# 6. COTR (Image Matching) 박희찬

**Title Paper**

## COTR: Correspondence Transformer for Matching Across Images

University of British Columbia, Kwang Moo Yi (2021, Aug) ICCV2021

https://arxiv.org/abs/2103.14167

---

**Ref**

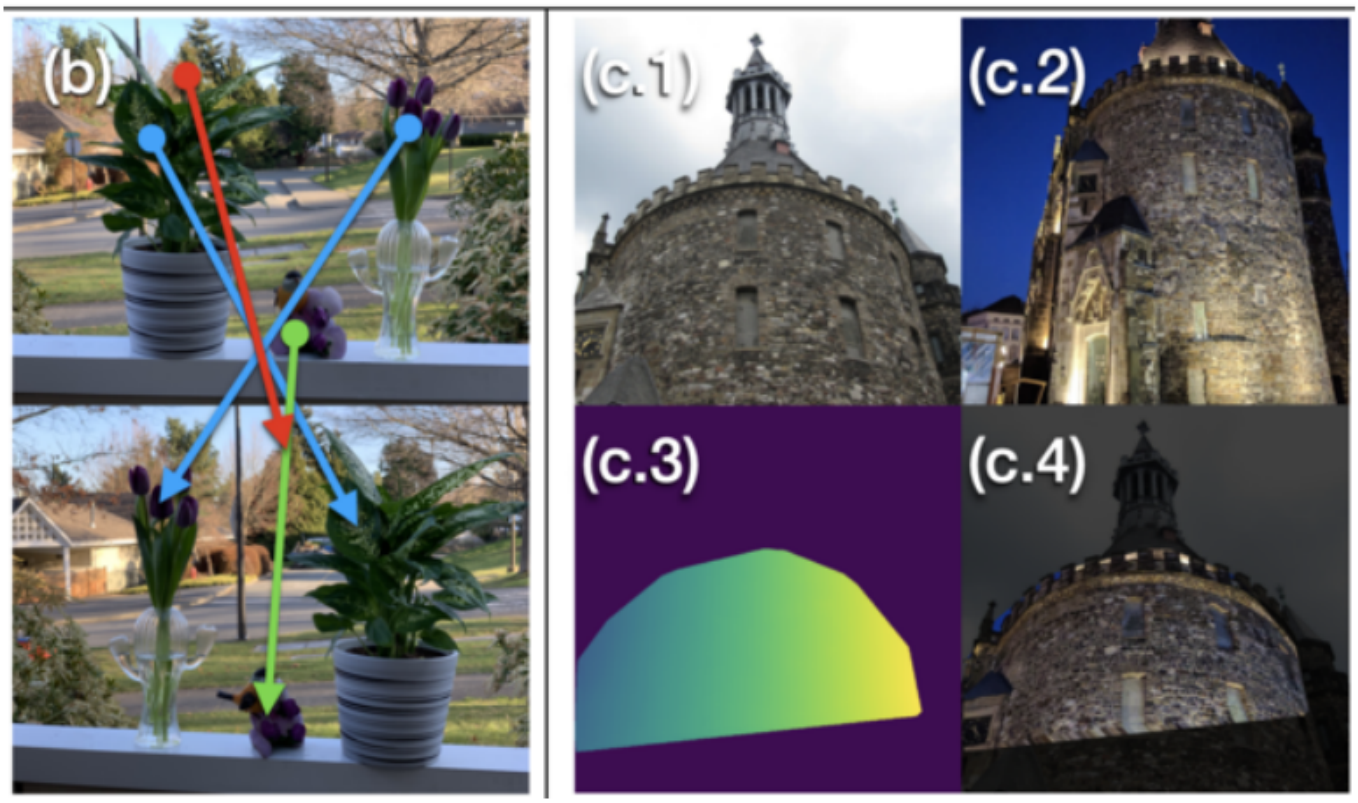
- DGC-Net (https://arxiv.org/abs/1810.08393 )
- GLU-Net (https://arxiv.org/abs/1912.05524 )
- GOCor (https://arxiv.org/abs/2009.07823 )

