Simultaneous multi-view instance detection with learned geometric soft-constraints

Ahmed Samy Nassar, Sébastien Lefèvre, Jan D. Wegner

Simultaneous multi-view instance detection with learned geometric soft-constraints

Ahmed Samy Nassar[1,2], Sébastien Lefèvre[1], Jan D. Wegner[2]

[1]IRISA, Université Bretagne Sud

[2]EcoVision Lab, Photogrammetry and Remote Sensing group, ETH Zurich

{ahmed-samy-mohamed.nassar, sebastien.lefevre}@irisa.fr, jan.wegner@geod.baug.ethz.ch

Abstract

We propose to jointly learn multi-view geometry and warping between views of the same object instances for robust cross-view object detection. What makes multi-view object instance detection difficult are strong changes in viewpoint, lighting conditions, high similarity of neighbouring objects, and strong variability in scale. By turning object detection and instance re-identification in different views into a joint learning task, we are able to incorporate both image appearance and geometric soft constraints into a single, multi-view detection process that is learnable end-to-end. We validate our method on a new, large data set of street-level panoramas of urban objects and show superior performance compared to various baselines. Our contribution is threefold: a large-scale, publicly available data set for multi-view instance detection and re-identification; an annotation tool custom-tailored for multi-view instance detection; and a novel, holistic multi-view instance detection and re-identification method that jointly models geometry and appearance across views.

1. Introduction

We propose a method to simultaneously detect objects and re-identify instances across multiple different street-level images using noisy relative camera pose as weak supervision signal. Our method learns a joint distribution across camera pose and object instance warping between views. While object detection in single street-level panorama images is straightforward since the introduction of robust, deep learning-based approaches like Faster R-CNN [25] for object detection or Mask R-CNN [10] for instance segmentation, establishing instance correspondences across multiple views with this wide baseline setting is very challenging due to strong perspective change between views. Moreover, Google street-view panoramas, which are our core data in this paper, are stitched together from multiple individual photos leading to stitching artefacts in addition to motion-blur, rolling shutter effects etc. that are common for these type of mobile mapping imagery. This makes correspondence search via classical structure-from-motion methods like [2, 1] impossible.

Our core motivation is facilitating city maintenance using crowd-sourced images. In general, monitoring street-side objects in public spaces in cities is a labor-intensive and costly process in practice today that is mainly carried out via in situ surveys of field crews. One strategy that can complement greedy city surveillance and maintenance efforts is crowd-sourcing information through geo-located images like proposed for street trees [30, 3, 17]. We follow this line of work, but propose an entirely new simultaneous multi-view object instance detection and re-identification method that jointly reasons across multi-view geometry and object instance warping between views. We formulate this problem as an instance detection and re-identification task, where the typical warping function between multiple views of the same tree (Fig. 2) in street-view panoramas is learned together with the geometric configuration. More precisely, instead of merely relying on image appearance for instance re-identification, we insert heading and geo-location of the different views to the learning process. Our model learns to correlate typical pose configurations with corresponding object instance warping functions to disentangle multiple possibly matching candidates in case of ambiguous image evidence.

Our contributions are (i) a novel multi-view object instance detection and re-identification method that jointly reasons across camera poses and object instances, (ii) a new object instance re-identification data set with thousands of geo-coded trees, and (iii) a new interactive, semi-supervised multi-view instance labeling tool. We show that learning geometry and appearance jointly end-to-end significantly

helps improving object detections across multiple views as well as final geo-coding of individual objects.

2. Related Work

We are not aware of any work that does simultaneous object detection and instance re-identification with soft geometric constraints. But our proposed method touches a lot of different research topics in computer vision like pose estimation, urban object detection, object geo-localization, and instance re-identification. A full review is beyond the scope of this paper and we thus provide only some example literature per topic and highlight differences with the proposed work.

Pose estimation: Learning to predict camera poses using deep learning has been made popular by the success of PoseNet [13] using single RGB images and many works have been published since then [21, 7, 31]. Tightly coupling pose with image content can, for example, be helpful for estimating a human hand's appearance from any perspective if seen from only one viewpoint [24]. Full human pose estimation is another task that benefits from combined pose reasoning across pose and scene content like [20] who employ a multi-task CNN to estimate pose and recognize action. In this paper, we rely on public imagery without fine-grained camera pose information.

Urban object detection: A large body of literature addresses urban object detection from an autonomous driving perspective with various existing public benchmarks, e.g. KITTI [8], CityScapes [6], or Mapillary [22]. In these scenarios, dense image sequences are acquired with minor viewpoint changes in driving direction with forward facing cameras. Such conditions make possible object detection and re-identification across views [5, 16, 37]. In our setup, significant changes occur between views, thus making the re-identification problem much more challenging.

