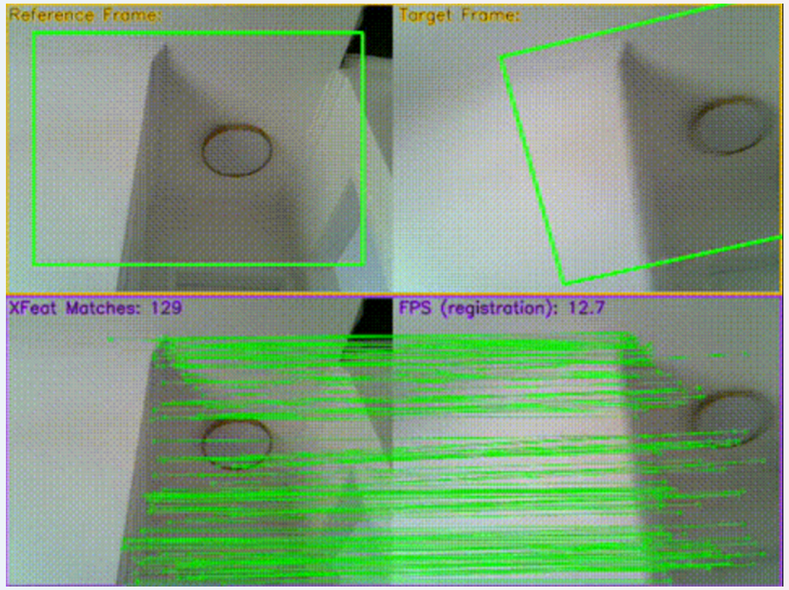
INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

XFeat: Accelerated Features for Lightweight Image Matching



CS7.505 - Computer Vision 2024-25

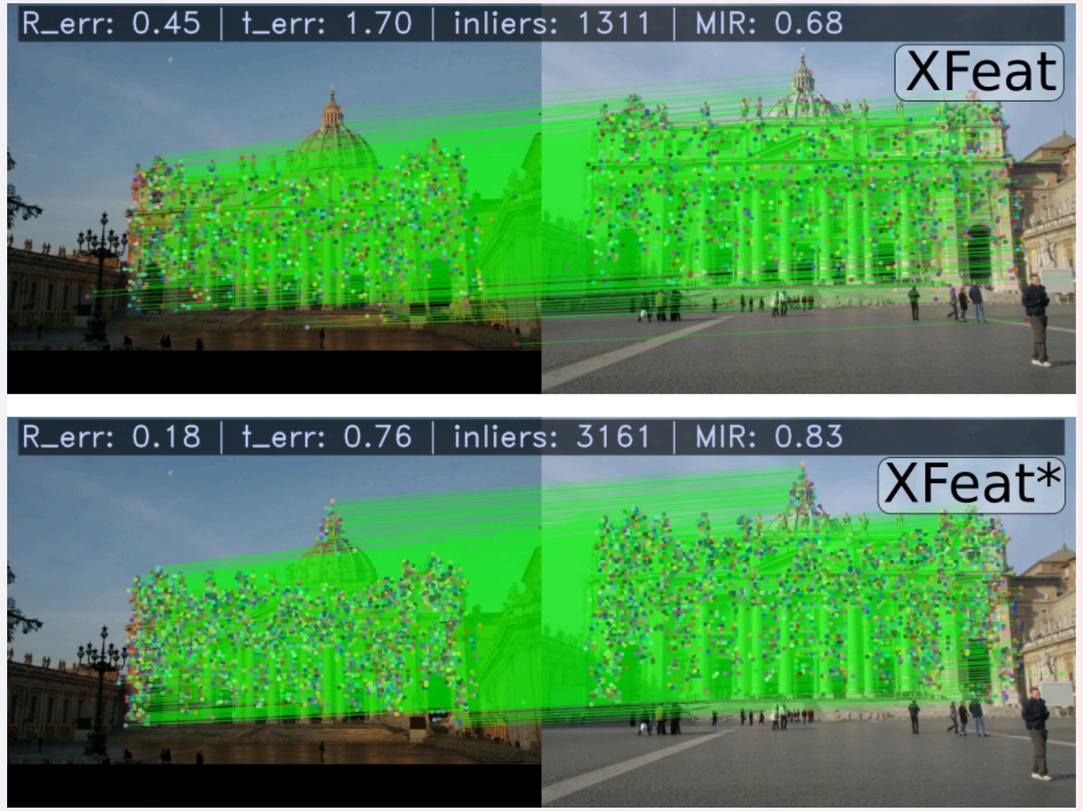
Aditya Raghuvanshi , Yash Bhaskar

Introduction

XFeat Objective: lightweight CNN for detecting, extracting and matching local features.

Matching Types:

1. **Sparse Matching (XFeat):**
   - Identifies salient points (or keypoints)
   - Optimized for computational efficiency
2. **Semi-Dense Matching (XFeat*):**
   - Enhances sparse matches by refining at pixel-level
   - Balances precision and resource efficiency



State of Art

Modern image matching balances **accuracy and efficiency**, with methods divided into **standard** for precision and **fast** for speed on constrained devices.

Standard

- Traditional Approaches (ex SiFT)
- Middle-End Approaches (ex SuperGlue)
- End-to-End Approaches (ex DISK)

👍**Performance**

👎**Efficiency**

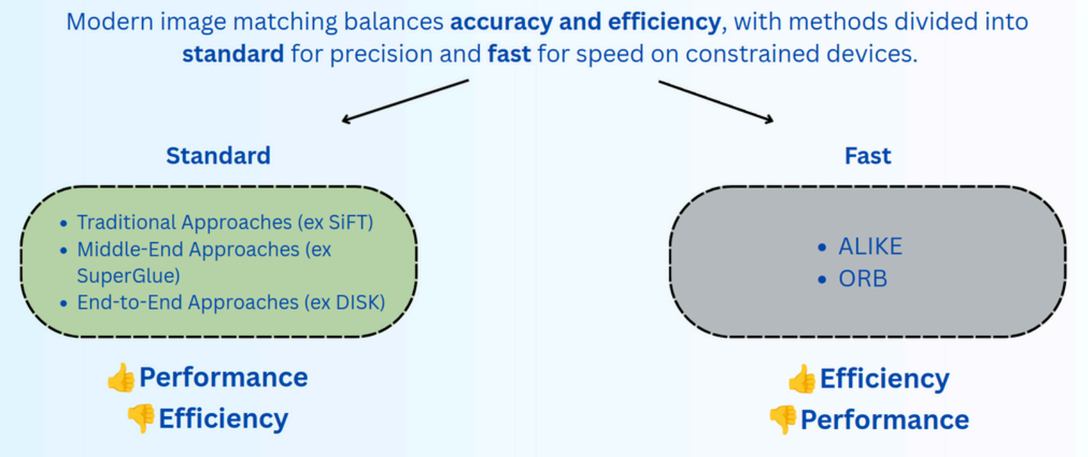
Fast

- ALIKE
- ORB

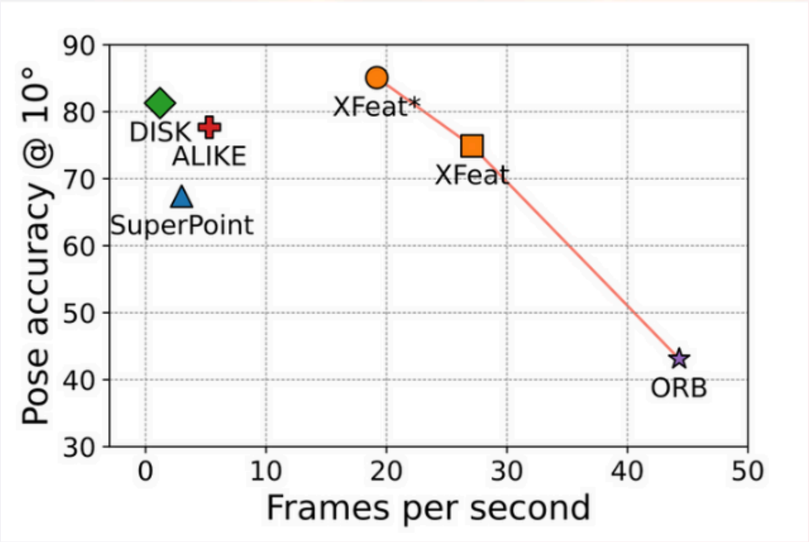
👍**Efficiency**

👎**Performance**

State of Art



As we can see XFeat reach the **best trade off between efficency and performance**



Xfeat (sparse matching)

Keypoint Detection

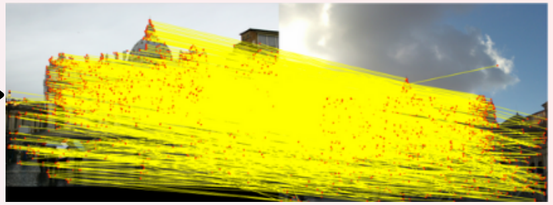
- Identifying **salient points** in the image
- Pixel level

Description & Reliability extraction

- **Description**: feature descriptors map for matching.
- **Reliability**: quantifies the confidence of each descriptor.