---

***감상 후기**

- 읽으면서 느낀건데.. Image Matching Challenge (IMC2020) 참가 후 쓴 후기형 논문의 느낌이 강하다.
- Transformer 모델에서 MLP로 대체하는 실험을 하고, MLP로 대체할 방법을 찾아보겠다고 하는걸 보니, 학습 속도나 학습 시간에 대한 정보를 공개하지 않는 이유를 알 것 같다.
- 입력 쿼리 포인트는 Random하게 정해지기 때문에, 산불 이미지상에서는 연기를 매칭 하는게 아니라 지형을 매칭하게 될 것 같다.
  COTR 모델이 아니라 하더라도, Sparse Method 정도는 좋은 참고가 될 것 같기도... → 안될 듯

.....



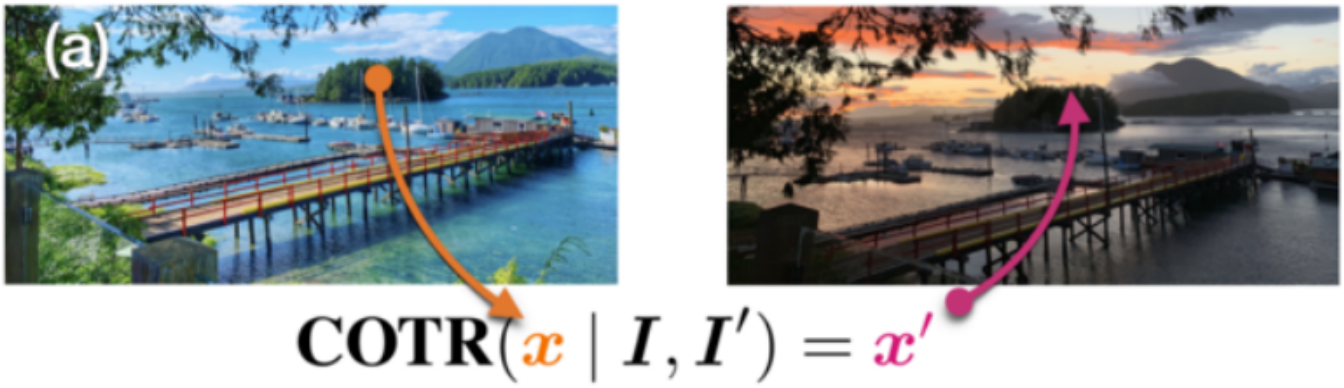

## 1. Image Matching (Image Correspondence)

- Sparse Method
  - 두 이미지 간 Keypoints 추출
  - Single 카메라 (geometrical constrains) 상에서의 이미지를 사용
  - Local features.
  - 두 이미지간의 robust matcher가 필요
    - 기존 방법론들은 matcher stage가 따로 있지만, COTR에서는 없다.
- Dense Method
  - 첫번째 이미지의 pixel 들을 두번째 이미지에 mapping.
  - small temporal changes (ex. optical flow)이나 large displacements에 집중하여 모델링

- local smoothness.
- texture가 없는 영역까지 포함하여 전체적으로 이미지를 매칭 (global)

→ Sparse: Local problem, Dense: Global problem

- Sparse + Dense?
  - DGC-Net (https://arxiv.org/abs/1810.08393 )
  - GLU-Net (https://arxiv.org/abs/1912.05524 )
  - GOCor (https://arxiv.org/abs/2009.07823 )




