POLITECNICO DI TORINO

Master's Degree in Mathematical Engineering

Master's Degree Thesis

Segmenting dynamic points in 3D scenarios

Supervisors

Prof. Tatiana TOMMASI

Prof. Chiara PLIZZARI

Candidate

Francesco BORGNA

March 2024

Summary

Ma che dici Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Acknowledgements

ACKNOWLEDGMENTS

"HI"
Goofy, Google by Google

Table of Contents

List of Tables

List of Figures

Acronyms

SfM
: Structure from Motion

pcd
: Pointcloud

Part I

Related Works

1. Epic Kitchens
2. Epic Fields
3. Photogrammetry
4. COLMAP
5. NeRF
6. NeuralDiff
7. Monocular Depth Estimation
8. motion estimation
9. (N3F)
10. (Gaussian Splatting)

Chapter 1

Datasets

Motivazioni per cui abbiamo usato questi datasets? The choice of the following datasets was dictated by the fact that up to our knowledge these are currently the largest datasets available in the egocentric scenarios. Let us see these in details.

1.1 EPIC-Kitchens

EPIC-Kitchens [1] is the largest and most varied dataset in egocentric vision up to our knowledge. It contains 55 hours of annotated video data recorded by a head-mounted camera of non scripted actions, meaning that the actors were not following any *scripted* actions(we will see this in more detail later).

1.1.1 Introduction/motivation

EPIC-Kitchens was born to fill the gap in the scarcity of annotated video datasets. As a leading comparison, at the time of writing significant progress have been seen in many domains such as image classification [2],object detection [3], captioning [4] and visual question answering [5]; due to the advances in deep learning but mainly due to the availability of large-scale image benchmarks such as PASCAL VOC [6],ImageNet [7],Microsoft COCO [8], ADE20K [9]. In the same way the authors thought that by introducing a large scale video dataset could contribute to the development of video domains.

Some video datasets were already available for action classification [10, 11, 12, 13, 14] but, a part from [13], these all contain very short videos, focusing on just a single action. A solution to this problem was given by Charades [15] where 10k videos have been collected of humans performing daily tasks at home. The problem with this dataset is that the action recorded were scripted, meaning that the actor had a text in which he was asked to perform some steps. In this way the actions

lose their naturalness, their inbred evolving and multi-tasking properties.

To solve these problems they decided to focus on first-person vision, such that the recording would not interfere with the actor actions, increasing the possibilities of a succesful recording. Also, the viewpoint given by first-person vision allows us to record multi-task actions and the many different ways to perform a variety of important everyday tasks. In Table 1.1 we report a summary of the datasets compared by the authors.

Dataset	Ego?	Non-Scripted?	Native Env?	Year	Frames	Sequences	Action Segments	Action Classes	Object BBs	Object Classes	Participants	No. Env.s
EPIC-KITCHENS	✓	✓	✓	2018	11.5M	432	39,596	149*	454,255	323	32	32
EGTEA Gaze+ [16]	✓	×	×	2018	2.4M	86	10,325	106	0	0	32	1
Charades-ego [41]	70%✓	×	✓	2018	2.3M	2,751	30,516	157	0	38	71	N/A
BEOID [6]	✓	×	×	2014	0.1M	58	742	34	0	0	5	1
GTEA Gaze+ [13]	✓	×	×	2012	0.4M	35	3,371	42	0	0	13	1
ADL [36]	✓	×	✓	2012	1.0M	20	436	32	137,780	42	20	20
CMU [8]	✓	×	×	2009	0.2M	16	516	31	0	0	16	1
YouCook2 [56]	×	✓	✓	2018	@30fps15.8M	2,000	13,829	89	0	0	2 K	N/A
VLOG [14]	×	✓	✓	2017	37.2M	114 K	0	0	0	0	10.7 K	N/A
Charades [42]	×	×	✓	2016	7.4M	9,848	67,000	157	0	0	N/A	267
Breakfast [28]	×	✓	✓	2014	3.0M	433	3078	50	0	0	52	18
50 Salads [44]	×	×	×	2013	0.6M	50	2967	52	0	0	25	1
MPII Cooking 2 [39]	×	×	×	2012	2.9M	273	14,105	88	0	0	30	1

Table 1.1: Comparative overview of relevant datasets(action classes with > 50 samples)

1.1.2 Data Collection

To the data collection were involved 32 people in 4 cities in different countries(in North America and Europe): 15 in Bristol/UK, 8 in Toronto/Canada, 8 in Catania/Italy and 1 in Seattle/USA between May and Nov 2017. Participants were asked to record each time they visit the kitchen for three consecutive days, starting filming just before entering the kitchen and stopping before leaving it. They participated to the process of their own free will without being paid in any way.

Few requests were asked to them. The first was to be in the kitchen alone during the recording, such that no inter-person interaction could interfere. The second one instead was to remove all items that could disclose their identity, for example portraits or mirrors. In this way they could remain anonymous.

Each participant was equipped with a head-mounted camera with adjustable mounting such that it could be adapted to the participant's height and possibly different environment. They had to check, before each recording, the battery life and the viewpoint, such that their stretched hand were approximately located at the middle of the camera frame. The camera settings was set for most of videos to linear field of view, using 59.94fps as frame rate and Full HD resolution of 1920x1080, however some subjects made minor changes like wide or ultra-wide FOV or resolution. In particular 1% of the videos were recorded at 1280x720 and 0.5% at 1920x1440. Also 1% at 30fps, 1% at 48fps and 0.2% at 90fps.

On average, each participant recorded 13.6 sequences, each of those lasted on average 1.7 h while the maximum duration recorded was of 4.6h. The duuration of the recording was obviously linked to the person's kitchen engagement. In Figure 1.1 we can see some statistics of the data acquired.

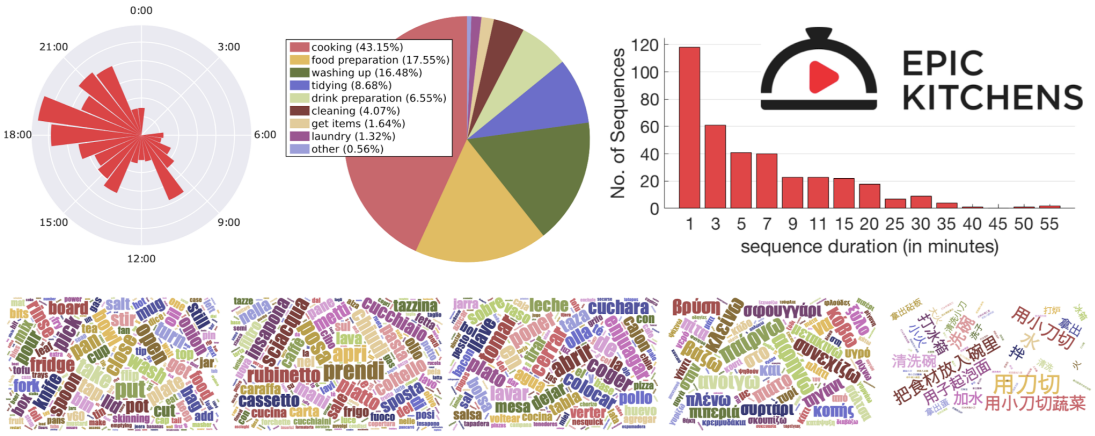

Figure 1.1: Top (left to right): time of day of the recording, pie chart of high-level goals, histogram of sequence durations and dataset logo; **Bottom**:Wordles of narrations in native languages(English, Italian, Spanish, Greek and Chinese).

1.1.3 Data Annotation pipeline

After the end of a sequence each participant was asked to watch the recording and narrate verbally the actions carried out to a microphone. The sound narration was chosen beacuse it was faster than a written one, and participants were thus more willing to provide these annotations. The guide lines for narrations are reported in Figure 1.2.

Use any word you prefer. Feel free to vary your words or stick to a few.
Use present tense verbs (e.g. cut/open/close).
Use verb-object pairs (e.g. wash carrot).
You may (if you prefer) skip articles and pronouns (e.g. "cut kiwi" rather than "I cut the kiwi").
Use propositions when needed (e.g. "pour water into kettle").
Use 'and' when actions are co-occurring (e.g. "hold mug and pour water").
If an action is taking long, you can narrate again (e.g. "still stirring soup").

Figure 1.2: Narration Guidelines given to each participant to be followed after the completion of a recording.

The most used language was English, but other languages was used, if the participant was not so fluent in English. In particular a total of 5 languages were used: 17 people narrated in English, 7 in Italian, 6 in Spanish, 1 in Greek and 1 in Chinese.

The motivation to obtain the narrations directly from the actors was due to the fact that they surely knew what they were doing, avoiding misinterpreting some possible actions. The posthumous narration was instead motivated by the fact that actors could perform their actions in the most natural way, without being concerned about labelling.

The second step of annotations consists in the transcription of the speech narrations. After testing some automatic audio-to-text algorithms, which led to inaccurate transcriptions, they opted for manual transcriptions and translation via Amazon Mechanical Turk(AMT), a crowdsourcing marketplace that allows to do task that computers are still unable to complete. More in detail requests to AMT are called HIT(Human Intelligence Tasks). To ensure consistency, the authors divided speechs in chunks of around 30 seconds by also removing silent parts and sent each chunk 3 times as HIT. In this way they selected just HIT which had a correspondance. An example of transcription is shown in Figure 1.3 Qua potrei

0:14:44.190,0:14:45.310	0:00:02.780,0:00:04.640	0:04:37.880,0:04:39.620	0:06:40.669,0:06:41.669	0:12:28.000,0:12:28.000	0:00:03.280,0:00:06.000
pour tofu onto pan	open the bin	Take onion	pick up spatula	pour pasta into container	open fridge
0:14:45.310,0:14:49.540	0:00:04.640,0:00:06.100	0:04:39.620,0:04:48.160	0:06:41.669,0:06:45.250	0:12:33.000,0:12:33.000	0:00:06.000,0:00:09.349
put down tofu container	pick up the bag	Cut onion	stir potatoes	take jar of pesto	take milk
0:14:49.540,0:15:02.690	0:00:06.100,0:00:09.530	0:04:48.160,0:04:49.160	0:06:45.250,0:06:46.250	0 :12:39.000,0:12:39.000	0:00:09.349,0:00:10.910
stir vegetables and tofu	tie the bag	Peel onion	put down spatula	take teaspoon	put milk
0:15:02.690,0:15:06.260	0:00:09.530,0:00:10.610	0:04:49.160,0:04:51.290	0:06:46.250,0:06:50.830	0:12:41.000,0:12:41.000	0:00:10.910,0:00:12.690
put down spatula	tie the bag again	Put peel in bin	turn down hob	pour pesto in container	open cupboard
0:15:06.260,0:15:07.820	0:00:10.610,0:00:14.309	0:04:51.290,0:05:06.350	0:06:50.830,0:06:55.819	0:12:55.000,0:12:55.000	0:00:12.690,0:00:15.089
take tofu container	pick up bag	Peel onion	pick up pan	place pesto bottle on table	take bowl
0:15:07.820,0:15:10.040	0:00:14.309,0:00:17.520	0:05:06.350,0:05:15.200	0:06:55.819,0:06:57.170	0:12:58.000,0:12:58.000	0:00:15.089,0:00:18.080
throw something into the bin	put bag down	Put peel in bin	tip out paneer	take wooden spoon	open drawer

Figure 1.3: Extracts from 6 transcription files in .sbv format

aggiungere ancora qualcosa a pag 7 del paper in cui descrivono come ogni HIT è composta da 10 consecutive narrated phrases... in più 4 annotators per ogni HIT -> Overlap regions come in Figure 1.4 o magari modificare immagin senza fare veedere $\alpha()$

In the end they collected 39,596 action narrations, corresponding to a narration every 4.9s in the video. These narrations gave them a good starting point for labelling all actions with a rough temporal alignment, obtained from the timestamp of the audio narration with respect to the video, but still were not perfect. Infact:

- The narrations can be incomplete. So only narrated action will be considered in evaluation.