Matching

- Mutual Nearest Neighbor
- **Select** top keypoint
- **Match** keypoint by cosine similarity using their descriptors

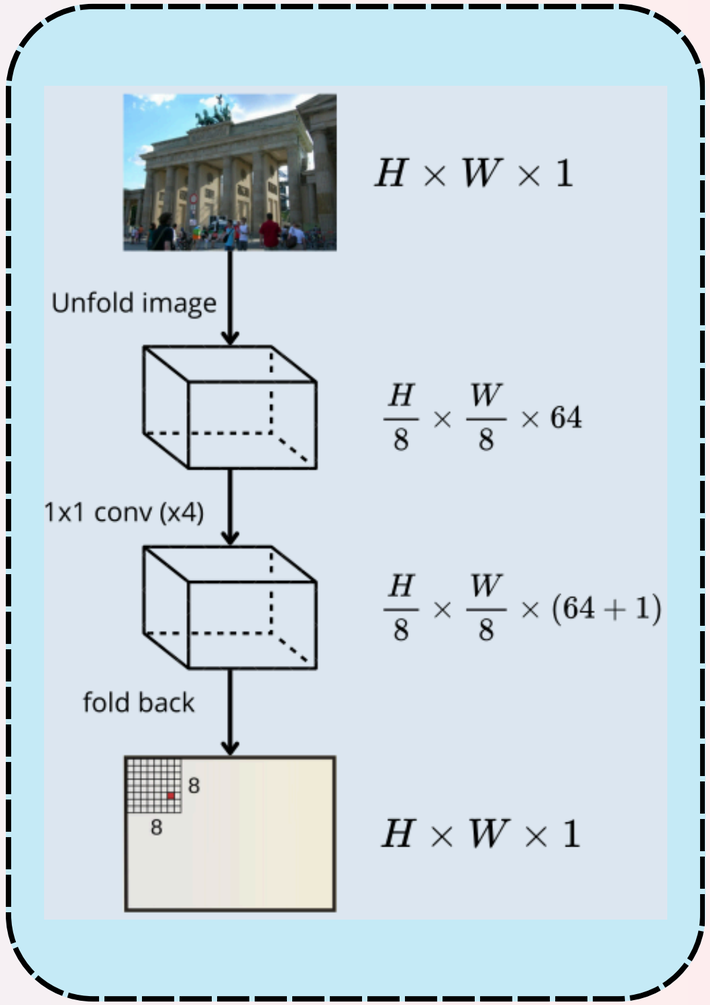


Keypoint Detection

Purpose: Find sparse features with minimal computational cost

Procedure:

1. **Divide the input image** in a 2D grid composed:
   - **8 × 8 pixels** on each grid cell
   - Reshape each cell into **64 channels**
2. **Apply 1×1 convolutions (x4)** to regress keypoint embeddings
   - Obtain $\mathbf{K} \in \mathbb{R}^{H/8 \times W/8 \times (64+1)}$
3. **Classify keypoints** within each cell or assign them to a "dustbin" if no keypoint is detected.
4. **Fold back** to image and they obtain the map of keypoints.



Description & Reliability extraction

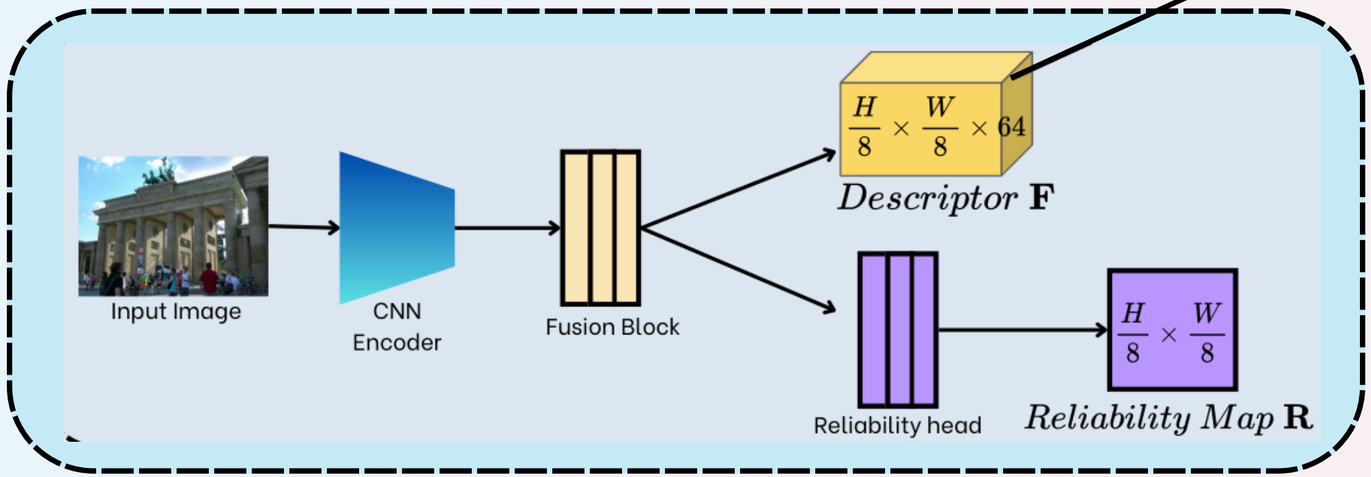
Purpose:

- Provides reliable **local descriptors** and **confidence measures** for efficient image matching.

Description:

- Use of a CNN backbone to obtain:
  - **Feature map** $F \in \mathbb{R}^{H/8 \times W/8 \times 64}$
  - **Reliability map** for feature matching confidence, $R \in \mathbb{R}^{H/8 \times W/8}$

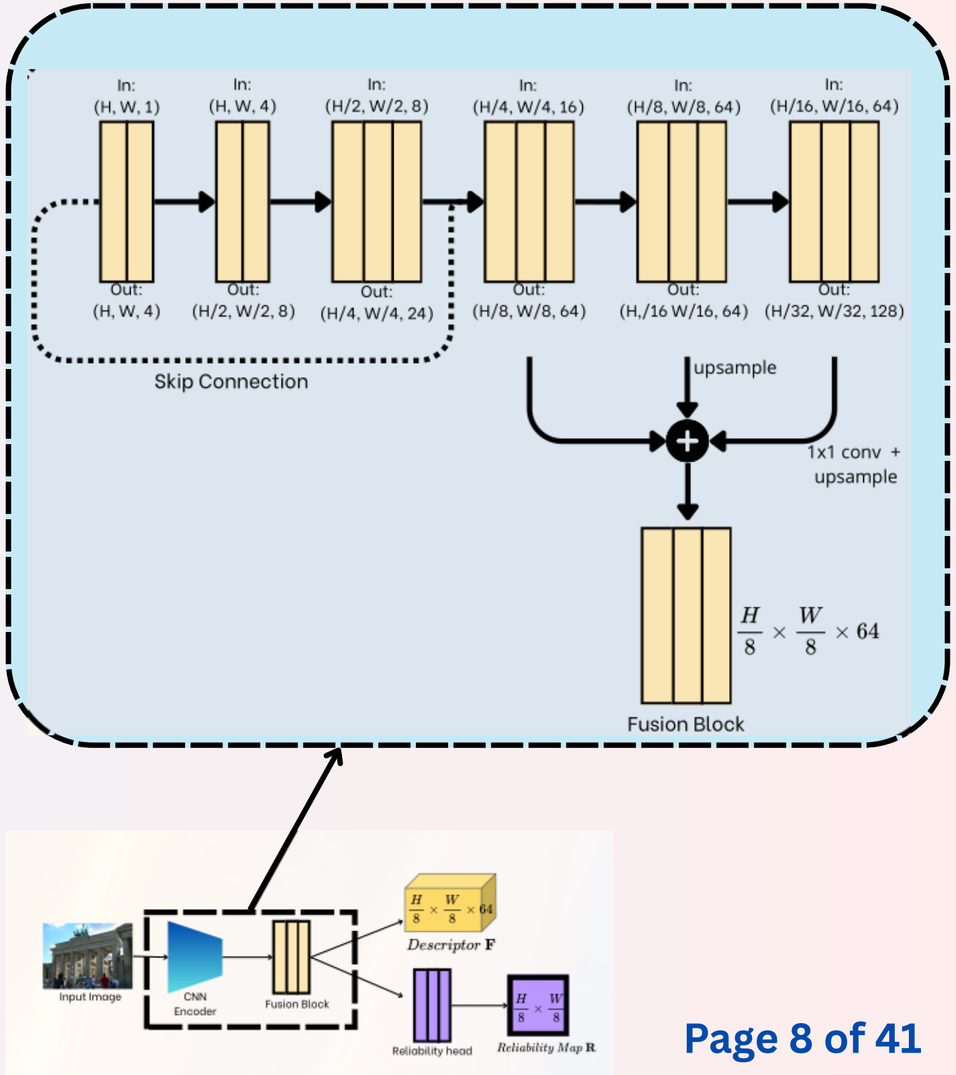
Associated to a Cell 8x8 pixels, not to a single pixel