## 2. COTR

**(1) Problem formulation**

$$\boldsymbol{x} \in [0,1]^2$$

$$\boldsymbol{x}' \in [0,1]^2$$

$$\mathcal{F}_{\boldsymbol{\Phi}}\left(\boldsymbol{x} | \boldsymbol{I}, \boldsymbol{I}'\right)$$

$$\arg\min_{\boldsymbol{\Phi}} \mathop{\mathbb{E}}_{(\boldsymbol{x},\boldsymbol{x}',\boldsymbol{I},\boldsymbol{I}')\sim\mathcal{D}} \mathcal{L}_{\text{corr}} + \mathcal{L}_{\text{cycle}}, \tag{1}$$

$$\mathcal{L}_{\text{corr}} = \left\|\boldsymbol{x}' - \mathcal{F}_{\boldsymbol{\Phi}}\left(\boldsymbol{x} \mid \boldsymbol{I}, \boldsymbol{I}'\right)\right\|_2^2, \tag{2}$$

$$\mathcal{L}_{\text{cycle}} = \left\|\boldsymbol{x} - \mathcal{F}_{\boldsymbol{\Phi}}\left(\mathcal{F}_{\boldsymbol{\Phi}}\left(\boldsymbol{x} \mid \boldsymbol{I}, \boldsymbol{I}'\right) \mid \boldsymbol{I}, \boldsymbol{I}'\right)\right\|_2^2, \tag{3}$$

loss_corr : correspondence estimation errors.

loss_cycle : correspondences to be cycle-consistent.

Φ : best set of parameters for parametric function **F**.