- The narration can be belated, after the action takes place.

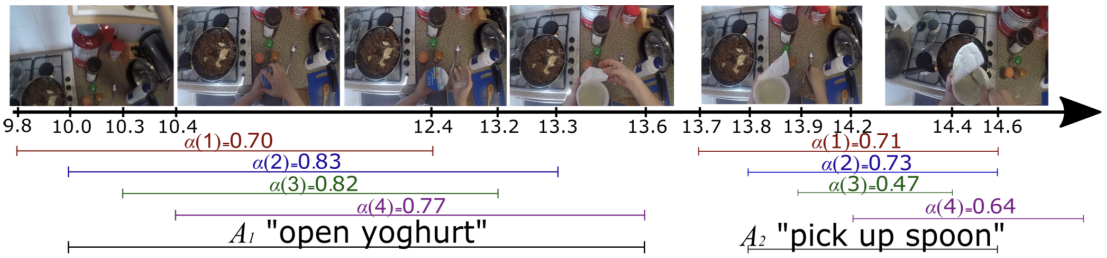

Figure 1.4: Example of annotated action segments for 2 consecutive actions

- The narration consists of participants' vocabulary and free language. Similar terms have been grouped in minimally overlapping classes.

It is worth adding few words about verb and noun annotations. Due to the freedom of terms and language, a variety of verbs and nouns have been collected. To reduce the number of them they grouped these into classes with minimal semantic overlapping. More in detail, as regards verbs they tried using automatic tools to cluster them but ended up manually clustering, due to the inefficient results; on the other hand for nouns they semi-automatically cluster them, preprocessing the compound nouns e.g. "pizza cutter" as a subset of the second noun e.g. "cutter" and also manually adjusting the clustering, merging the variety of names used for the same object, e.g. "cup" and "mug". In total they obtained 125 verb classes and 331 noun classes. In Figure 1.5 we can see some examples of grouped verbs and nouns into classes, while in Figure 1.6 the authors show the verb classes oredered by frequency of occurrence in action segments, as well as the noun classes ordered by number of annotated bounding boxes.

	ClassNo (Key)	Clustered Words
VERB	0 (take)	take, grab, pick, get, fetch, pick-up, ...
	3 (close)	close, close-off, shut
	12 (turn-on)	turn-on, start, begin, ignite, switch-on, activate, restart, light, ...
NOUN	1 (pan)	pan, frying pan, saucepan, wok, ...
	8 (cupboard)	cupboard, cabinet, locker, flap, cabinet door, cupboard door, closet, ...
	51 (cheese)	cheese slice, mozzarella, paneer, parmesan, ...
	78 (top)	top, counter, counter top, surface, kitchen counter, kitchen top, tiles, ...

Figure 1.5: Sample Verb and Noun Classes

In addition to verb and nouns annotations they also provide active object bounding box annotations. Similarly to verbs and nouns, they use AMT also for this task. Where each HIT aims to get an annotation for one object, for the maximum duration of 25s, which corresponds to 50 consecutive frames at 2fps. The

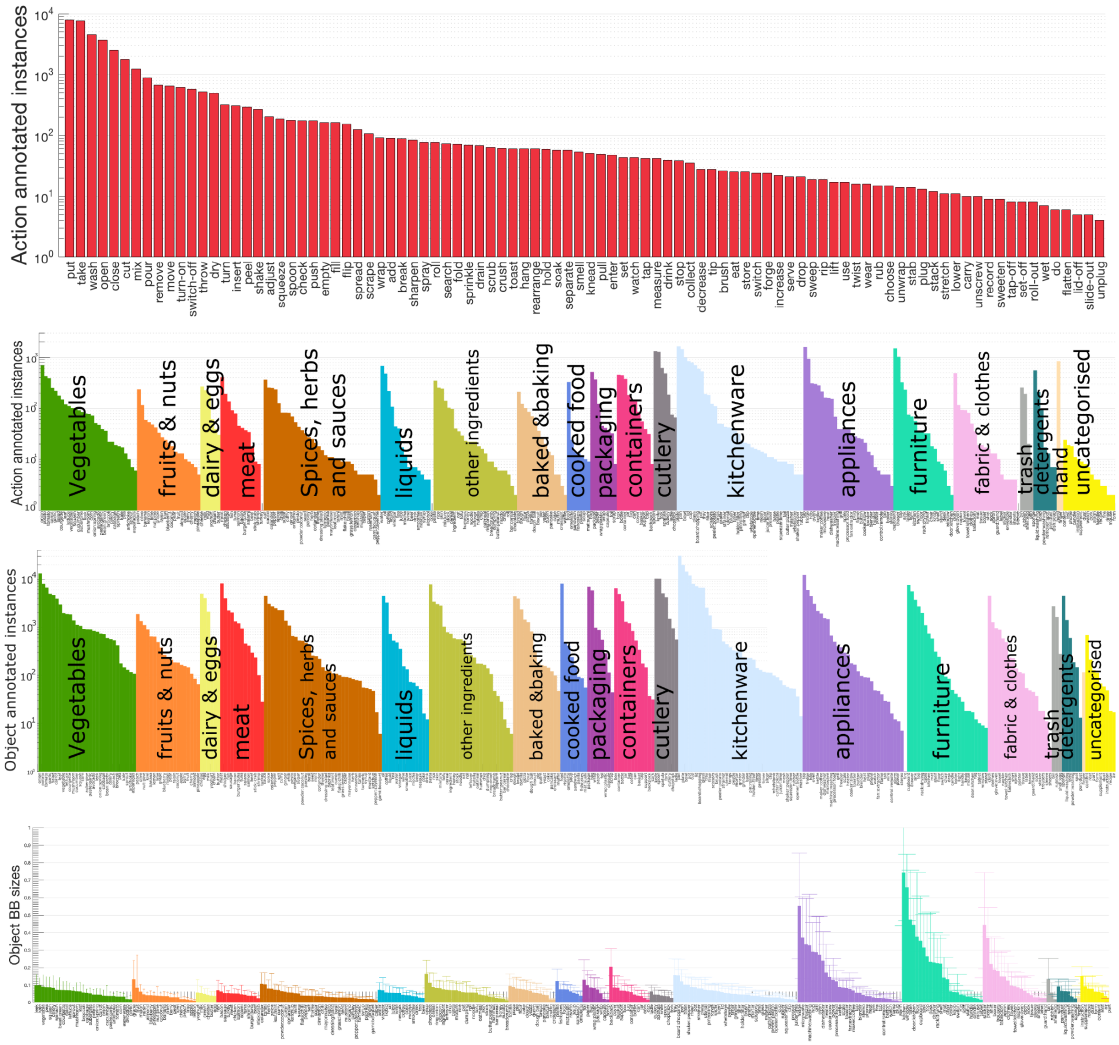

Figure 1.6: From Top: Frequency of verb classes in action segments; Frequency of noun clusters in action segments, by category; Frequency of noun clusters in bounding box annotations, by category; Mean and standard deviation of bounding box, by category

annotator can also state that the object is inexistent in at frame *f*. In total they collected 454,255 bounding boxes, some examples are provided in Figure 1.7.

1.1.4 Benchmarks and Baseline Results

The introduction of a new video datasets implies a variety of potential challenges that were not available before. Some of these are routine understanding, activity

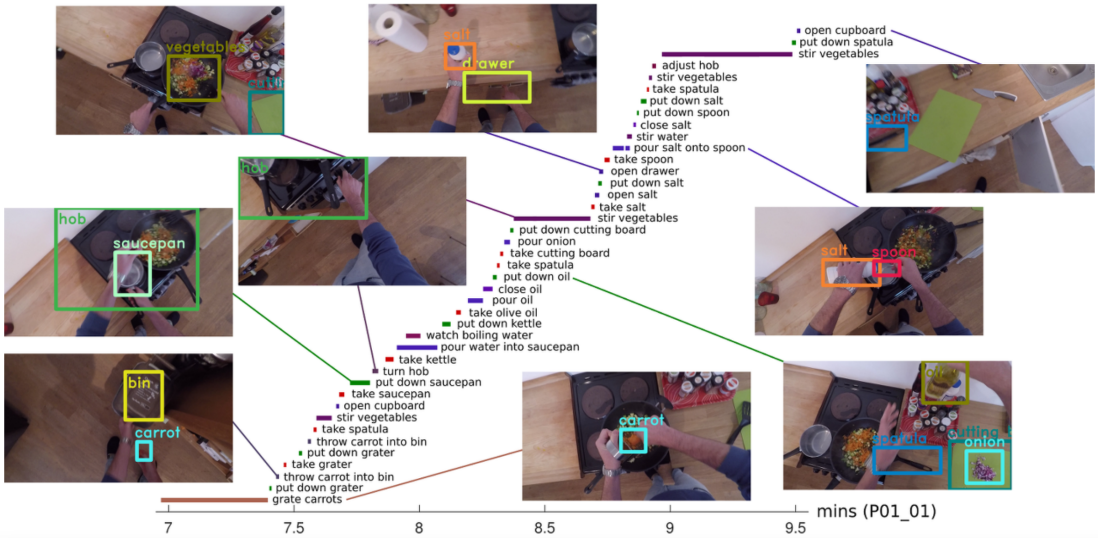

Figure 1.7: Sample consecutive action segments with keyframe object annotations

recognition and object detection. To spur the beginning the authors define the previous stated three challenges, providing baseline results. Let us see the challenges in more detail.

Action Recognition Challenge

Provided a trimmed action segment, the challenge requires to recognize what action class is performed, detecting the pair of verb and noun classes that compose the action. To participate to the challenge is asked to test the model on both splits[1] and for each test segment report the econfidence scores for each verb and noun class. In Figure 1.8 is reported a qualitative example of the task.

Action Anticipation Challenge

Provided an anticipation time, which is 1s before the action starts, the challenge consists in classifying the future action into its action class composed of the pair of

[1]To test the generalizability to novel environments they structured the test set to have a collection of *seen* and *unseen* kitchens.