Description Head - Backbone

Design:
The main bottleneck in CNN is **H*W*C** so to alleviate the problem, the backbone is implemented with these features:

- **Progressive Downsampling:** Reduces spatial resolution step-by-step
- **Channel Depth Management:** Dynamically adjusts channel depth
- **Multiscale Integration:** Merges features across multiple resolutions(1/8, 1/16, 1/32) to a common resolution (H/8×W/8)



Pipeline - xfeat* (semi- dense matching)

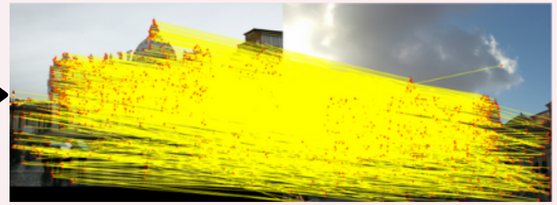
Description & Reliability extraction

- **Description**: feature descriptors map for matching.
- **Reliability**: quantifies the confidence of each descriptor.

Matching
(Mutual Nearest Neighbor)

Match Refinement

- Enhances at pixel-level precision for **semi-dense matching**.



N.B. NOT PIXEL LEVEL!!
CELL LEVEL 64 pixels

Refinement Module

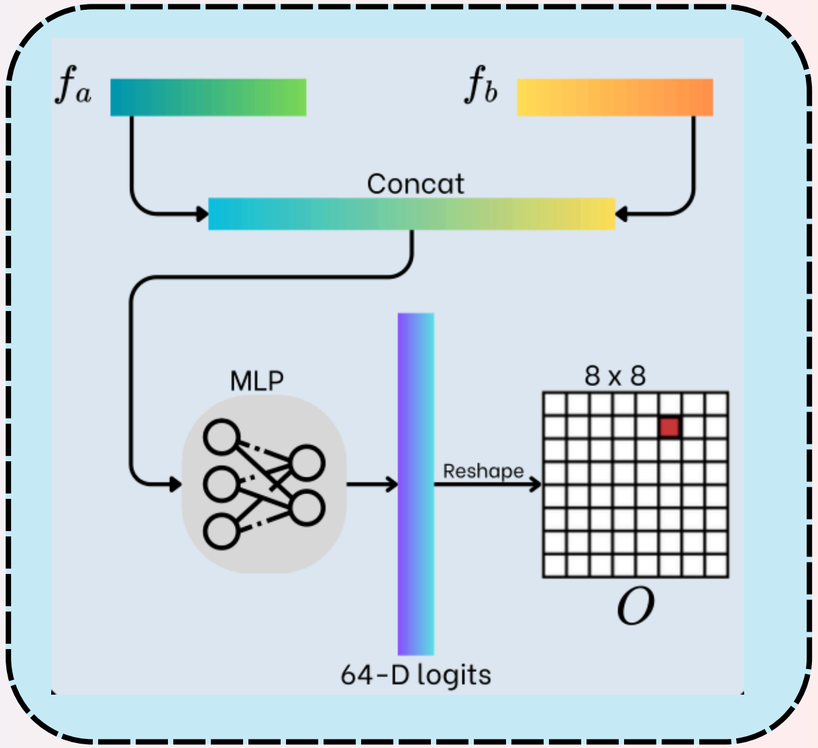
Purpose:

- Achieve pixel level precision on semi-dense image matching. This due to descriptors matching are at cell level (64 pixels)

Procedure:

1. **Initial Matching:** Use mutual nearest neighbor techniques to pair features and from image pair .
2. **Concat the descriptors** that are matched
3. **Offset Calculation:** Employ an MLP to predict pixel offsets so the right pixel to match.

.



Dual-Softmax and Reliability Losses

Xfeat is trained in a supervised manner with pixel-level ground truth correspondences.

1. **Descriptors -> Local descritors Loss** (L_{ds})
   - **Purpose:** Optimize local descriptor matching
   - **Key Steps:**
     - Compute similarity matrix $S = F_1 F_2^T$
     - Match features bidirectionally (forward + reverse)

$$L_{\text{rel}} = |\sigma(R_1) - (\overline{R}_1 \odot \overline{R}_2)| + |\sigma(R_2) - (\overline{R}_1 \odot \overline{R}_2)|$$

2. **Reliability ->Reliability Loss** (L_{rel})
   - **Purpose:** Train reliability map to quantify feature confidence
   - **Key Features:**
     - Used dual-softmax probabilities as supervision
     - Use of sigmoid activation function

$$L_{\text{ds}} = -\sum_i \left[\log(\text{softmax}_r(S))_{ii} + \log(\text{softmax}_r(S^T))_{ii}\right]$$

$$\bar{R}_1 = \max_r(\text{softmax}_r(S))$$

Keypoints Loss, Pixel Offset and Final Loss

3. **Refinement Module -> Pixel offsets Loss** (L_{fine})
   - **Supervision:**
     - Ground-truth correspondences $M_{I_1 \leftrightarrow I_2}$
   - **Objective:**
     - Refine matches to pixel-level precision using **NLL** loss

$$\mathcal{L}_{\text{fine}} = -\sum_i \log(\text{softmax}(\mathbf{o}_i))_{\bar{y}_i, \bar{x}_i}$$

4. **Keypoint-> Keypoints Loss** (L_{kp})
   - **Supervision:**
     - Knowledge distillation form ALIKE's tiny backbone
   - **Process:**
     - Map keypoints to linear indices:

$$tidx = t_x + t_y \cdot 8, t_{idx} \in \{0, 1, \ldots, 64\}$$

   - Use **NLL loss** for detection

$$\mathcal{L}_{\text{kp}} = -\sum_k \log(\text{softmax}(\mathbf{k}_{i,j}))_{t_{\text{idx}}}$$

Final Loss: $L = \alpha L_{\text{ds}} + \beta L_{\text{rel}} + \gamma L_{\text{fine}} + \delta L_{\text{kp}}$

Experiments: Relative Pose Estimation

Definition:

- Determines the spatial relationship between two camera poses by estimating the rotation and translation that align their views

Metrics:

- **AUC@{5°,10°,20°}:** Measure accuracy across angular threshold
- **Acc@10°:** Proportion of poses with angular error below 10 degrees
- **Mean Inlier Ratio (MIR):** Ratio of matching points that comply with the esimated model