x : Normalized Coordinates (query point) → 이미지 내에서 무작위로 선정

**(2) Network Architecture**

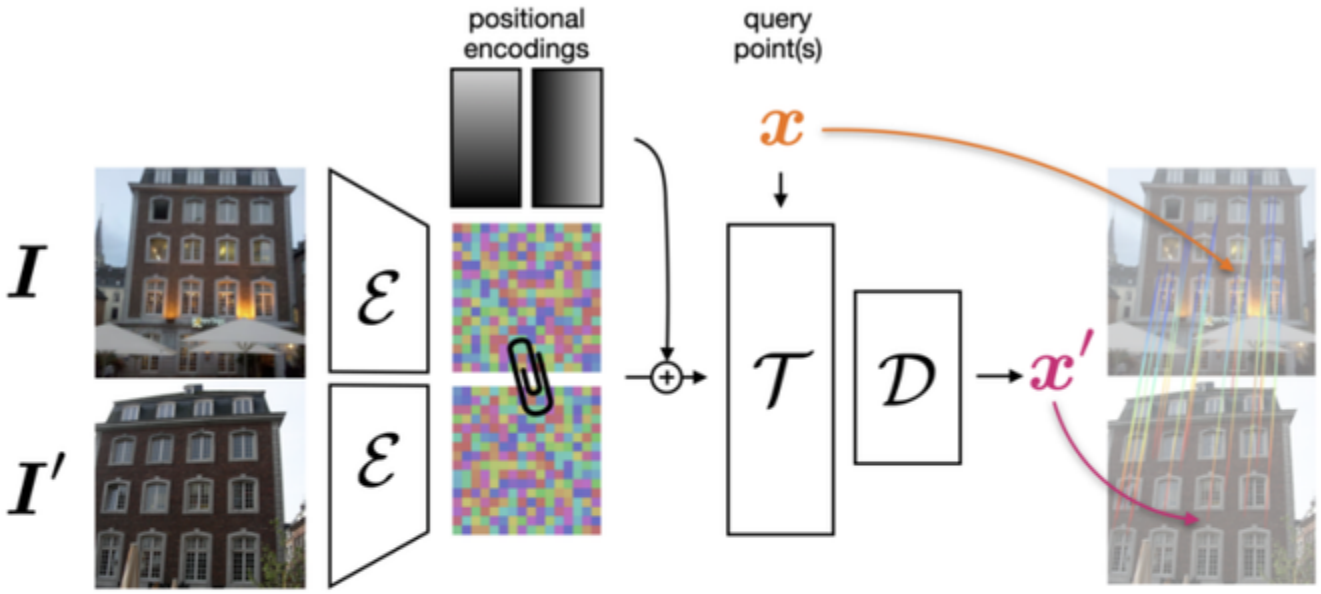


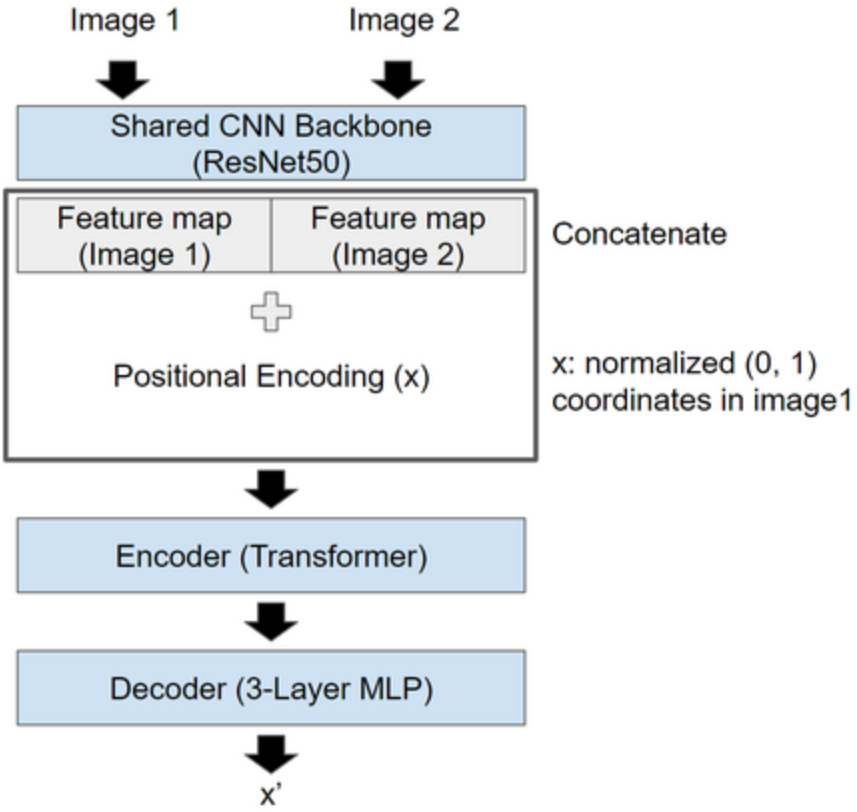

Figure 2. **The COTR architecture –** We first process each image with a (shared) backbone CNN $\mathcal{E}$ to produce feature maps size 16x16, which we then concatenate together, and add positional encodings to form our context feature map. The results are fed into a transformer $\mathcal{T}$, along with the query point(s) $\boldsymbol{x}$. The output of the transformer is decoded by a multi-layer perceptron $\mathcal{D}$ into correspondence(s) $\boldsymbol{x}'$.

$$\boldsymbol{x}' = \mathcal{F}_{\boldsymbol{\Phi}}\left(\boldsymbol{x}|\boldsymbol{I},\boldsymbol{I}'\right) = \mathcal{D}\left(\mathcal{T}_{\mathcal{D}}\left(\mathcal{P}\left(\boldsymbol{x}\right),\mathcal{T}_{\mathcal{E}}\left(\mathbf{c}\right)\right)\right). \quad (5)$$

(1) Shared CNN backbone를 이용해 16x16x256 feature map을 얻음

(2) 두 feature map을 Concatenate (16x32x256) 하고, 첫번째 이미지 query point (x)에 대한 positional encoding을 더한다. → **c**

(3) **c** 를 Transformer 의 Encoder에 넣고, Fully Connected layer로 이루어진 Decoder로 두번째 이미지 query point (x')를 얻는다.




- **Concatenation with transformer.**
  - 두 이미지를 하나인 것처럼 만들고 Self Attention 하여 두 이미지상의 동일한 객체에 대해 locations 과 relation 을 학습하도록 유도
  - 16x16x256 2개를 16x32x256 으로 channel dimension이 아니라 spatial dimension 을 합침.
- **Positional Encoding**

$$\mathcal{P}(\boldsymbol{x}) = \left[p_1(\boldsymbol{x}), p_2(\boldsymbol{x}), \ldots, p_{\frac{N}{4}}(\boldsymbol{x})\right], \quad (6)$$

$$p_k(\boldsymbol{x}) = \left[\sin(k\pi\boldsymbol{x}^\top), \cos(k\pi\boldsymbol{x}^\top)\right], \quad (7)$$

  - N=256 channel 수, pk 마다 4개의 값을 내기 때문에 N/4 (x는 coordinates)
- **Querying multiple points**
  - set of query point (x)를 이용하여 self-attention 할 때, 서로간에 독립적으로 매칭을 하기 위해서 다른 point 들은 masking 처리.