- **Seen Kitchens (S1):** in this split each kitchen is seen in both training and testing.
- **Unseen Kitchens (S2):** This divides the participants/kitchens so all sequences of tge same kitchen are either in training or testing.

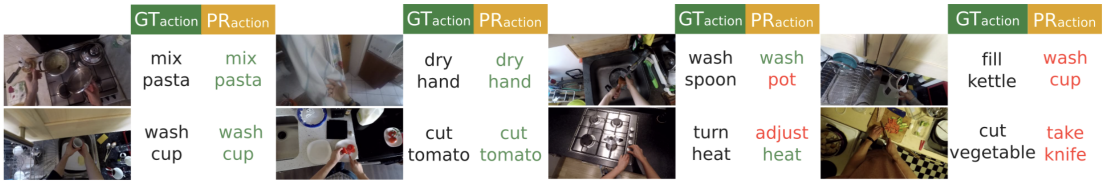

Figure 1.8: Sample qualitative results from the challenge's baseline of the Action Recognition Task

verb and noun classes. To participate to the challenge is asked to test the model on both splits and for each test segment report the econfidence scores for each verb and noun class. In Figure 1.9 is reported a qualitative example of the task.

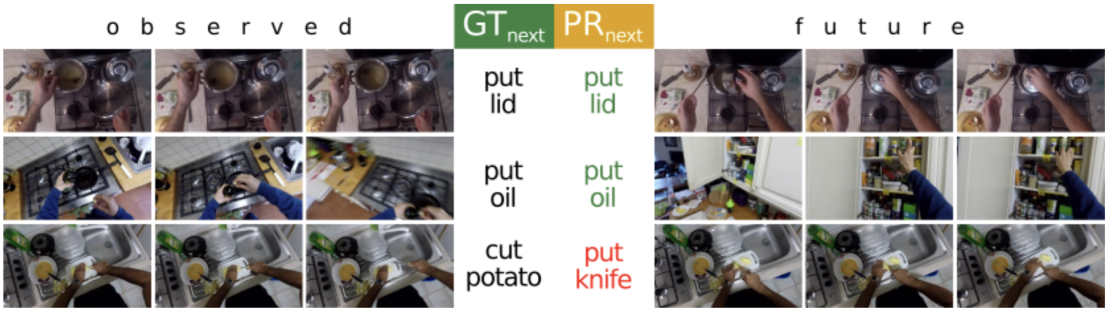

Figure 1.9: Sample qualitative results from the challenge's baseline of the Action Anticipation Task

Object Detection Challenge

In this challenge is required to perform object detectino and localisation. It must be noted that the annotations captured only *active* objects, namely objects involved in the action. To partecipate is required to provide predicted bounding boxes and their confidence scores on both dataset splits. In Figure 1.10 a qualitative example is reported.

1.1.5 Dataset Release

- Dataset sequences, extracted frames and optical flow are available at:

  `http://dx.doi.org/10.5523/bris.3h91syskeag572hl6tvuovwv4d`

- Annotations, challenge leader-board results and updates and news are available at `http://epic-kitchens.github.io`

Figure 1.10: Sample qualitative results from the challenge's baseline of the Object Detection Task

1.2 EPIC-Kitchens 100

In 2021 with [16] a new pipeline is introduced to extend EPIC-Kitchens dataset. EPIC-KITCHENS-100 collects 100 hours, 20M frames, 90k actions in 700 variable-length videos, capturing longterm unscripted actions in 45 different environments using headmounted cameras. Due to its novel annotation pipeline, which will be described more in detail later, more complete annotations of fine-grained actions are available, allowing the creation of new challenges such as: action detection[2] ,cross-modal retrieval(e.g.Audio-Based Interaction Recognition) and domain adaptation[3].

1.2.1 Motivation

The introduction of EPIC-KITCHENS has transformed egocentric vision, showcasing the unique potential of first-person views for action recognition and in particular hand-object interactions. To continue on this previously marked path they decided to enlarge EPIC-KITCHENS, mantaining the *unscripted* and *unedited* object interactions nature. In fact, the unscripted characteristics make the dataset results in a unbalance of data, with novel compositions of actions in new environments, making it a challenging dataset for domain adaptation.

[2]Action detection involves both recognizing the action and localizing the temporal intervals and spatial regions where the actions occur in a video.

[3]Training on a domain, e.g a specific kitchen, and test on another domain, e.g. a kitchen of a different person.

The most important novelty is the new annotation pipeline which allows to obtain denser and more complete actions' annotations in the recorded videos, enabling different task on the same dataset.

1.2.2 Data Collection

The additional videos were obtained by half of the previous participants, 16 persons, half of the 32 previously involved, and 5 new additional subjects. In the end the total participants reached 37 and the different kitchens were 45.

The new request for the subjects was to reecord 2-4 days of their kitchen routine.

1.2.3 Annotation

An overview of the pipeline taken from the paper is reported in Figure 1.11.

Narrator

The non-stop audio narration has been replaced with a *pause-and-talk* approach. By pausing the narrator can propose an initial temporal "pointing" but mostly avoids to miss or misspoke some actions due to lack of time. He does not have to narrate past actions while watching future actions, so short and overlapping actions are easier to be annotated.

For this an interface was built for the participants, it can be seen in Figure 1.11(a). An important new feature is the possibility to re-record and to delete a narration.

Transcriber

Each narration is first transcribed and then translated in English by a hired translator for correctness and consistency. The transcription process have been facilitated by providing a new transcriber interface showing three images sampled around the time stamp. As a matter of fact, in the old EPIC-Kitchens transcriptor struggled to understand some of the narration wwithout any video context.

Each narration was analyzed by 3 AMT workers using a consensus of 2 or more workers. A transcription was rejected if its Word2Vec **??** embeddings was lower than a threshold of 0.9. In case of consensus failure, the transcription was selected manually.

Parser

They used spaCy (`https://spacy.io`) to parse the transcriptions into verbs and nouns. Then they manually grouped those into minimally overlapping classes.

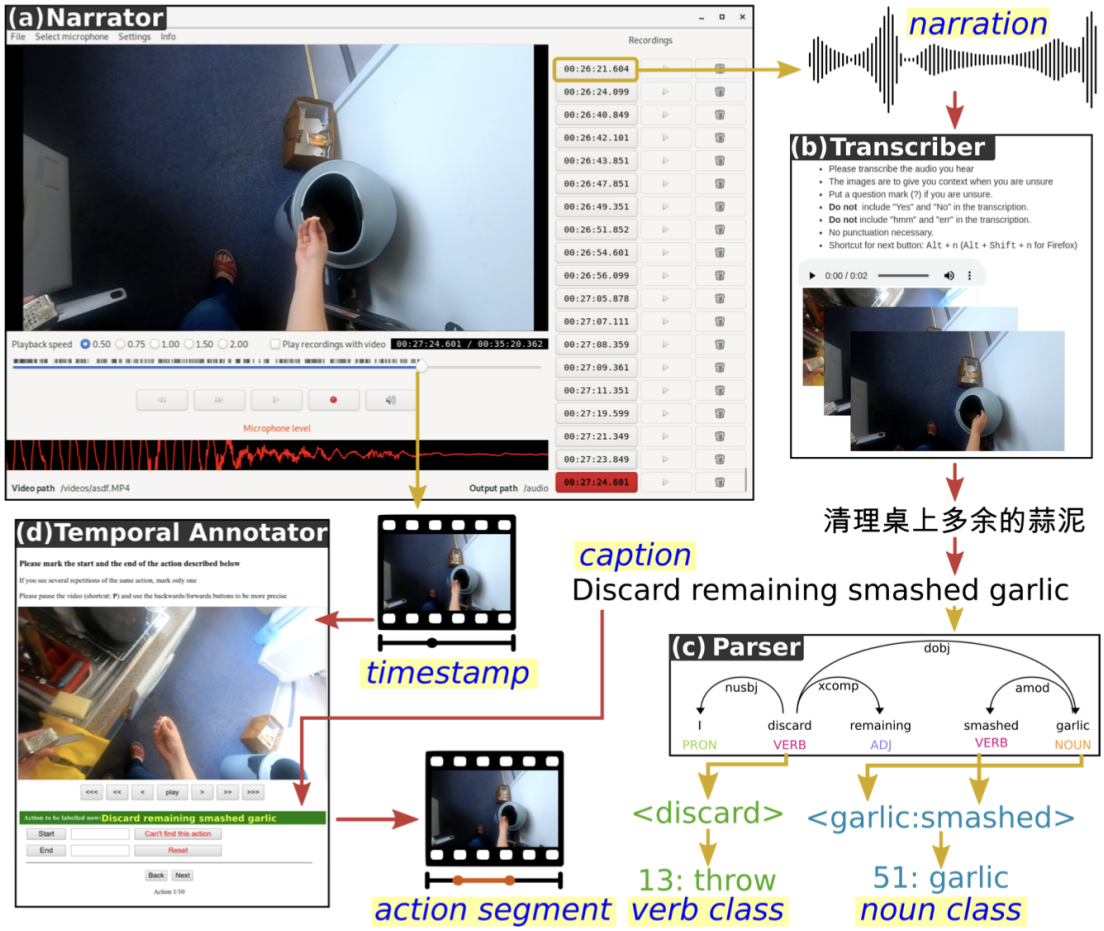

Figure 1.11: Annotation pipeline: **a** narrator, **b** transcriber **c** temporal segment annotator and **d** dependency parser. Red arrows show AMT crowdsourcing of annotations.

Temporal Annotator

They built a AMT interface for the start/end times of action segments(see Figure 1.11.d). To improve the quality this time the number of workers were increased from 4 to 5.

1.2.4 Quality Improvements

The attentions cared during the annotation process led to denser and more accurate annotations. We can see the results by comparing the same action and their respective annotation from the two different pipelines in Figure 1.12.

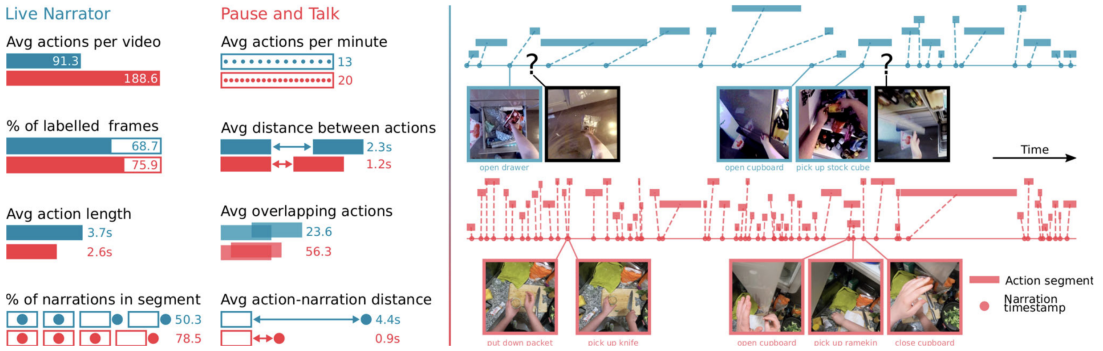

Figure 1.12: Comparing non-stop narrations (blue) to 'pause-and-talk' narrations (red). Right: timestsamps (dots) and segments (bars) for two sample sequences. "pause-and-talk" captures all actions including short ones. Black frames depict missed actions.