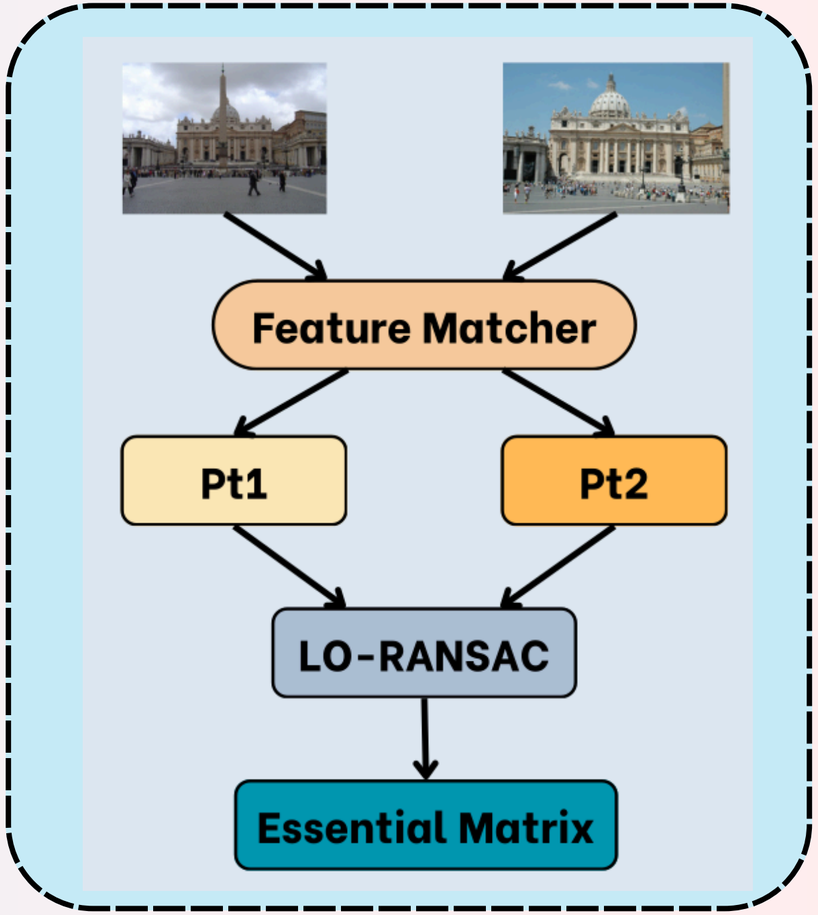


Table 4: Megadepth-1500 relative camera pose estimation

Method	AUC@5°	AUC@10°	AUC@20°	Acc@10°
Fast				
ORB	17.9	27.6	39.0	43.1
ORB (our)	15.7	24.5	34.3	37.7
ALIKE	49.4	61.8	71.4	77.7
ALIKE (our)	48.4	60.5	70.0	76.5
XFeat	42.6	56.4	67.7	74.9
XFeat (our)	41.6	55.5	67.3	74.3
XFeat*	50.2	65.4	77.1	85.1
XFeat* (our)	**50.9**	**66.7**	**78.5**	**86.8**

Table 1: Relative camera pose estimation results for the Megadepth-1500 dataset,

Robustness Indoors: ScanNet-1500 Pose Estimation

- **Task:** Relative camera pose estimation between image pairs.
- **Dataset:** ScanNet-1500, featuring challenging indoor scenes (different domain from primary training data like MegaDepth landmarks).
- **Evaluation:** Measured accuracy using Pose Error AUC @ 5°, 10°, 20°.

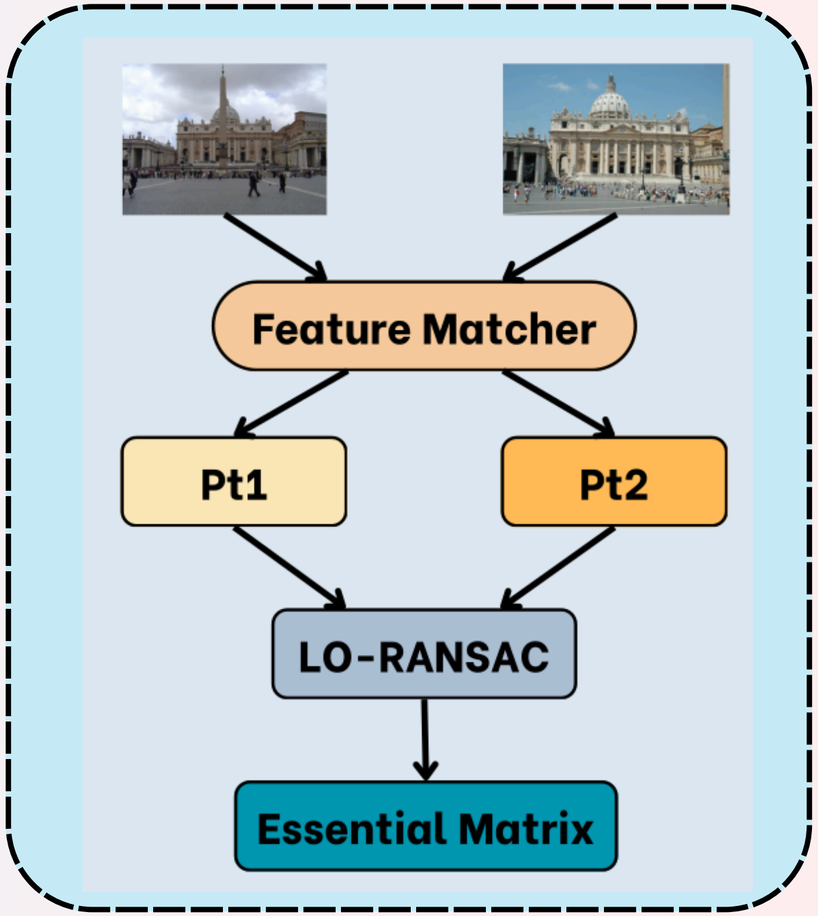


Table 6: ScanNet-1500 relative pose estimation: paper vs. our results (AUC).

AUC threshold	ORB(paper/ours)		ALIKE(paper/ours)		XFeat(paper/ours)		XFeat*(paper/ours)	
@5°	9.0	/5.704	8.0	/8.201	16.7	/16.910	18.4	/18.403
@10°	18.5	/12.484	16.4	/16.449	32.6	/32.941	34.7	/34.907
@20°	29.9	/20.265	25.9	/26.232	47.8	/48.583	50.3	/50.564

Model Training Strategies

- **Default:** The standard XFeat model architecture and training procedure as fully described in the paper (using mixed real+synthetic data and the separate keypoint head).
- **No synthetic data:** The standard XFeat model trained only on the real Megadepth dataset, excluding the synthetically warped COCO images.
- **Smaller model:** An XFeat variant with reduced channel capacity (halved channels in the last three blocks), making it computationally lighter.

Model Training Strategies Replication

Strategy	XFeat			XFeat*		
	AUC@5°	AUC@10°	AUC@20°	AUC@5°	AUC@10°	AUC@20°
Default (Paper)	42.6	—	—	50.2	—	—
Default (Ours)	41.6	55.5	67.3	50.9	66.7	78.5
(i) No synthetic data (Paper)	41.5	—	—	33.9	—	—
(i) No synthetic data (Ours)	**43.4**	**57.6**	**69.0**	**51.4**	**67.1**	**78.8**
(ii) Smaller model (Paper)	37.4	—	—	40.7	—	—
(ii) Smaller model (Ours)	31.1	44.3	56.9	—	—	—

Table 2: Comparison of AUC scores between Paper and Reimplementation Results across Variants

Strategy	XFeat			XFeat*		
	mAcc@5°	mAcc@10°	mAcc@20°	mAcc@5°	mAcc@10°	mAcc@20°
Default (Ours)	62.8	74.3	81.7	75.7	**86.8**	92.5
(i) No synthetic data (Ours)	**65.1**	**76.9**	**83.5**	**76.1**	86.7	**92.9**
(ii) Smaller model (Ours)	50.3	63.5	73.7	—	—	—

Table 3: Comparison of mean Accuracy (mAcc) between Paper and Reimplementation Results across Variants

Model Architecture Experiments

Idea 1 - Adding Attention

- **Goal:** Enhance feature representation by focusing on more relevant spatial and channel information.
- **Changes vs. Original:**
  - **NEW Component:** Introduced the CBAM (Convolutional Block Attention Module) class.
    - **Channel Attention:** Learns to weight the importance of different feature channels.
    - **Spatial Attention:** Learns to weight the importance of different spatial locations.
  - **Integration:** The CBAM module (self.cbam) is applied directly after the pyramid feature fusion (self.block_fusion) and before the features (feats) are passed to the heatmap_head.
  - **No other changes:** The BasicLayer, backbone structure, other heads, and fine_matcher remain identical to the original.