**(3) Inference**

- **Recursive with Zoom-in**


Figure 3. **Recursive COTR at inference time –** We obtain accurate correspondences by applying our functional approach recursively, zooming into the results of the previous iteration, and running the *same* network on the pair of zoomed-in crops. We gradually focus on the correct correspondence, with greater accuracy.

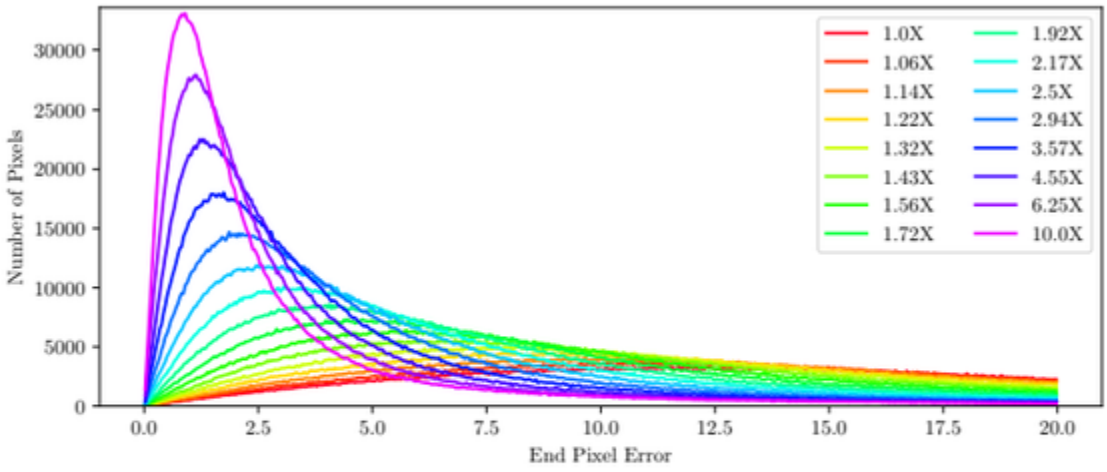



Figure 10. **Zooming –** We plot the distribution of the end pixel error (EPE) at different zoom-in levels, on the HPatches dataset. The error clearly decreases as more zoom is applied.

- **Scale Mismatch**

zoom-in으로 인하여 두 이미지간에 scale이 달라질 가능성이 존재.