1.2.5 Challenges and Baselines

With respect to EPIC-Kitchens, 4 new challenges have been added.Magari non lo metto?

1.3 EPIC-Fields:Lo metto?Anche se non l'abbiamo usato?

The necessity of suitable datasets and benchmarks in the unified problem of 3D geometry and video understanding, which has been pushed by Neural Rendering (See Section 2), led to the rise of EPIC Fields. EPIC Fields is a expanded version of EPIC-KITCHENS comprehending 3D camera information. 96% of EPIC-KITCHENS videos were reconstructed, registering 19M frames in 99 hours recorded in 45 kitchens.

EPIC-KITCHENS is suited for studying the unified problem of geometric reconstruction and semantic understanding. As a matter of fact egocentric videos are relevant to mixed and augmented reality applications which are spreading in the last years, and the videos probe dynamic neural reconstruction due to their length (up to one hour) and to their dynamic nature.

Anyway obtaining camera information from EPIC-KITCHENS is difficult due to the complexity of its videos. Removing this step the authors try to ease the research in marrying 3D geometry to video understanding.

In conclusion they made two contributions:

- Intelligent Subsampling of frames for SfM algorithms

- Introduce a set of benchmark tasks:
  - dynamic novel view synthesis: reconstruct the same scene from a different point of view.
  - identifying independently from the camera moving objects
  - segmenting independently from the camera moving objects
  - video object segmentation.

1.3.1 Data

Some of past egocentric datasets [17, 18] contain static 3D scans of the environment, separately reconstructed from the actions. This additional step is an additional expense both in time and money, since the reconstructions are done with some dedicated costly hardware. In this work they provide a pipeline to extract the geometric reconstruction of the scene by just processing the egocentric video. EPIC Fields extends EPIC-KITCHENS(See Section 1.2) to include camera pose information. For each frame camera extrinsics and intrinsics parameterrs are provided, which enable tasks like 3D reconstruction. In total the succesfully processed 671 videos resulting in 18,790,333 registered video frames with estimated camera poses.

Motivation.

The 3D reconstruction could help recognizing different actions. Some actions could be located in the same 3D spot, e.g washing the dishes at the sink. Also the construction of tihs dataset could enable studying the relevance of 3D egocentric trajectories to actions(for anticipation),objects(for understanding object state changes) and hand-object understanding.

Collection

Since EPIC-KITCHENS did not collect videos with 3D reconstruction in mind, its videos are difficult to reconstruct. In fact Structure from Motion algorithms take as assumption that the recorded scene is *static*, meaning that each object will always have the same posiiton in 3D. However kitchen's activities involve the movement of objects like ingredients or utensils, and above all the presence of the operating hands.

Some other difficulties are introduced by:

- the **length of videos**, which on average last 9 mins

- the **skewed distribution of viewpoints**: the time spent in different part of the scene is different. In particular we have alternating phases of small motion around hot-spots,e.g. washing dishes, and of fast motions, like taking something to finish some task.

The solutions to these problems were given by:

- **Intelligent subsampling** of video frames.
- Using **SfM** for reconstructing the filtered frames.
- **Registering remaining frames** to the reconstruction.

Filtering

The aim of this step is to reduce the number of frames while keeping enough overlapping viewpoints for accurate reconstruction while diminishing the viewpoint skew. Overlap is measured by estimating homographies on matched SIFT features. Given a homography H, we define visual overlap r as the fraction of image area covered by the quadrilateral formed by warping the image corners by H. Windows are formed greedily, finding runs of frames (i+1,...,i+k) with overlap $r \geq 0.9$ to the first frame i. Filtering discards about 81.8 % of frames

Sparse reconstruction

Once filtered, frames are fed to COLMAP(Its functioning is reported in Section 3.1.5).

Dense reconstruction, automated verification, and restart

The remaining frames are fed to COLMAP with initial reconstruction. The final reconstruciton is accepted if over 70% of frames are registered succesfully. In the end 631 videos were obtained.

In case of failure the threshold r is increase,e.g. r$\geq$0.95. This usually results in doubling the frames, but increasing success rate to 96%.

1.3.2 Benchmarks, Experiments and Results

The authors defined three new benchmarks to explore the combination of 3D and video understanding.

New-View Synthesis(NVS)

Given a reconstruction based on a subsample of frames, the goal is to predict new video frames based on their timestamps and camera parameters. The quality of the reconstruction is evaluated as proposed in [19],measuring Peak Signal-to-Noise Ratio (PSNR) of the reconstructed frames compared to the real ones, making the lack of a 3D ground-truth irrelevant.

Video and Frame Selection. Due to the computational expensive cost of Neural Reconstruction, they provided a benchmark of limited selection of videos, namely 50 for a duration of 14.7 hours and 2.86M registered frames.

Frame selection instead is needed to divided the data in train and evaluation splits. The evaluation frames were divided into three tiers of difficulty:

- **In-Action (Hard):** frames belonging to an annotated action segment. During an action it is likely that object in the scene are moved making it difficult to reconstruct.

- **Out-of-Action:** frames NOT belonging to an annotated action segment. These frames can be further divided in:

  - **Easy**: Frames for which exists a neighbouring frame in the training set. The temporal proximity should ease the process of reconstruction.
  - **Medium**: Frames for which do not exists a neighbouring frame in the training set.

Benchmark methods. Three different neural rendering techniques were used to illustrate their possibilities and limits in such challenging scenarios like EPIC Fields. The methods used were:

- **NeuralDiff [20]:** consists of three different NeRFs, each one tailored to a part of the scene: static background, moving foreground and the actor. See Section 2 for more details.

- **NeRF-W [21]:** extend NeRF abilities by learning a low latent space that can modulate scene appearance and geometry. As a results it can separates static and transient components.

- **T-NeRF+ [22]:** time conditioned NeRF at which was added another NeRF to model the static background.

In Table 1.2 I report the authors' resulting experiments, while in Figure **??** we can see the comparison of the output of the three methods.

Method	Easy	Medium	Hard		
			All	BG	FG
NeRF-W [21]	21.13	19.3	17.93	18.99	13.54
T-NeRF+ [22]	21.58	19.81	18.44	19.73	13.74
NeuralDiff [20]	22.14	19.88	18.36	19.54	13.37

Table 1.2: Dynamic New View Synthesis. Comparison of different neural rendering methods on varying difficult frames. The values reported corresponds to PSNR considering all pixels in each test frame.

Unsupervised Dynamic Object Segmentation(UDOS)

In Unsupervised Dynamic Object Segmentation(UDOS) the objective is to find those regions in each frames that correspond to dynamic objects. The lack of a 3D ground-truth make 2D segmentation accuracy the only way to assess the model. In particular mean average precision (mAP) was used, as proposed in [20].

DEVO CONTROLLARE CHE EFFETTIVAMENTE VISOR NON ABBIA QUELLO CHE CI SERVE. forse qui in epic fields li hanno messi a posto in qualche modo... **Video and Frame selection.** The used videos were the same of NVS, with the difference that only In-Action frames where considered, with VISOR annotations as ground-truth. Actually VISOR annotations were processed in the following way: the original masks were converted into foreground-background masks in three different ways, depending on the type of objects present.

- **Dynamic objects only** setting:a dynamic object is an object that is currently being moved by visible hands.
- **Dynamic and semi-static objects** setting: objects that moved, not necessarily in the current frame, are semi-static objects.
- **Dynamic and semi-static excluding body parts** setting: active hands are excluded, as some methods overfit to predicting hands solely as dynamic objects, ignoring other moving objects.

As Baseline methods the authors used 4 methods: three based on 3D neural rendering techniques(NeRF-W,T-NeRF,NeuralDiff) and one based on 2D optical flow(Motion Grouping(MG) [23]. The results are shown in Figure 1.14 and Table 1.3.It is worth noting how 3D methods are better discovering semi-static objects. However none of the 3D methods explicitly consider motion. and this can be seen as MG performs better on purely dynamic motion, due to the input being the optical flow.

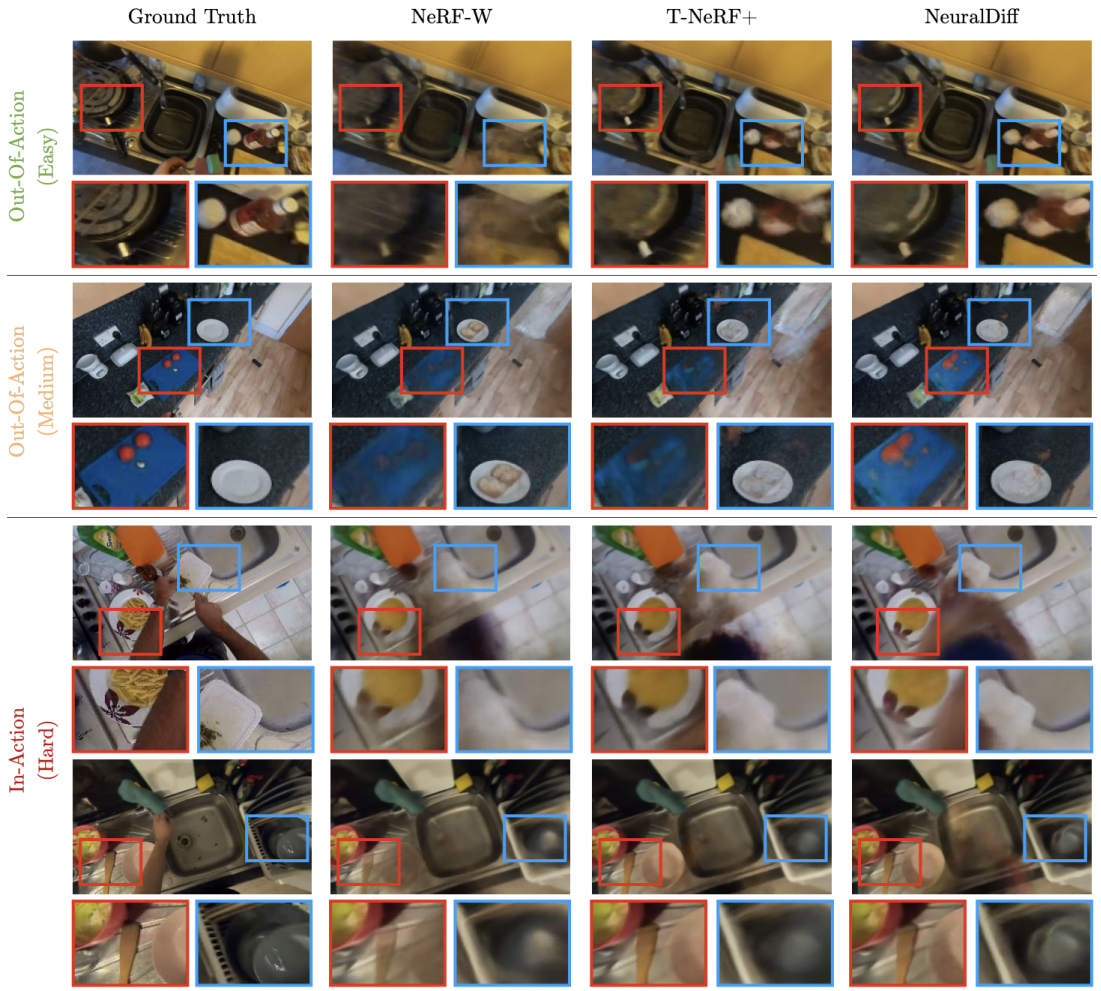

Figure 1.13: Dynamic New View Synthesis. I report an example of the output for the three different methods used: NeRF-W, T-NeRF+ and NeuralDiff. We can see how the initial labelling of difficulty for frames was actually accurate as the reconstructions struggle with Hard frames.

