can also propose another direction to improve [2].

Project plan:

1. Read and understand the papers [1] and [2] on motion synthesis.
2. Use the code from [2] to train and reproduce the results from [2] (both qualitatively and quantitatively).
3. Improve the results (show qualitatively and quantitatively): choose one or more directions below or choose your own:
   a. Add a foot contact loss to avoid "foot slip", as in [3]. Evaluate with a metric that measures foot contacts.
   b. Finetune (or train from scratch) the model (trained with SMPL pose parameters [5]) with a loss on the vertices (after the SMPL layer).
   c. Train the model on the HumanML3D dataset [4] (two ways are possible):
      c.1. Use the features used in [4] in the TEMOS code [2].
      c.2. Load the motion directly from AMASS [6] with the TEMOS code [2].
      c.2.extra (Very technical) Find a way to mirror the SMPL pose parameters [5], and use the data augmentation from [4].
   d. Choose your own direction

[1] ACTOR: Action-Conditioned 3D Human Motion Synthesis with Transformer VAE, Petrovich et al. ICCV 2021.
Code: https://github.com/Mathux/ACTOR
[2] TEMOS: Generating diverse human motions from textual descriptions, Petrovich et al. ECCV 2022.
Code: https://github.com/Mathux/TEMOS
[3] HuMoR: 3D Human Motion Model for Robust Pose Estimation, Rempe et al. ICCV 2021.
Code: https://github.com/davrempe/humor
[4] Generating Diverse and Natural 3D Human Motions From Text, Guo et al. CVPR 2022.
Code: https://github.com/EricGuo5513/HumanML3D
[5] SMPL: A Skinned Multi-Person Linear Model, Loper et al. SIGGRAPH 2015.
Website: https://smpl.is.tue.mpg.de
[6] AMASS: Archive of Motion Capture As Surface Shapes, Mahmood et al. ICCV 2019.
Website: https://amass.is.tue.mpg.de

Topic X - Your own chosen topic

You can also choose your own topic, e.g. based on a paper which has been discussed in the class. Please validate the topic with the course instructor (Ivan Laptev <ivan.laptev@ens.fr>) before submitting the project proposal.