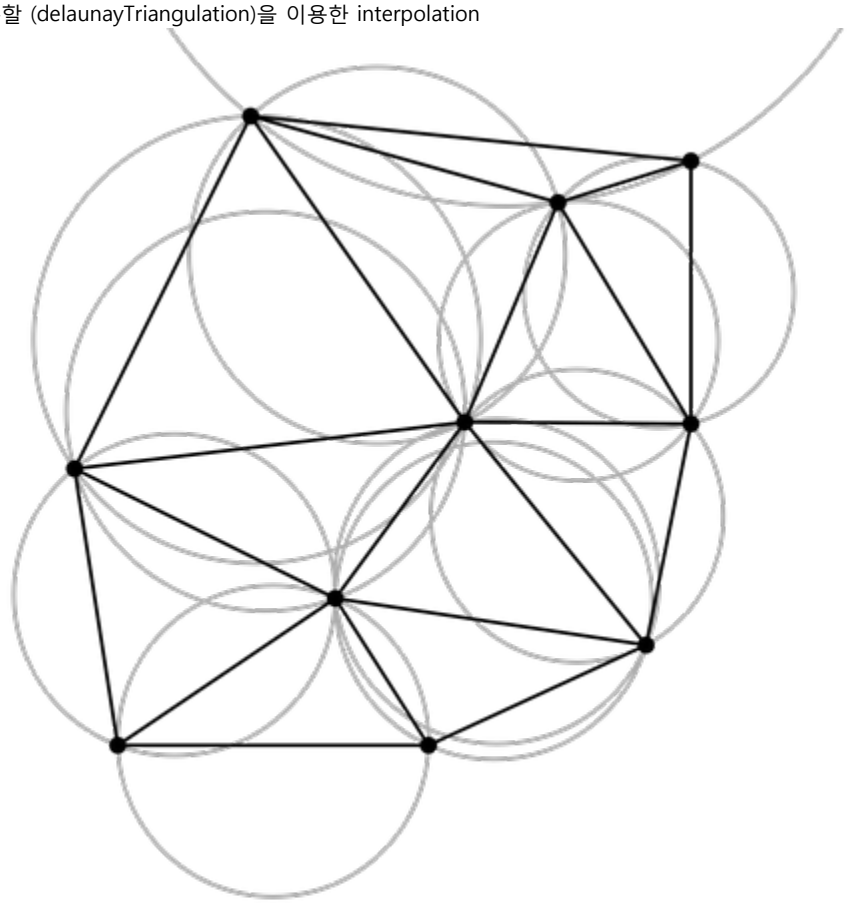
recursive iteration 을 진행하면서, Cycle Consistency 에러가 큰 영역을 masking 처리 해주면서 Zoom-in에 대한 보상 (Compensating)을 진행

- **Erroneous Correspondences**

Figure 4. **Estimating scale by finding co-visible regions** – We show two images we wish to put in correspondence, and the estimated regions in common – image locations with a high cycle-consistency error are masked out.

- **Dense Estimation을 하고 싶다면?**

query point에 들로네 삼각분할 (delaunayTriangulation)을 이용한 interpolation



- **Transformer 를 MLP로 대체한다면?**


## 2. Dataset

(1) HPatches

*AEPE: Average End Point Error

*PCK: Percentage of Correct Keypoints

| Method | AEPE ↓ | PCK-1px ↑ | PCK-3px ↑ | PCK-5px ↑ |
|---|---|---|---|---|
| LiteFlowNet [25] CVPR'18 | 118.85 | 13.91 | – | 31.64 |
| PWC-Net [61, 62] CVPR'18, TPAMI'19 | 96.14 | 13.14 | – | 37.14 |
| DGC-Net [38] WACV'19 | 33.26 | 12.00 | – | 58.06 |
| GLU-Net [69] CVPR'20 | 25.05 | 39.55 | 71.52 | 78.54 |
| GLU-Net+GOCor [70] NeurIPS'20 | 20.16 | **41.55** | – | 81.43 |
| **COTR** | **7.75** | <u>40.91</u> | **82.37** | **91.10** |
| **COTR** +Interp. | <u>7.98</u> | 33.08 | <u>77.09</u> | <u>86.33</u> |

Table 1. **Quantitative results on HPatches –** We report Average End Point Error (AEPE) and Percent of Correct Keypoints (PCK) with different thresholds. For PCK-1px and PCK-5px, we use the numbers reported in literature. We **bold** the best method and <u>underline</u> the second best.

(2) KITTI

| Method | KITTI-2012 | | KITTI-2015 | |
|---|---|---|---|---|
| | AEPE↓ | Fl.[%]↓ | AEPE↓ | Fl.[%]↓ |
| LiteFlowNet [25] CVPR'18 | 4.00 | 17.47 | 10.39 | 28.50 |
| PWC-Net [61, 62] CVPR'18, TPAMI'19 | 4.14 | 20.28 | 10.35 | 33.67 |
| DGC-Net [38] WACV'19 | 8.50 | 32.28 | 14.97 | 50.98 |
| GLU-Net [69] CVPR'20 | 3.34 | 18.93 | 9.79 | 37.52 |
| RAFT [65] ECCV'20 | <u>2.15</u> | <u>9.30</u> | <u>5.04</u> | 17.8 |
| GLU-Net+GOCor [70] NeurIPS'20 | 2.68 | 15.43 | 6.68 | 27.57 |
| **COTR**[3] | **1.28** | **7.36** | **2.62** | **9.92** |
| **COTR** +Interp.[3] | 2.26 | 10.50 | 6.12 | <u>16.90</u> |