Semi-Supervised Video Object Segmentation(VOS)

Semi-Supervised Video Object Segmentation is a standard video understanding task which consists in propagating some given masks for one or more objects in a reference frame to the subsequent ones. Usually this task is performed by 2D models but here the authors show how integrating the third dimension could be beneficial. The idea is to project the 2D mask in 3D, fixing its position in the 3D scene and reproject it depending on the new camera position.

Two baselines were provided, a 2D and a 3D one. In Figure 1.15 we can see the

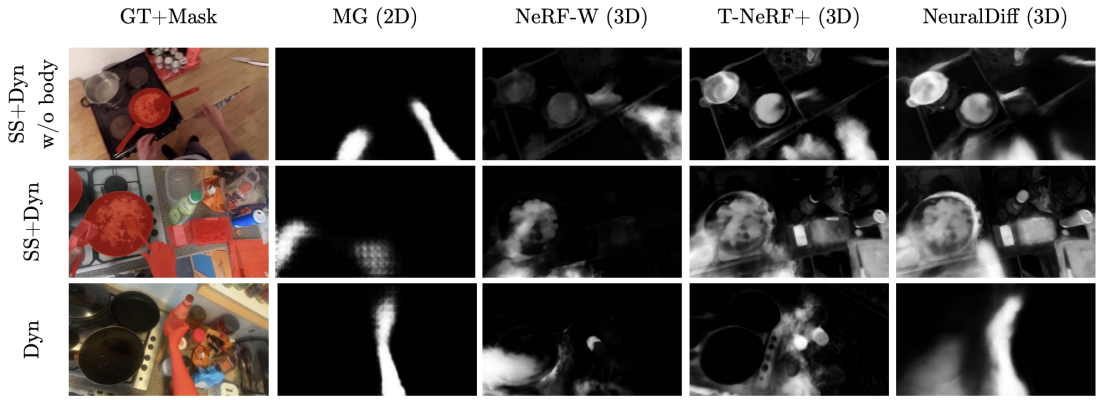

Figure 1.14: UDOS. Here is reported the comparison of the different methods' output. The 2D based perform very good on dynamic objects, while 3D methods struggle a bit but can detect even semi-static objects.

Method	3D	SS+Dyn	SS+Dyn (w/o body)	Dynamic
MG [23]	-	60.19	21.65	69.26
NeRF-W [21]	✓	37.26	22.96	27.41
T-NeRF+ [22]	✓	54.23	31.23	42.68
NeuralDiff [20]	✓	62.30	31.11	55.10

Table 1.3: UDOS. Here I reported the author results for UDOS. The values visible corresponds to the mAP on segmenting semi-static(SS) and dynamic(Dyn) components of the scene.

results and how the intuition previously described led to a good improvement.

1.4 VISOR Lo metto?non usato

Non va bene perchè gli oggeti che vengono segmentati sono solo quelli rilevanti all azione descritta, quindi se sposto un bicchiere mentre sto pelando le carote non viene segmentato il bicchiere -> non va bene per noi. Oppure oggetti attivi possono essere il lavandino, il tosta pane che sono però statici.

1.5 ((Ego4D))

Lo metto? Non lo ho usato per niente...?

Figure 1.15: VOS. Here is reported the comparison of the two different methods. The 3D method is clearly beter having as output something really close to the groundtruth. The 2D method instead is performing poorly.

Chapter 2

Neural Rendering

Chapter 3

Photogrammetry

Photogrammetry is the science and technology of obtaining reliable information about physical objects and the environment through the process of recording, measuring and interpreting photographic images and patterns of electromagnetic radiant imagery and other phenomena[24].

It comprises all techniques concerned with making measurements of real-world objects features from images. Its utility range from the measuring of coordinates, quantification of distances, heights, areas and volumes, preparation of topographic maps, to generation of digital elevation models and orthophotographs. The functioning rely mostly on optics and projective geometry rules.

As first assumption we have the modellization of the camera as a simplified version of itself: the *Pinhole Camera*. As in the first designed cameras(*camera obscura*), in the Pinhole Camera world's light is captured through a pinhole and then projected into the *focal plane*.

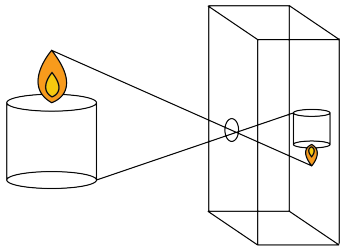

Figure 3.1: Pinhole Camera

A Pinhole camera is characterized by two collection of parameters:

- **Extrinsic** parameters: they give information on location and rotation in the world.

- **Intrinsic** parameters: gives us internal property such as: focal length, field of view, resolution etc.

These parameters can be rewritten in their corresponding matrices:

$$Intrinsic = K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

where f_x, f_y are are the *focal lengths* of the camera in the x and y directions, they are needed to keep the image aspect ratio; c_x, c_y are the coordinates of the *principal point* (the point where the optical axis intersects the image plane).

$$Extrinsic = \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{t}_{3x1} \\ 0_{1x3} & \mathbf{1}_{1x1} \end{bmatrix}$$

where:$\mathbf{R}_{3x3}$ is a rotation matrix; $\mathbf{t}_{3x1}$ is a translation vector.

with $\mathbf{R}$ that can be decomposed in its components:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \qquad R_y = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \qquad R_z = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Extrinsic matrix is also known as the 4x4 transformation matrix that converts points from the world coordinate system to the camera coordinate system.

Exploting homogeneous coordinates we can rewrite the image capturing process of a specific camera as the combination of its characteristic matrices:

$$\begin{bmatrix} u \\ v \\ z \end{bmatrix} = \begin{bmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{t}_{3x1} \\ 0_{1x3} & \mathbf{1}_{1x1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

3.1 Structure from Motion: SfM

Structure from motion (SfM) is the process of estimating the 3D structure of a scene from a set of 2D images. SfM is used in many applications, such as 3-D scanning, augmented reality, and visual simultaneous localization and mapping [25].

We can compute SfM in different ways depending on the data and tools at our disposal. Factors such as the **type** and the **number** of cameras can imply using different methods. Also the **ordering** of the frames may be relevant for a successful reconstruction.

The most basic scenario consists in two images captured from the same camera from two different point of view:

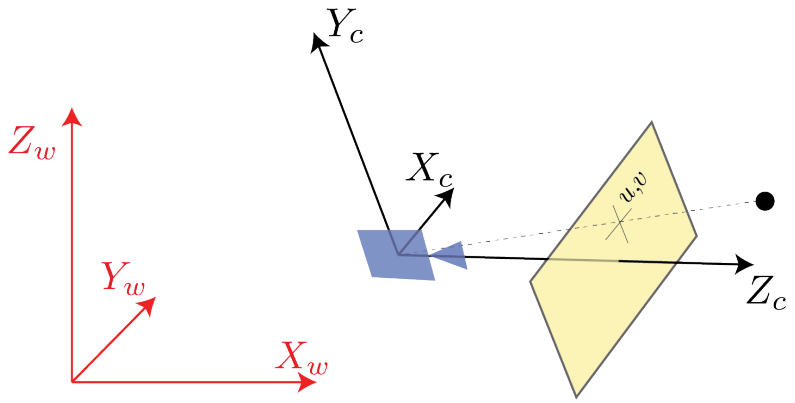

Figure 3.2: Reference systems

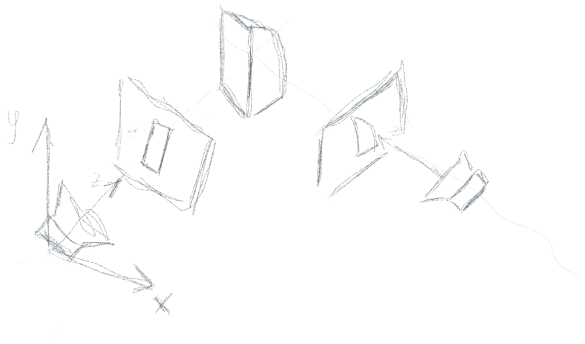

Figure 3.3: Basic SfM Scenario

In this case the 3D structure can be recovered *up to scale*, meaning that the relations between the obtained points is faithful to the reality, but it can differ from the real size by a *scale* factor. The scale factor could be retrieved if we know the size of an object in the scene or by having informations from other sensors.

In the basic scenario the method consists in the following main steps:

1. **Feature Detection** and **Matching**: consists in finding relevant points in each image and find the corresponding,if exist, in other images.

2. Estimating **Fundamental matrix**: which describes the geometric relationship

between two images of a scene taken from different viewpoints.

3. Estimating **Essential Matrix** from Fundamental matrix: which represents the intrinsic geometry of the scene, independent of the camera parameters.

4. Estimating **Camera Pose** from Essential Matrix: in this step the positions of the cameras in the world are extracted.

3.1.1 Feature Detection and Matching

In order to find corresponding points in two images we need to detect some points of interest in each image. Instead of trying to match each pixel, which would be computational inefficient and possibly misleading due to color, it has been shown succesful to use *feature points*. Feature points are relavant points which encapsulate the local appeareance of its surrounding pixels which is invariant under changes in illumination, translation, scale and in-plane rotation. Good features are *unique*, *can be easily tracked* and *can be easily compared*.

A practical example could be to imagine how us humans find correspondances in pictures, looking at Figure 3.4. If we would be asked to find the exact position of the six patches in the underlying image, the job would be easier for patches E and F, being corners they have less possible misleading correspondances, as happens instead for patches A or B.