Idea 2 - Improving Regularization & Normalization

- **Goal:** Enhance model robustness and potentially improve generalization by adding dropout and using a different normalization technique (GroupNorm).
- **Changes vs. Original:**
  Modified BasicLayer:
  - **NEW:** Added nn.Dropout2d (spatial dropout)
  - **NEW:** Added selectable normalization (norm_type).
- **Backbone & Head Modifications:**
  - Most BatchNorm2d layers within the block1 to block5 and the heatmap_head / keypoint_head have been replaced with GroupNorm (via the modified BasicLayer).
  - Spatial Dropout2d (with p=0.3) has been selectively added to several BasicLayer instances within the backbone and heads.
- **Input Normalization Change:** self.norm changed from nn.InstanceNorm2d(1) to nn.BatchNorm2d(1). (Note: While GroupNorm was added elsewhere, BatchNorm is used here).
- **fine_matcher:** Remains unchanged (still uses BatchNorm1d).

Idea 3 - Combining Attention and Regularization/Normalization

- **Goal:** Leverage potential benefits from both attention mechanisms and improved regularization/normalization techniques simultaneously.
- **Changes vs. Original:** This version combines the modifications from both Idea 1 and Idea 2.
- **Includes all changes from Idea 2:**
  - Modified BasicLayer with optional Dropout2d and selectable GroupNorm.
  - Systematic use of GroupNorm throughout most convolutional blocks.
  - Selective addition of Dropout2d in various layers.
  - Input normalization uses nn.BatchNorm2d(1).
- **Includes the change from Idea 1:**
  - Adds the CBAM module (self.cbam).
  - Applies CBAM to the fused features (feats) after block_fusion and before the heads, exactly like in Idea 1.

Results

Model	Setup	auc@5	auc@10	auc@20	mAcc@5	mAcc@10	mAcc@20
Xfeat	Default	**41.6**	**55.5**	**67.3**	62.8	74.3	**81.7**
	CBAM	**41.6**	55.4	66.7	**62.9**	**74.0**	80.5
	Reg+Norm	37.9	52.1	64.3	59.1	71.3	79.7
	CBAM+Reg+Norm	38.3	52.2	63.7	59.5	70.2	78.3
Xfeat*	Default	**50.9**	**66.7**	**78.5**	**75.7**	86.8	**92.5**
	CBAM	49.9	66.0	78.3	74.9	**86.9**	92.4
	Reg+Norm	37.4	53.3	67.3	59.6	75.8	85.3
	CBAM+Reg+Norm	35.6	51.6	66.1	56.8	74.9	84.9

Table 1: Architecture Ablation of Xfeat & Xfeat* on Megadepth-1500 test set

Out of Domain Testing

Frame Generation

- **Core Idea:** Adapt an existing motion-feature-based interpolation framework. Use XFeat to detect, describe, and match keypoints between consecutive frames to derive these motion features. Warp pixels based on the estimated motion from these matched XFeat features to synthesize the intermediate frame.
- **Methodology:**
  - **Motion Estimation via XFeat:** Leveraged the efficient and robust XFeat model for inter-frame motion analysis.
  - **Feature Extraction:** Applied **XFeat.detectAndCompute** on consecutive frames (N and N+1) to get keypoints and 64D descriptors.
  - **Feature Matching:** Used XFeat.match (cosine similarity + mutual nearest neighbor) to find reliable correspondences between frames.
  - **Motion Feature Derivation:** Converted matched XFeat keypoint pairs into MotionFeature objects, representing pixel motion vectors between frames.
  - ***Quality Metric:*** Used descriptor distance as a proxy for match confidence.
  - **Interpolated Frame Synthesis:**
    - Generated the intermediate frame (N+0.5) pixel by pixel.
  - For each target pixel, calculated a weighted average of colors sampled from Frame N and Frame N+1.
  - Sampling locations were determined by warping the target pixel using nearby MotionFeature vectors.
  - Weights incorporated feature quality and inverse distance.

Frame Generation

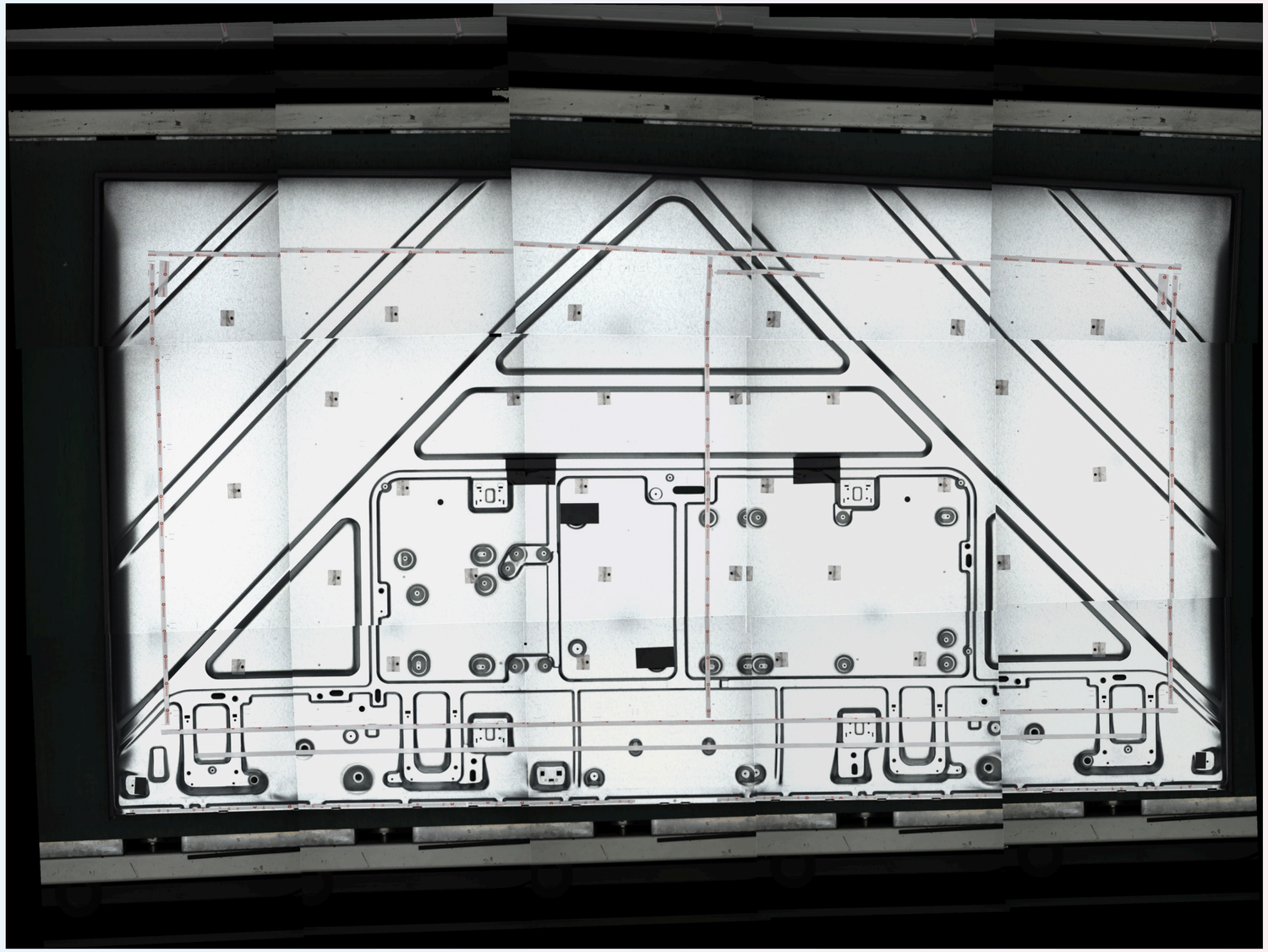
Multi-Image Stitching

Goal: Seamlessly combine multiple overlapping images into a single panorama.

Key Steps:

- **Image Preprocessing:** Enhance input images (Sharpening, Contrast).
- **Feature Detection & Matching:** Use **XFeat** to find robust feature matches in overlapping regions.
- **Transformation Estimation:** Calculate Partial Affine transformations for each image relative to a base image (using a layered, multi-path approach for robustness).
- **Gain Compensation:** Adjust color/brightness differences between images based on overlap analysis.
- **Pyramid Blending:** Warp images onto a common canvas and blend overlap areas using Laplacian Pyramids for smooth transitions.