Table 2. **Quantitative results on KITTI –** We report the Average End Point Error (AEPE) and the flow outlier ratio ('Fl') on the 2012 and 2015 versions of the KITTI dataset. Our method outperforms most baselines, with the interpolated version being on par with RAFT, and slightly edging out GLU-Net+GOCor.


Figure 5. **Qualitative examples on KITTI –** We show the optical flow and its corresponding error map ("jet" color scheme) for three examples from KITTI-2015, with GLU-Net [69] as a baseline. COTR successfully recovers both the global motion in the scene, and the movement of individual objects, even when nearby cars move in opposite directions (top) or partially occlude each other (bottom).

(3) ETH3D

| Method | AEPE ↓ | | | | | | |
|---|---|---|---|---|---|---|---|
| | rate=3 | rate=5 | rate=7 | rate=9 | rate=11 | rate=13 | rate=15 |
| LiteFlowNet [25] CVPR'18 | **1.66** | 2.58 | 6.05 | 12.95 | 29.67 | 52.41 | 74.96 |
| PWC-Net [61, 62] CVPR'18, TPAMI'19 | 1.75 | 2.10 | 3.21 | 5.59 | 14.35 | 27.49 | 43.41 |
| DGC-Net [38] WACV'19 | 2.49 | 3.28 | 4.18 | 5.35 | 6.78 | 9.02 | 12.23 |
| GLU-Net [69] CVPR'20 | 1.98 | 2.54 | 3.49 | 4.24 | 5.61 | 7.55 | 10.78 |
| RAFT [65] ECCV'20 | 1.92 | 2.12 | 2.33 | 2.58 | 3.90 | 8.63 | 13.74 |
| **COTR** | **1.66** | **1.82** | **1.97** | **2.13** | **2.27** | **2.41** | **2.61** |
| **COTR** +Interp. | 1.71 | 1.92 | 2.16 | 2.47 | 2.85 | 3.23 | 3.76 |

Table 3. **Quantitative results for ETH3D –** We report the Average End Point Error (AEPE) at different sampling "rates" (frame intervals). Our method performs significantly better as the rate increases and the problem becomes more difficult.


Figure 6. **Qualitative examples on ETH3D –** We show results for GLU-Net [69] and COTR for two examples, one indoors and one outdoors. Correspondences are drawn in **green** if their reprojection error is below 10 pixels, and **red** otherwise.