Figure 3.4: Feature Detection Example [26]

In a similar way computers extract features. Two of the first algorithms are in fact based on corners detection: **Harris Corner Detector** and **Shi-Tomasi Corner Detector**. Depending on the differences between the two images to be compared, different algorithms have been developed. Here I describe some of them as reported in [26]:

- **SIFT** [27]: Harris corner detector is not good enough when scale of image changes. Lowe developed a breakthrough method to find scale-invariant features and it is called SIFT
- **SURF** (Speeded-Up Robust Features) [28]: faster SIFT version
- **FAST Algorithm for corner detection** All the above feature detection methods are good in some way. But they are not fast enough to work in real-time applications like SLAM. There comes the FAST algorithm, which is really "FAST".
- **BRIEF**(Binary Robust Independent Elementary Features) [29]:SIFT uses a feature descriptor with 128 floating point numbers. Consider thousands of

such features. It takes lots of memory and more time for matching. We can compress it to make it faster. But still we have to calculate it first. There comes BRIEF which gives the shortcut to find binary descriptors with less memory, faster matching, still higher recognition rate.

- **ORB(Oriented FAST and Rotated BRIEF)** [30]: Open source alternative to SIFT and SURF released by OpenCV.

Figure 3.5: SIFT features extracted

After having found these points of interest, we need to match them from the different pictures. This step is also known as *Feature Matching*. The most basics algorithms are:

- **Brute-Force Matcher**: Given a set of images, for each one the descriptor of one feature is compared with th one of every other image using a distance metric. The matched feature is the one whose distance is the smallest. It is called Brute-Force because it involves nearly no optimization, taking all possible combinations of images.

- **FLANN**: which stands for *Fast Library for Approximate Nearest Neighbors*. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features, such as hierarchical k-means clustering. It works faster than BFMatcher for large datasets. FLANN focuses on giving approximate nearest neighbor search rather than exact matches. Sacrificing some accuracy in favor of speed makes it suitable for large-scale datasets. In this way the feature matching problem becomes a clustering problem.

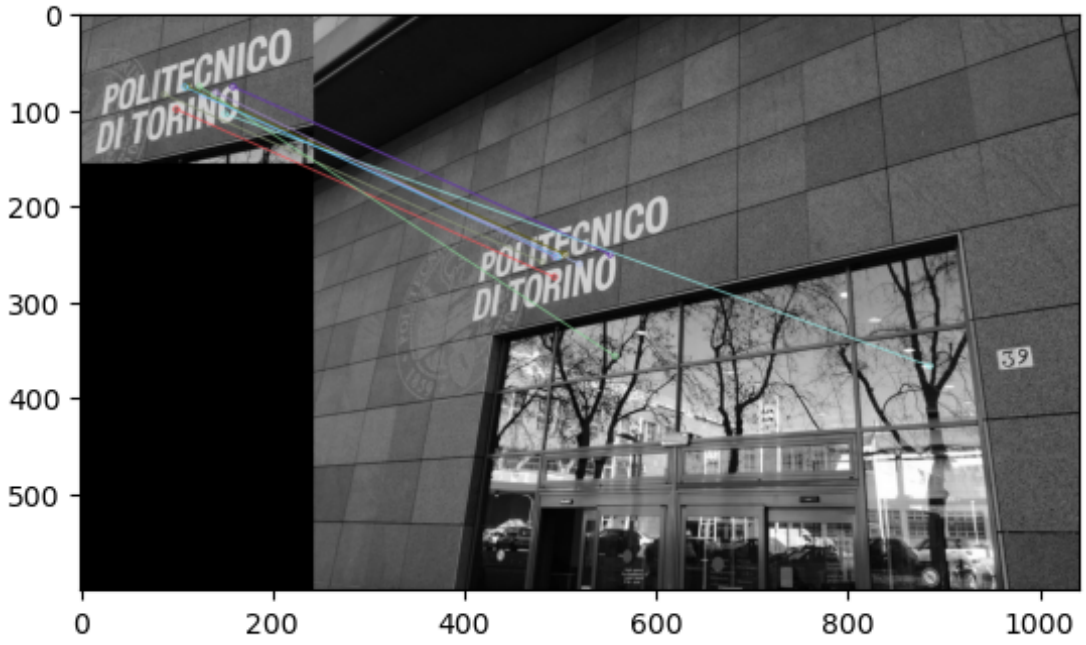

Figure 3.6: Features correspondence computed using BruteForce Matcher.

3.1.2 Estimating Fundamental and Essential Matrices

The Fundamental (F) and the Essential (E) matrices allow to relate the projection of a point located in space from one image to the other. These matrices are based on the so called *Epipolar Geometry*, which describes the relationship between two images. As we can see from Figure 3.7, given two cameras C_l and C_r the following

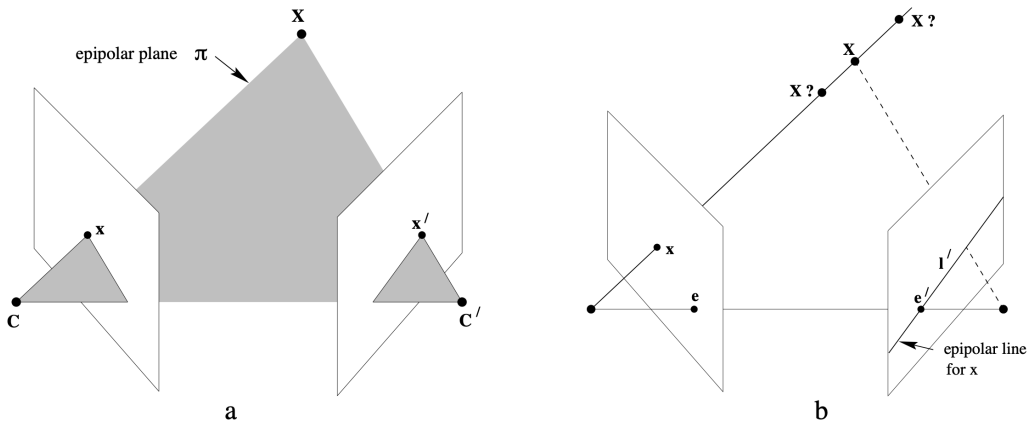

Figure 3.7: Point correspondence geometry.

definitions can be given:

- The **Epipole**: which is the point of intersection of the **baseline**(the line that connects the two camera centers C_l and C_r) with the image plane. In Figure 3.7 denoted by e_i.
- An **Epipolar plane**, which is any plane containing the baseline.
- An **Epipolar line** which is the intersection of any epipolar plane with the image plane.

Now that we have a clear understanding of the underlying geometry, let's proceed to see the actual derivation of the two matrices.

Let's consider the following scenario,reported in Figure 3.8: we have a point X and two cameras C_l and C_r. The relative positions are respectively x_l and x_r, while the pixel projections are p_l and p_r. If we consider as reference the left camera, the position of the right one is shifted of a vector $\mathbf{t}$.

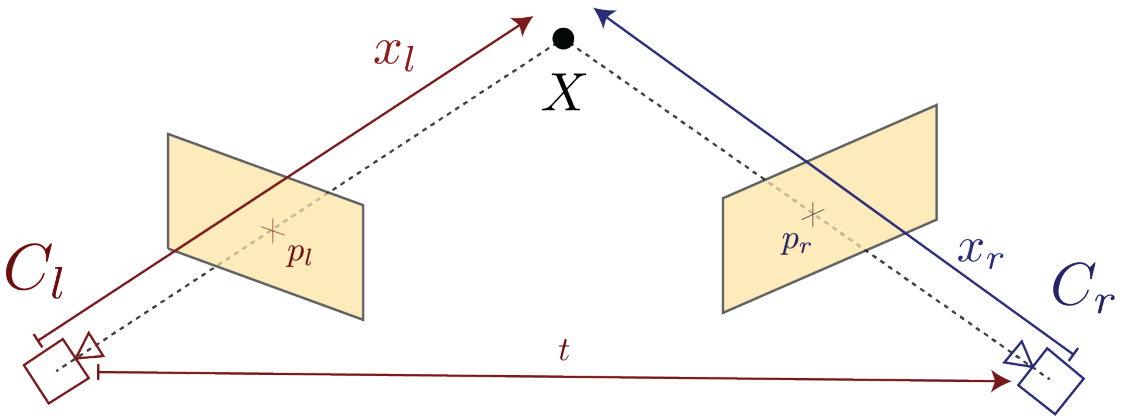

Figure 3.8

We can extract the Essential Matrix by making the following considerations:

$$x_l \cdot (t \times x_l) = 0 \tag{3.1}$$

but x_l can be written as:

$$x_l = Rx_r + t \tag{3.2}$$

So Equation 3.1 becomes:

$$x_l \cdot (t \times (Rx_r + t)) = x_l \cdot (t \times Rx_r) \stackrel{a}{=} x_l^T \, [t]_\times \, Rx_r = 0 \tag{3.3}$$

where equality (a) is due to the cross-product matrix notation[1]. We call Essential Matrix the term $[t]_\times R$, such that:

$$x_l^T [t]_\times R x_r = x_l^T E x_r = 0 \tag{3.4}$$

In a similar way we can get the Fundamental Matrix, by replacing the relative positions with the pixel positions. Recalling that the pixel positions are linked to the relative positions by:

$$p_l = \frac{1}{z_l} K_l x_l \quad p_r = \frac{1}{z_r} K_r x_r \tag{3.5}$$

where z_i are the focal distances.We can substitute in Eq. 3.4 and obtain:

$$p_l^T z_l {K_l^{-1}}^T E K_r^{-1} z_r p_r = 0 \quad z_l, z_r \neq 0 \tag{3.6}$$

Since z_i are constants can be simplified, obtaining:

$$p_l^T {K_l^{-1}}^T E K_r^{-1} p_r = p_l^T F p_r = 0 \quad z_l, z_r \neq 0 \tag{3.7}$$

where F is the Fundamental Matrix. We can thus write the relation between the two matrices:

$$E = K_l^T F K_r \tag{3.8}$$

3.1.3 Estimating Camera Pose from Essential Matrix

Since $[t]_\times$ is skew symmetric and R is orthonormal (since it is a rotation matrix), if we know the Essential Matrix we can decompose it in its components using singular value decomposition, from Equations 3.9, 3.10, 3.11

$$[t]_\times = U W \Sigma U^T \tag{3.9}$$

$$R = U W^T V^T \tag{3.10}$$

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.11}$$

[1]

$$\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

3.1.4 Multi-view Structure from Motion

Now that we have grasped the basics to find images correspondances, let's see how we can relate multiple image to recover the structure of a scene. The last stage is called *bundle adjustment* and it is a iterative algorithm used to adjust structure and motion parameters by minimising a cost function. The possible methods for bundle adjustment are:

- **Sequential**: it works by considering an additional images at each time, extending in this way the initial reconstruction.
- **Factorization**: it works by computing camera poses and scene geometry using every image measurement at the same time.

Bundle Adjustment is needed since the image measurements are usually noisy. Minimising an appropriate cost function we can obtain a clean model as in a Linear Regression.