Multi-Image Stitching



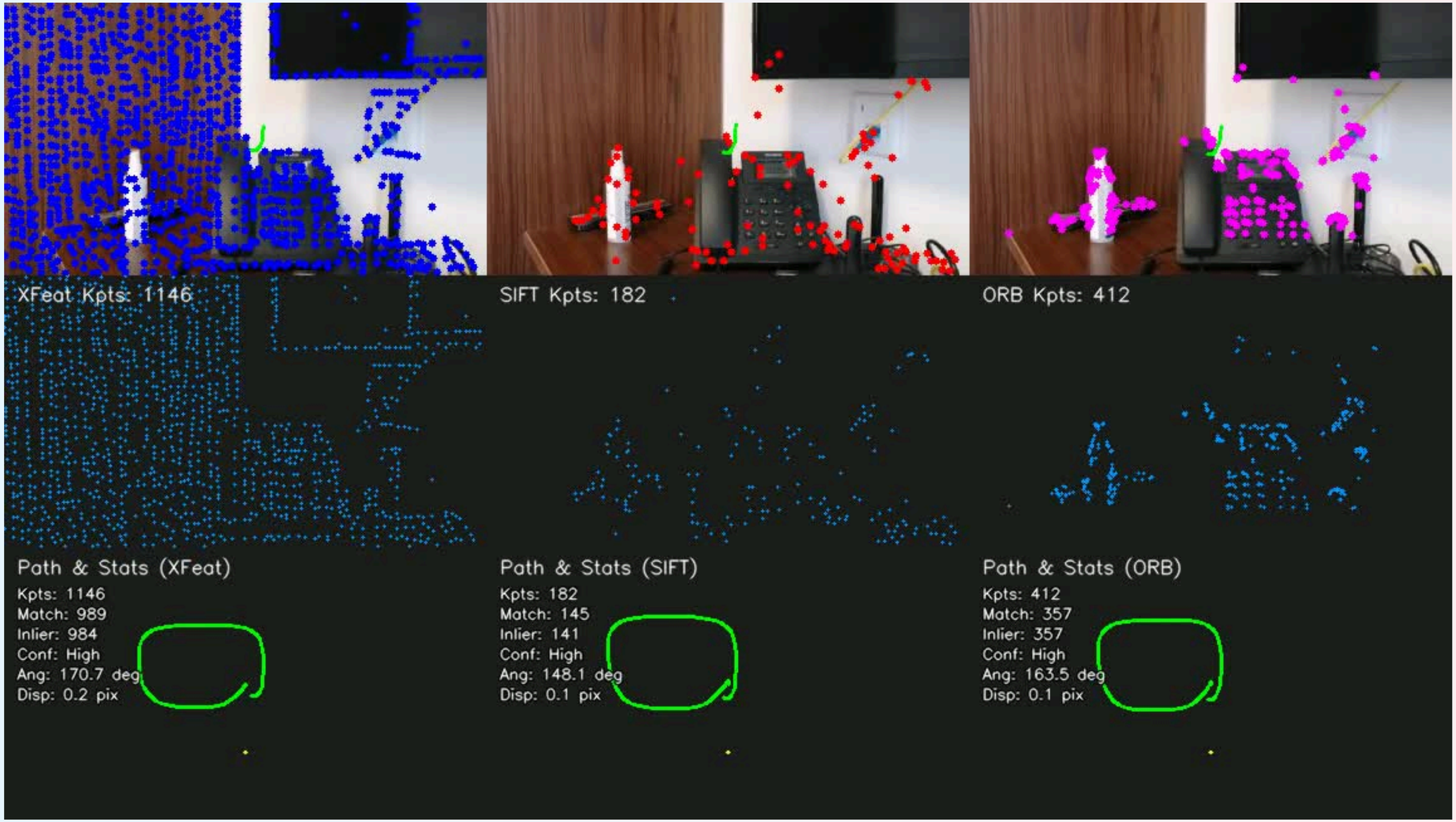
Visual Odometry Tracking

Goal: Analyze and visualize frame-to-frame camera motion using different feature methods.

Key Steps:

- **Input:** Reads video frames or webcam feed.
- **Feature Extraction:** Finds keypoints and descriptors in each frame using XFeat, SIFT, and ORB.
- **Feature Matching:** Matches features between the previous and current frames for each method.
- **Motion Estimation:** Calculates camera translation and rotation using robust estimation (RANSAC) on matched features.
- **Position Tracking:** Cumulatively updates an estimated position path based on the calculated motion per frame.
- **Visualization:** Displays live views showing detected features, inlier matches, estimated motion vectors, and the tracked path side-by-side for XFeat, SIFT, and ORB.

Visual Odometry Tracking



Post Training Improvemnets

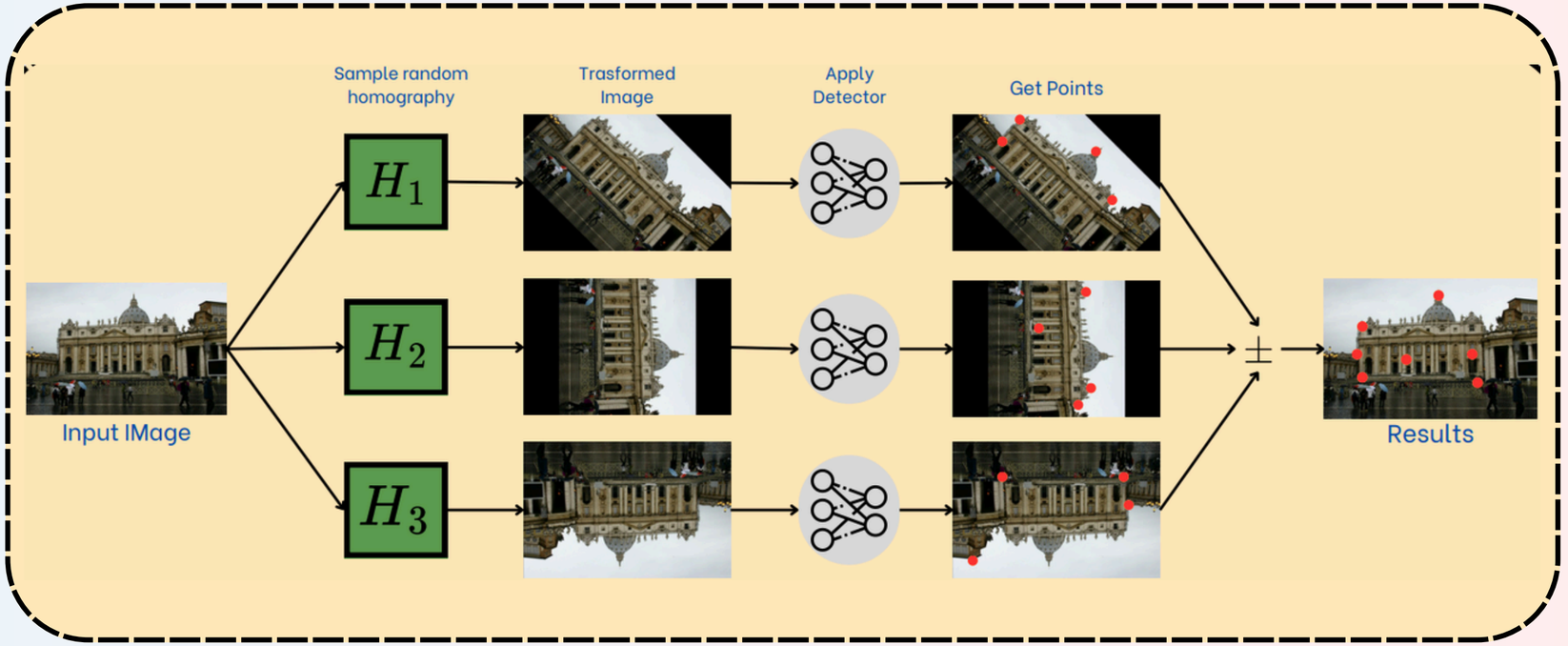
XFeat transformed

Idea:

- Search to improve the selection of features with somo homography trasformation

Procedure:

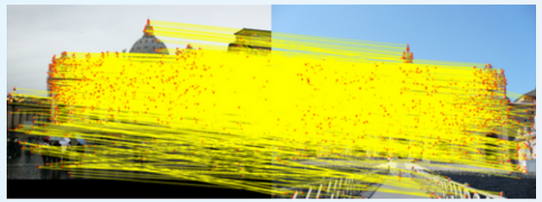
- Sample random homography
- Apply trasnformation
- Apply detector
- Unify/Intersect points