(4) Image Matching Challenge (IMC2020) : photo-tourism

*mAA: mean Average Accuracy

| | Method | Num. Inl.↑ | mAA(5°)↑ | mAA(10°)↑ |
|---|---|---|---|---|
| 8k-keypoints | DoG [34]+HardNet [42]+ModifiedGuidedMatching | 762.0 | **0.476** | **0.611** |
| | DoG [34]+HardNet [42]+OANet [76]+GuidedMatching | 765.3 | 0.471 | 0.603 |
| | DoG [34]+HardNet [42]+AdaLAM [11]+DEGENSAC [12] | 627.7 | 0.460 | 0.583 |
| | DoG [34]+HardNet8 [47]+PCA+BatchSampling+DEGENSAC [12] | 583.1 | 0.464 | 0.590 |
| 2k-keypoints | SP [13]+SG [54]+DEGENSAC [12]+SemSeg+HAdapt | (441.5) | (0.452) | (0.590) |
| | SP [13]+SG [54]+DEGENSAC [12]+SemSeg | (404.7) | (0.429) | (0.568) |
| | SP [13]+SG [54]+DEGENSAC [12] | 320.5 | 0.416 | 0.552 |
| | DISK [71]+DEGENSAC [12] | 404.2 | 0.388 | 0.513 |
| | DoG [34]+HardNet [42]+CustomMatch+DGNSC [12] | 245.4 | 0.369 | 0.492 |
| | DoG [34]+HardNet [42]+MAGSAC [3] | 181.8 | 0.318 | 0.438 |
| | DoG [34]+LogPolarDesc [17]+DEGENSAC [12] | 162.2 | 0.333 | 0.457 |
| Ours | **COTR** +DEGENSAC [12] ($N = 2048$) | 1676.6 | 0.444 | 0.580 |
| | **COTR** +DEGENSAC [12] ($N = 1024$) | 840.3 | 0.435 | 0.571 |
| | **COTR** +DEGENSAC [12] ($N = 512$) | 421.3 | 0.418 | 0.555 |
| | **COTR** +DEGENSAC [12] ($N = 256$) | 211.7 | 0.392 | 0.529 |
| | **COTR** +DEGENSAC [12] ($N = 128$) | 106.8 | 0.356 | 0.492 |

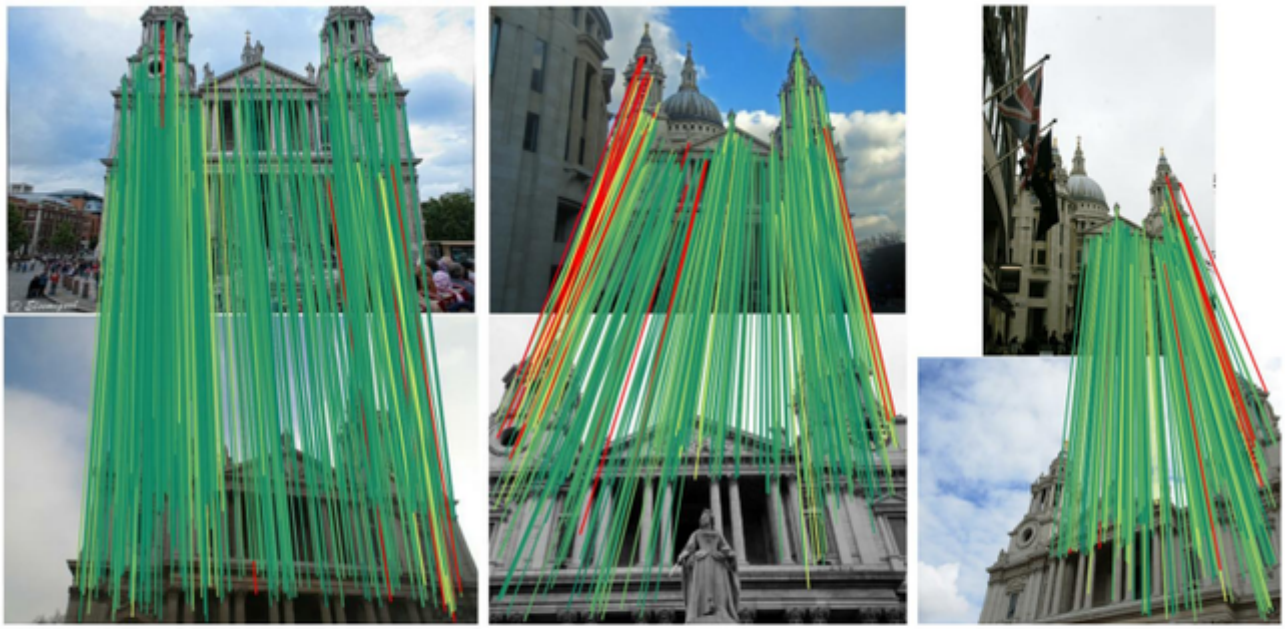


Figure 7. **Qualitative examples for IMC2020 –** We visualize the matches produced by COTR (with $N = 512$) for some stereo pairs in the Image Matching Challenge dataset. Matches are coloured **red** to **green**, according to their reprojection error (high to low).