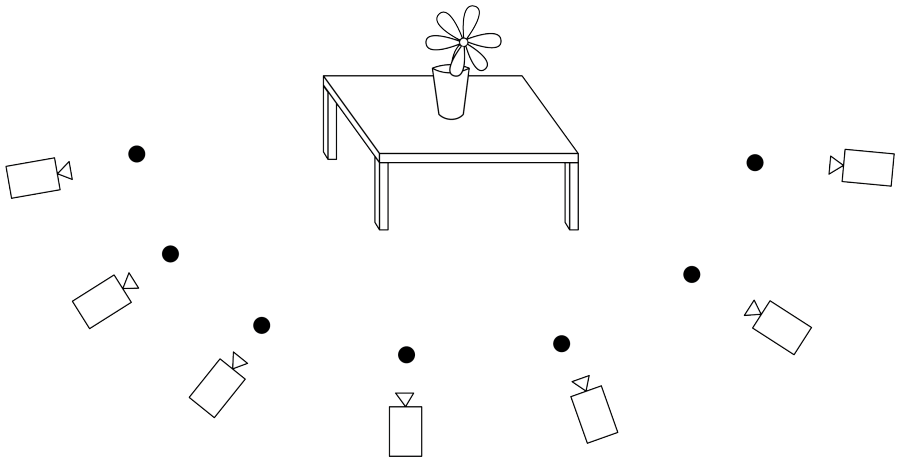

Figure 3.9: Multiple View Scenario.

3.1.5 COLMAP

Abandoning theory and moving on to practice, let's see COLMAP.

"COLMAP is a general-purpose, end-to-end image-based 3D reconstruction pipeline (i.e., Structure-from-Motion (SfM) and Multi-View Stereo (MVS)[2]) with

[2]*Even if they seem similar, these two pipelines have different goals. In fact, SfM The primary goal of SfM is to estimate the 3D structure of a scene and camera poses (positions and orientations) simultaneously from a set of 2D images taken from different viewpoints. MVS,on the other hand, is*

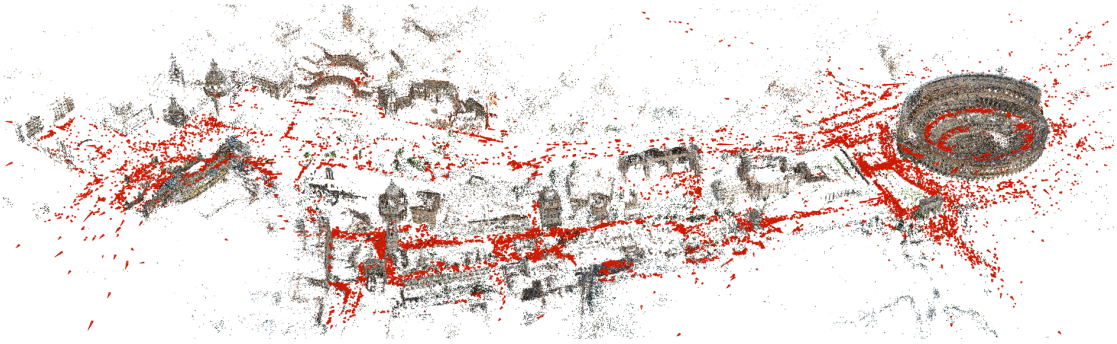

Figure 3.10: Building Rome in one day

a graphical and command-line interface. It offers a wide range of features for reconstruction of ordered and unordered image collections. The software runs under Windows, Linux and Mac on regular desktop computers or compute servers/clusters. COLMAP is licensed under the BSD License"[31].

The concepts we explained in the theoretical part are not always applicable In real world, or their results are not quite satisfying. In literature a vastity of authors tried and succeded in obtaining refined algorithms for specific scenarios. Anyway they still lacked a general-purpose method. With COLMAP they managed to compensate for the lack of generalization. The actual implementation and characteristics are explained from the authors in [32, 33].

Usage

The usage of COLMAP is pretty straight forward. After the creation of a project folder with a *images* directory containing the images we want to process we can simply launch the script reported in Listing (3.1).

Listing 3.1: Automatic COLMAP Reconstruction

```
#!/bin/bash
#The project folder contains a folder "images" with all images.

DATASET_PATH=/path/to/project
colmap automatic_reconstructor \
    --workspace_path $DATASET_PATH \
    --image_path $DATASET_PATH/images \
    --single_camera 1
```

specifically focused on dense 3D reconstruction. It involves estimating the depth or 3D coordinates for every pixel in the images to create a detailed 3D model with meshes and textures.

```
9
```

Anyway the program offers ample freedom to the single usage of the various steps need for *SfM* or *MVS*. Here I report the actual pipeline that I used for the extraction of the pointclouds for our experiments.

Listing 3.2: COLMAP Single Commands

```
1  #!/bin/bash
2  DATASET_PATH="/scratch/fborgna/EPIC_Diff/"
3
4  colmap feature_extractor \
5      --database_path $DATASET_PATH/database.db \
6      --image_path $DATASET_PATH/images \
7      --ImageReader.single_camera 1 \
8
9  colmap exhaustive_matcher \
10     --database_path $DATASET_PATH/database.db
11
12 mkdir $DATASET_PATH/sparse
13
14 colmap mapper \
15     --database_path $DATASET_PATH/database.db \
16     --image_path $DATASET_PATH/images \
17     --output_path $DATASET_PATH/sparse \
18     --camera_model SIMPLE_PINHOLE \
19
20 mkdir $DATASET_PATH/dense
21
22 colmap image_undistorter \
23     --image_path $DATASET_PATH/images \
24     --input_path $DATASET_PATH/sparse/0 \
25     --output_path $DATASET_PATH/dense \
26     --output_type COLMAP \
27     --max_image_size 2000
28
29 colmap patch_match_stereo \
30     --workspace_path $DATASET_PATH/dense \
31     --workspace_format COLMAP \
32     --PatchMatchStereo.geom_consistency true
33
34 colmap stereo_fusion \
35     --workspace_path $DATASET_PATH/dense \
```

```
        --workspace_format COLMAP \
        --input_type geometric \
        --output_path $DATASET_PATH/dense/fused.ply

    colmap poisson_mesher \
        --input_path $DATASET_PATH/dense/fused.ply \
        --output_path $DATASET_PATH/dense/meshed-poisson.ply


```

3.1.6 Monocular Depth Estimation

Until now we have always considered scenarios in which multiple images were available from different points of view. What if we have just one image?

This scenario takes the name of *Monocular Depth Estimation*. As we have previously seen, the projection of a scene on the bidimensional image plane of a camera inevitably lose the three-dimensional structure of the scene. In this way we can no more use optics laws, since we do not have enough informations to solve those equations.

Possible solutions include the usage of neural architectures. As a matter of fact, these methods try to learn relations between the possible objects or elements of the image, thus extracting a higher semantic knowledge from the picture. This task is known to be solved particularly good from neural networks, in particular convolutional or transformer based networks (See Section (2)), trained on large amount of data, which are nowadays growing in number and quality.

In particular the advent of transformers has marked a new state of the art, not only in natural language processing, but also in dense prediction. As shown and stated in [34], they "*introduce dense vision transformers, an architecture that leverages vision transformers in place of convolutional networks as a backbone for dense prediction tasks.*[...]*this architecture yieds substantial improvements on dense prediction tasks.*[...]*we observe an improvement of up to 28% in relative performance when compared to a state-of-the-art fully-convolutional network*". The motivation is due to the different functioning of the two networks. Infact, both methods implies an encoder-decoder structure but the nature of convolutional layer limits its performance. As explained in Sec.**??** convolutional networks progressively downsample the input image to extract features at multiple scales. Unfortunately in dense prediction tasks, keeping feature resolution and granularity showed to be essential, since the decoder can't recover the initial informations from the compressed ones. "*Unlike fully-convolutionals networks, the vision transformer backbone foregoes explicit downsampling operations after an initial image embedding*

has been computed and mantains a representation with constant dimensionality throughout all processing stages"[34].

The second motivation that made us choose for the transformer-based architecture is its versatibility and generalizability. In [35] the authors introduce a method that enable mixing multiple datasets during training. In this way the model will be effective across a variety of scenarios since each dataset is equally varied and captures the diversity of the visual world. In particular they experimented with eleven datsets which consists of RGB images with corresponding depth annotation of some form:

- Training: DIML Indoor,MegaDepth,ReDWeb,WSVD, 3D Movies
- Testing:DIW,ETH3D,Sintel,KITTI,NYUDv2,TUM-RGBD

Each single dataset has its own characteristic and has its own biases and problems. Mixing them during the train phase allows to mitigate those problems and obtain a good generalization. To evaluate the generalization they used the *Zero-Shot Cross-dataset Transfer* which consists in evaluating on datasets that were not seen during training. This proved to be extremely succesful.

Dataset	Indoor	Outdoor	Dynamic	Video	Dense	Accuracy	Diversity	Annotation	Depth	# Images
DIML Indoor	✓			✓	✓	Medium	Medium	RGB-D	**Metric**	220K
MegaDepth		✓	(✓)		(✓)	Medium	Medium	SfM	No scale	130K
ReDWeb	✓	✓	✓		✓	Medium	**High**	Stereo	No scale & shift	3600
WSVD	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	1.5M
3D Movies	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	75K
DIW	✓	✓	✓			Low	**High**	User clicks	Ordinal pair	496K
ETH3D	✓	✓			✓	**High**	Low	Laser	**Metric**	454
Sintel	✓	✓	✓	✓	✓	**High**	Medium	Synthetic	(Metric)	1064
KITTI		✓	(✓)	✓	(✓)	Medium	Low	Laser/Stereo	**Metric**	93K
NYUDv2	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	407K
TUM-RGBD	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	80K

Figure 3.11: Monocular datasets

Scene	Feature Extractor	Exhaustive Matcher	Mapper	Image Undistorter	Patch Match Stereo	Stereo Fusion	Total Time
A							
B							
C							
D							
E							
F							
G							

Table 3.1: Your Table Caption Here

Figure 3.12: Top: input images. Middle: Inverse depth maps predicted by the presented approach. Bottom: corresponding point clouds rendered from a novel view-point.(Taken from [35])

Scene	P01-01	P03-04	P04-01	P05-01	P06-03	P08-01	P09-02	P13-03	P16-01	P21-01
Succesful Reconstruction	✓	✓	✓	×	×	×	✓	×	✓	✓
Feature Extractor	00:00:02									
Exhaustive Matcher	00:00:23									
Mapper	00:46:37									
Image Undistorter										
Patch Match Stereo										
Stereo Fusion										
Total Time										

Table 3.2: Your Table Caption Here

Scene	**P01-01**	**P03-04**	**P04-01**	**P05-01**	**P06-03**	**P08-01**	**P09-02**	**P13-03**	**P16-01**	**P21-01**
Succesful Reconstruction	✓	✓	✓							
Feature Extractor										
Exhaustive Matcher										
Mapper										
Image Undistorter										
Patch Match Stereo										
Stereo Fusion										
Total Time										

Table 3.3: Your Table Caption Here

Part II

Da Sistemare

3.2 Metrics

CHIEDERE COME MOTIVARE LA SCELTA DELLE NOSTRE METRICHE
As regards metrics we looked in literature for a way to evaluate our results but unfortunately each method involved a ground truth which for our dataset is not available. Possible ways to obtain a groundtruth could be manual annotations or simulating the environments. Both these two methods would take a considerable large amount of time and are also beyond the scope of this thesis.