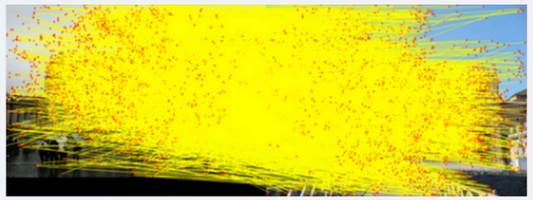
XFeat transformed - Results

Qualitative:

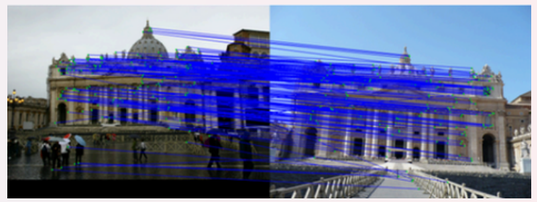
Original



Xfeat Original
1293 points



XFeat Trasformed unified
3810 points (x3 original)



XFeat Trasformed intersected
181 points (÷7 original)

Quantitative:

Method	N trasnformation	AUC@5°	AUC@10°	AUC@20°	ACC@10°	FPS
XFeat original	0	**40.4**	54.4	66.2	73.5	**6.1±0.2**
XFeat trasformed unify	1	33.5	46.4	57.5	63.9	2.3±0.2
XFeat trasformed unify	2	25.0	37.0	49.2	55.1	1.4±0.2
XFeat trasformed unify	3	23.1	34.4	46.6	51.9	1.0 ±0.2
XFeat trasformed intersect	1	**40.4**	**54.8**	**66.5**	**73.9**	3.1±0.2
XFeat trasformed intersect	2	33.8	49.0	62.6	70.5	2.1±0.2
XFeat trasformed intersect	3	29.3	43.9	58.0	66.1	1.7 ±0.2

Relative camera pose estimation results for the Megadepth-1500 dataset

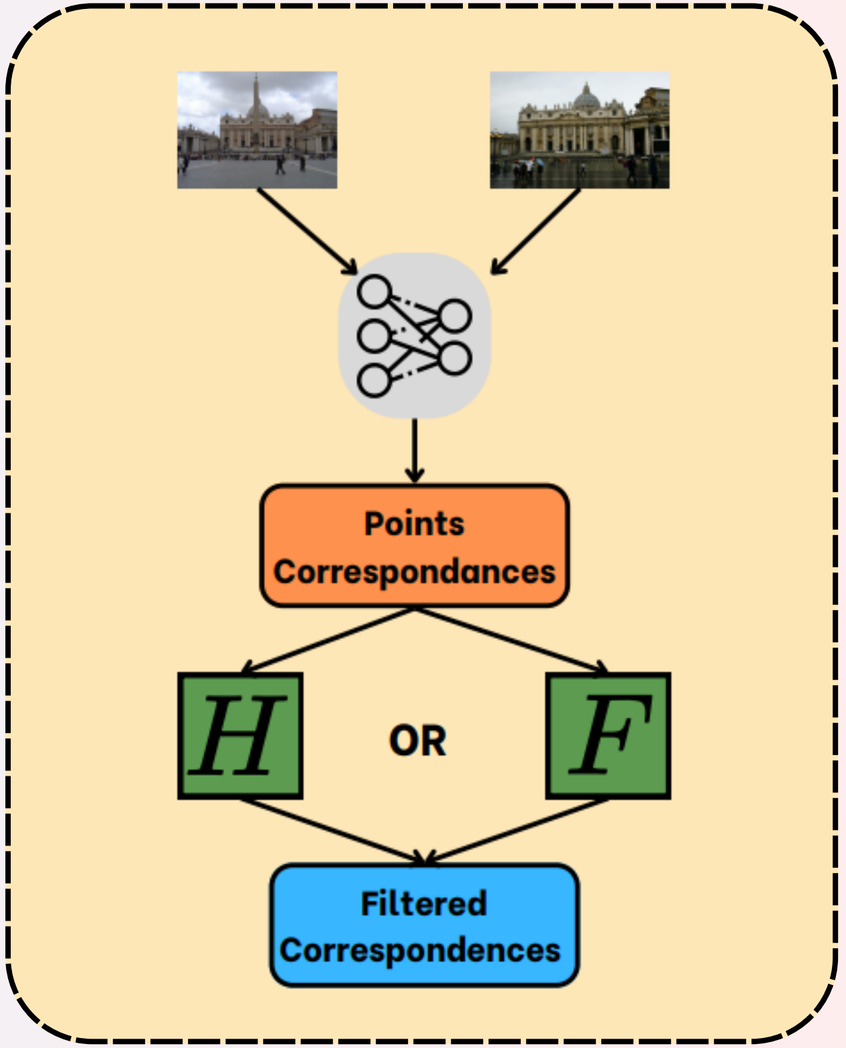
XFeat Refined

Idea:

- Refine the points correspondences using Homography or Fundamental Matrix

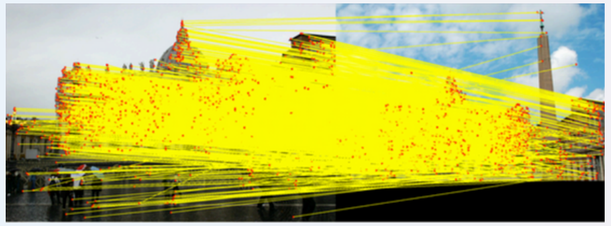
Procedure:

1. **Extract Point Correspondences:** Use XFeat to match features between two images.
2. **Compute Transformation Matrix:** Estimate the Homography or Fundamental Matrix.
3. **Filter Correspondences:** Refine the point correspondences by keeping only the inliers that are consistent with the estimated matrix.

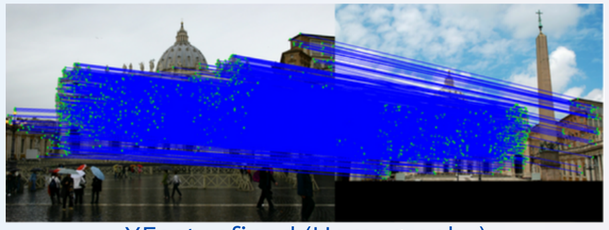


XFeat Refined - Results

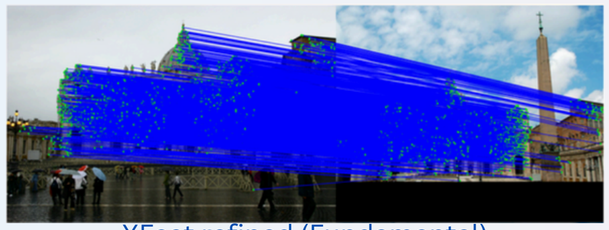
Original



Xfeat Original
1566 points



XFeat refined (Homography)
1193 points



XFeat refined (Fundamental)
1317 points

Quantitative

Method	AUC@5°	AUC@10°	AUC@20°	Acc@10°	FPS
XFeat original	40.4	54.4	66.2	73.5	**6.1±0.2**
XFeat refined (H)	**41.8**	**56.2**	**68.0**	**75.7**	6.0 ±0.2
XFeat refined (F)	40.5	54.7	66.8	74.6	6.0 ±0.2