For this reason we ended up by using the metrics proposed in [20]. Namely these are:

- **PSNR**
- **AP**

3.2.1 PSNR:Peak signal-to-noise ratio

The Peak Signal-to-Noise Ratio (PSNR) is a metric commonly used in image and video processing to quantify the quality of a reconstructed or processed signal, like an image or video. It gives a measures of the ratio between the maximum possible power of a signal (MAX) and the power of the distortion or noise that affects the signal (MSE).

The formula for PSNR is usually expressed in decibels (dB) and is given by:

$$\text{PSNR} = 20 \cdot \log_{10} \left(\frac{\text{MAX}}{\text{MSE}} \right)$$

where:

- MAX is the maximum possible pixel value of the image (1 in our case).
- MSE is the Mean Squared Error, which represents the average squared difference between the original signal and the reconstructed or distorted signal.

It is worth noting that a high PSNR does not guarantee that the processed signal will be perceived as visually pleasing or high-quality by humans, especially in the case of perceptually sensitive applications like image and video compression.

3.2.2 AP:Average Precision

Average Precision (AP) is a metric commonly used in object detection and information retrieval to evaluate the performance of machine learning models. It measures the *precision-recall* trade-off of a model.

It can be useful to remind what *Precision* and *Recall* are. Namely:

$$Precision = \frac{TP}{TP + FP} \tag{3.12}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.13}$$

where:

- TP=True positive
- FP=False positive
- TN=True negative

Average precision is then computed as the area below the precision-recall curve, specifically the curve obtained by varying the confidence threshold of the inference model as shown in Figure 3.13. That is why it can also be found in literature as AUC(Area Under Curve). Its scalar value summarize the precision-recall performance of the model. A higher AP is desirable, indicating a model that effectively retrieves relevant instances while minimizing false positives.

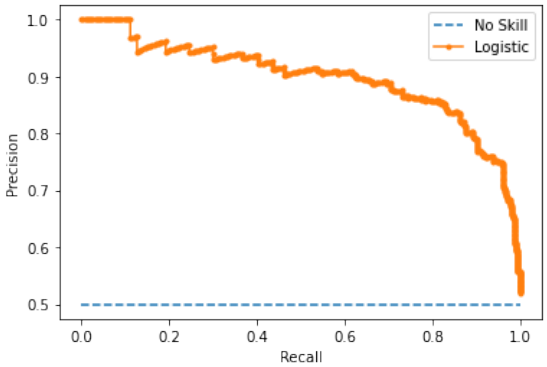

Figure 3.13: Example of Precision-recall curve.We can see how the bottom line model represents the worst a model can perform, e.g. predict every sample as it is coming from the same class,if the dataset is balanced. A better model would *tend* to the upper-right corner, which instead represents the best possible model, a model that have maximum precision and recall.

Scene	Feature Extractor	Exhaustive Matcher	Mapper	Image Undistorter	Patch Match Stereo	Stereo Fusion	Total Time
A							
B							
C							
D							
E							
F							
G							

Table 3.4: Your Table Caption Here

Scene	P01-01	P03-04	P04-01	P05-01	P06-03	P08-01	P09-02	P13-03	P16-01	P21-01
Succesful Reconstruction	✓	✓	✓	×	×	×	✓	×	✓	✓
Feature Extractor	00:00:02									
Exhaustive Matcher	00:00:23									
Mapper	00:46:37									
Image Undistorter										
Patch Match Stereo										
Stereo Fusion										
Total Time										

Table 3.5: Your Table Caption Here

Scene	P01-01	P03-04	P04-01	P05-01	P06-03	P08-01	P09-02	P13-03	P16-01	P21-01
Succesful Reconstruction	✓	✓	✓							
Feature Extractor										
Exhaustive Matcher										
Mapper										
Image Undistorter										
Patch Match Stereo										
Stereo Fusion										
Total Time										

Table 3.6: Your Table Caption Here

Appendix A

Galileo

```
    import os
    os.system("echo 1")
```

$\mathcal{O}\left(n \log n\right)$
`numpy`

Appendix B

Math Notation

$$\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_{\times} \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\mathbf{a} \times \mathbf{b} = [\mathbf{b}]_{\times}^{T} \mathbf{a} = \begin{bmatrix} 0 & b_3 & -b_2 \\ -b_3 & 0 & b_1 \\ b_2 & -b_1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Bibliography

[1] Dima Damen et al. «Scaling Egocentric Vision: The EPIC-KITCHENS Dataset». In: *European Conference on Computer Vision (ECCV)*. 2018 (cit. on p. 3).

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. «Deep Residual Learning for Image Recognition». In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (cit. on p. 3).

[3] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks». In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497 (cit. on p. 3).

[4] Andrej Karpathy and Li Fei-Fei. «Deep Visual-Semantic Alignments for Generating Image Descriptions». In: *CoRR* abs/1412.2306 (2014). arXiv: 1412.2306. URL: http://arxiv.org/abs/1412.2306 (cit. on p. 3).

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. «VQA: Visual Question Answering». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015 (cit. on p. 3).

[6] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. «The Pascal Visual Object Classes (VOC) Challenge.» In: *Int. J. Comput. Vis.* 88.2 (2010), pp. 303–338. URL: http://dblp.uni-trier.de/db/journals/ijcv/ijcv88.html#EveringhamGWWZ10 (cit. on p. 3).

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. «ImageNet: A large-scale hierarchical image database». In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848 (cit. on p. 3).

[8] Tsung-Yi Lin et al. «Microsoft COCO: Common Objects in Context». In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312 (cit. on p. 3).

[9] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. «Scene Parsing through ADE20K Dataset». In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5122–5130. DOI: 10.1109/CVPR.2017.544 (cit. on p. 3).

[10] Raghav Goyal et al. «The "something something" video database for learning and evaluating visual common sense». In: *CoRR* abs/1706.04261 (2017). arXiv: 1706.04261. URL: http://arxiv.org/abs/1706.04261 (cit. on p. 3).

[11] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. «YouTube-8M: A Large-Scale Video Classification Benchmark». In: *CoRR* abs/1609.08675 (2016). arXiv: 1609.08675. URL: http://arxiv.org/abs/1609.08675 (cit. on p. 3).

[12] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. «A Dataset for Movie Description». In: *CoRR* abs/1501.02530 (2015). arXiv: 1501.02530. URL: http://arxiv.org/abs/1501.02530 (cit. on p. 3).

[13] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. «MovieQA: Understanding Stories in Movies through Question-Answering». In: *CoRR* abs/1512.02902 (2015). arXiv: 1512.02902. URL: http://arxiv.org/abs/1512.02902 (cit. on p. 3).

[14] David F. Fouhey, Weicheng Kuo, Alexei A. Efros, and Jitendra Malik. «From Lifestyle Vlogs to Everyday Interactions». In: *CoRR* abs/1712.02310 (2017). arXiv: 1712.02310. URL: http://arxiv.org/abs/1712.02310 (cit. on p. 3).

[15] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. «Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding». In: *CoRR* abs/1604.01753 (2016). arXiv: 1604.01753. URL: http://arxiv.org/abs/1604.01753 (cit. on p. 3).

[16] Dima Damen et al. «Rescaling Egocentric Vision: Collection, Pipeline and Challenges for EPIC-KITCHENS-100». In: *International Journal of Computer Vision (IJCV)* 130 (2022), pp. 33–55. URL: https://doi.org/10.1007/s11263-021-01531-2 (cit. on p. 11).

[17] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F. Fouhey. *Understanding Human Hands in Contact at Internet Scale*. 2020. arXiv: 2006.06669 [cs.CV] (cit. on p. 15).

[18] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. *Large-scale interactive object segmentation with human annotators*. 2019. arXiv: 1903.10830 [cs.CV] (cit. on p. 15).

[19] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. 2020. arXiv: 2003.08934 [cs.CV] (cit. on p. 17).

[20] Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. «NeuralDiff: Segmenting 3D objects that move in egocentric videos». In: *CoRR* abs/2110.09936 (2021). arXiv: 2110.09936. URL: https://arxiv.org/abs/2110.09936 (cit. on pp. 17, 18, 20, 40).

[21] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. *NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections*. 2021. arXiv: 2008.02268 [cs.CV] (cit. on pp. 17, 18, 20).

[22] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. *Monocular Dynamic View Synthesis: A Reality Check*. 2022. arXiv: 2210.13445 [cs.CV] (cit. on pp. 17, 18, 20).

[23] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. *Self-supervised Video Object Segmentation by Motion Grouping*. 2021. arXiv: 2104.07658 [cs.CV] (cit. on pp. 18, 20).

[24] Wolf and Dewitt. 2000; McGlone, 2004 (cit. on p. 23).

[25] Mathworks 2023. URL: https://www.mathworks.com/help/vision/ug/structure-from-motion.html#:~:text=Structure%20from%20motion%20(SfM)%20is,localization%20and%20mapping%20(vSLAM). (cit. on p. 24).

[26] OpenCV 2024. URL: https://docs.opencv.org/4.x/df/d54/tutorial_py_features_meaning.html (cit. on p. 27).

[27] David G. Lowe. «Distinctive Image Features from Scale-Invariant Keypoints». In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf (cit. on p. 27).

[28] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. «Speeded-Up Robust Features (SURF)». In: *Computer Vision and Image Understanding* 110.3 (2008). Similarity Matching in Computer Vision and Multimedia, pp. 346–359. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2007.09.014. URL: https://www.sciencedirect.com/science/article/pii/S1077314207001555 (cit. on p. 27).

[29] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. «BRIEF: Binary Robust Independent Elementary Features». In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 778–792. ISBN: 978-3-642-15561-1 (cit. on p. 27).

[30] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. «ORB: An efficient alternative to SIFT or SURF». In: *2011 International Conference on Computer Vision*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544 (cit. on p. 28).

[31] Colmap 2024. URL: https://colmap.github.io/index.html (cit. on p. 33).

[32] Johannes Lutz Schönberger and Jan-Michael Frahm. «Structure-from-Motion Revisited». In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 33).

[33] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. «Pixelwise View Selection for Unstructured Multi-View Stereo». In: *European Conference on Computer Vision (ECCV)*. 2016 (cit. on p. 33).

[34] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. «Vision Transformers for Dense Prediction». In: *ICCV* (2021) (cit. on pp. 35, 36).

[35] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. «Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.3 (2022) (cit. on pp. 36, 37).