Table 6: Relative camera pose estimation results for the Megadepth-1500 dataset,

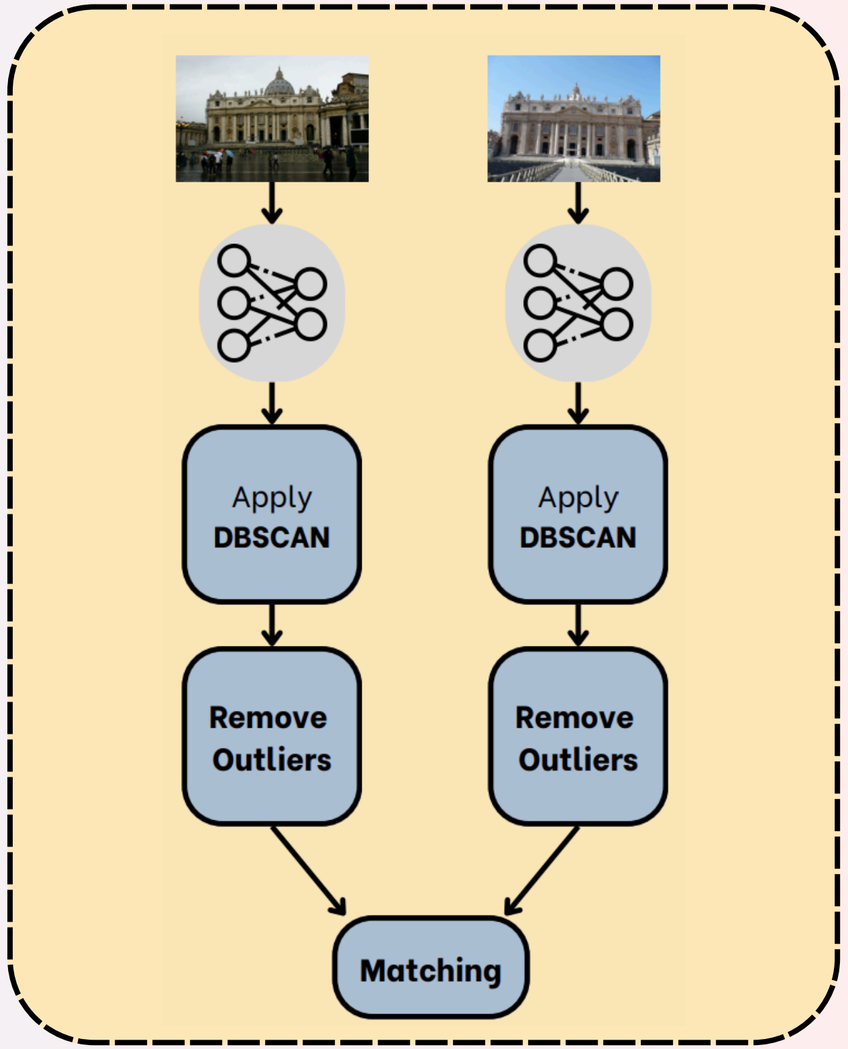
XFeat* Clustering

Idea:

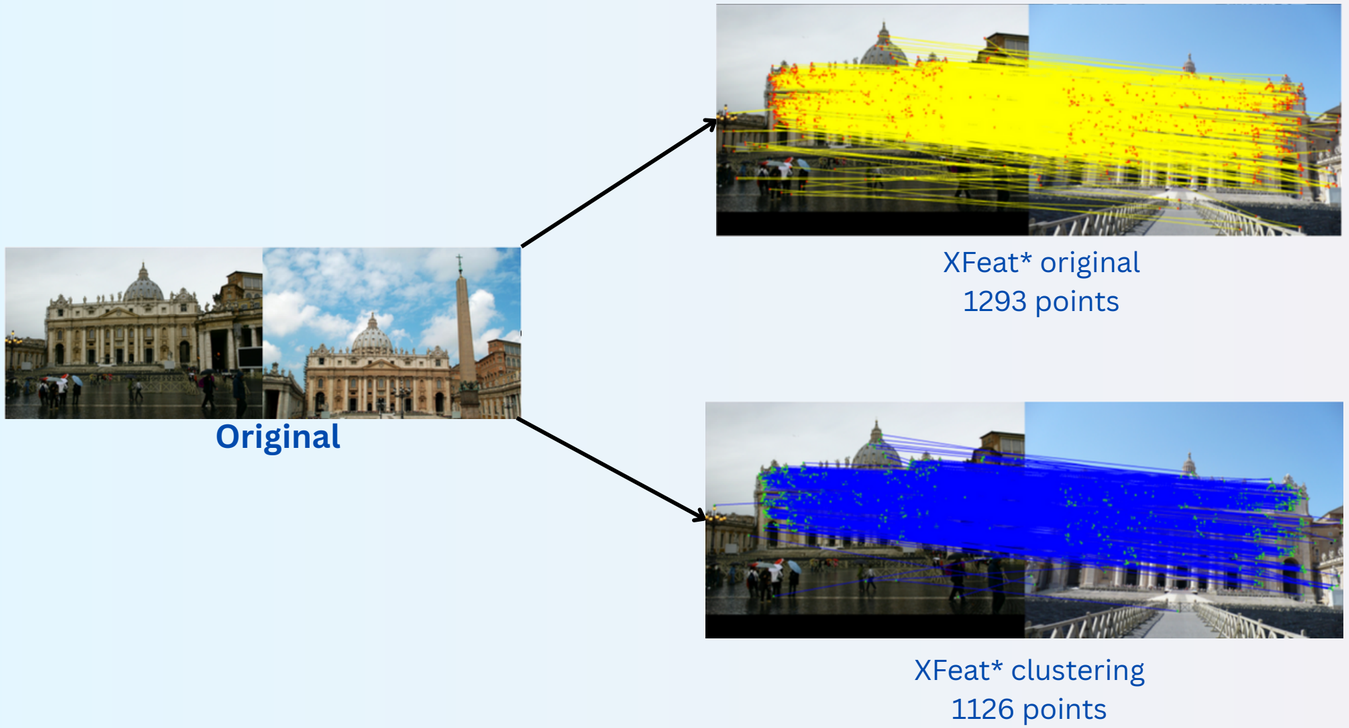
- Improve the semi-dense matching by removing outliers

Procedure:

1. **Extract** the top 10,000 points
2. Use **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise) to group points into meaningful clusters based on spatial proximity
3. **Identify and remove outliers** (points that do not belong to any cluster).
4. Perform **semi-dense feature matching**



XFeat* Clustering - Results



Quantitative

Method	AUC@5°	AUC@10°	AUC@20°	ACC@10°	FPS
XFeat* original	50.2	65.4	77.1	85.1	**5.7±0.2**
XFeat* clustering	**50.7**	**66.0**	**77.6**	**85.9**	3.8 ±0.2

Table 7: Relative camera pose estimation results for the Megadepth-1500 dataset,

Conclusion

- **Innovation in XFeat:**
  - Lightweight CNN for efficient **keypoint detection** and local **feature extraction.**
  - Dual modes: **Sparse Matching (XFeat)** for speed and **Semi-Dense Matching (XFeat)*** for precision.

- **Our Improvements:**
  - **Homography Transformations:** Search to increase the quality of keypoints.
  - **Point Refinement:** Applied RANSAC-based filtering for robust feature matching**.**
  - **Clustering with DBSCAN:** Effectively removed outliers, boosting semi-dense match reliability.

Other Interesting Ideas For Future

- **Domain-Specific Adaptation:** Fine-tune XFeat on specialized datasets (e.g., aerial, medical, underwater imagery) to optimize performance for specific challenging conditions.
- **Architectural Exploration:** Explore different backbone design choices, like how the number of channels increases across layers and the types of layers used, as well as changes to the decoupled keypoint head to achieve better performance.
- **Dense Feature Applications:** Leverage the efficiently computed dense/semi-dense XFeat features for tasks like dense 3D reconstruction.

Challenges Faced

Reproducibility & Details:

- Code for Majority of the Baseline experiments/ablations (e.g., variations) was not available in the repository, even for Xfeat for many dataset, it wasn’t provided.
- Limited architectural details provided for ablated models like the "Smaller model" (table 5).

Data Handling:

- The primary training dataset (MegaDepth) is very large (~150GB+), posing data management challenges as it was out first time handle data at this scale.

References

- [1] Hongkai Chen, Zixin Luo, Lei Zhou, Yurun Tian, Mingmin Zhen, Tian Fang, David Mckinnon, Yanghai Tsin, and Long Quan. Aspanformer: Detector-free image matching with adaptive span transformer. In ECCV, pages 20–36. Springer, 2022.
- [2] Johan Edstedt, Ioannis Athanasiadis, Marten Wadenb ˚ack, ¨ and Michael Felsberg. Dkm: Dense kernelized feature matching for geometry estimation. In CVPR, pages 17765– 17775, 2023
- [3] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In CVPR, pages 8922–8931, 2021.
- [4] Prune Truong, Martin Danelljan, Radu Timofte, and Luc Van Gool. Pdc-net+: Enhanced probabilistic dense correspondence network. IEEE TPAMI, 2023.
- [5] Li, Zhengqi, and Noah Snavely. "Megadepth: Learning single-view depth prediction from internet photos." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
- [6] Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014,

Thank You