Object geo-localization: Geo-localization of objects from Google street-view imagery with noisy pose information was introduced in [30, 3]. In a similar attempt, [14] geo-localize traffic lights and telegraph poles by applying monocular depth estimation using CNNs, then using a Markov Random Field model to perform object triangulation. The same authors extend their approach by adding LiDAR data for object segmentation, triangulation, and monocular depth estimation for traffic lights [15]. [36] propose a CNN-based object detector for poles and apply a line-of-bearing method to estimate the geographic object position. We rather suggest here to follow an end-to-end learning strategy.

Instance re-identification: Matching image patches can be viewed as a simple version of re-identifying image content across different views, e.g. in structure-from-motion [9], tracking [29], super-resolution [34], depth map estimation [35], object recognition [27], image retrieval [38], and image classification [39]. Our scenario is closely related to works on re-identifying object instances across views. Siamese CNNs have been established as a common technique to measure similarity, e.g. for the person re-id problem that tries to identify a person in multiple views [18]). [33] detects and re-identifies objects in an end-to-end learnable CNN approach with an online instance matching loss. [32] solves re-identification with a so-called center loss that tries to minimize the distance between candidate boxes in the feature space. In contrast to prior work [30, 3, 15], which does detection, geo-coding and re-identification in a hierarchical procedure, our method does it simultaneously in one pass. Methods based on Siamese models [18] alone are not a viable solution to our problem, since they need image crops of the object and can not fully utilize re-identification annotations due to their pairwise labelling training setup. [32] searches for a crop within the detections in a gallery of images, in comparison to our method which aims at matching detections from both full images. The key differences between our work and [33] is that we both ensure object geolocalization and avoid storing features from all identities since that is impractical in a real-world application like the one considered in the paper where objects actually look very similar in appearance.

3. Multi-view detection and instance re-identification

Our method learns to detect and re-identify object instances across different views simultaneously. We compensate for inaccurate or missing image evidence by learning a joint distribution of multi-view camera poses together with the respective warping function of object instances. Intuitively, our method learns to correlate a particular geometric pose setup (e.g., an equilateral triangle, a right triangle, etc.) with the corresponding change of object appearance in the images. As shown in Fig. 3, trees in many situations being the same species, and planted the same time look very similar making it hard to detect or re-identify. Learned relative camera pose configurations help re-identifying object instances across views if appearance information in the images is weak while strong image evidence helps improving noisy camera pose. In general, one can view the relative camera pose estimation task as imposing soft geometric constraints on the instance re-identification task. This joint reasoning of relative camera poses and object appearance ultimately improves object detection, instance re-identification, and also the final object geo-coding accuracy.

A big advantage of this simultaneous computation of relative camera poses, object instance warping and finally object geo-coding is that the model learns to compensate and distribute all small errors that may occur. It thus implicitly learns to fix inaccurate relative poses relying on image evidence and vice versa. An overview of the archi-

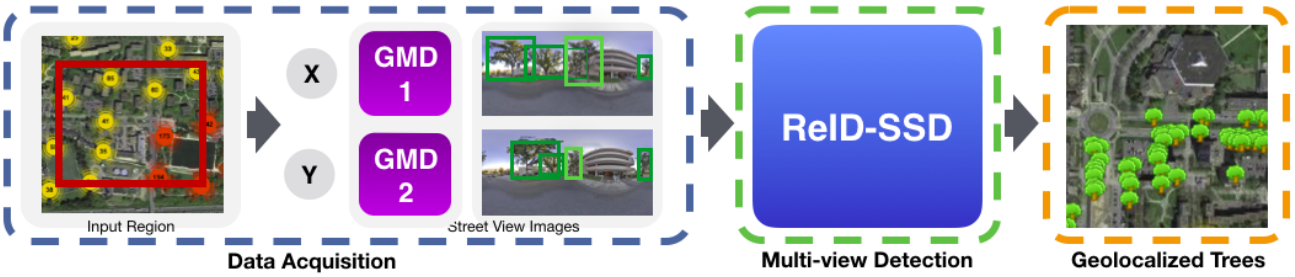

Figure 1. A pair of images is fed to our multi-view object detectors, matching projected predictions is learned, and the geo-coordinate of the object predicted.

tecture of our method is shown in Fig. 4. The basic layout follows a Siamese architecture as proposed originally by [4]. The main concept of Siamese CNNs is constructing two identical network branches that share (at least partially) their weights. Features are computed for both input images and then compared to estimate the degree of similarity. This can be achieved by either evaluating a distance metric in the feature space or by evaluating the final classification loss. Here, our primary data source are Google street-view (GSV) panoramic images along with their geographic meta data (GMD) because they are publicly available at a global scale, fit our purpose of city-scale object mapping for maintenance purposes, and constructing large data sets amenable to deep learning is straightforward. Fig. 2 illustrates the setup of the problem, where the GSV panoramas captured from C^* contain our object of interest T from different viewpoints. The GMD contains many useful properties of the panorama image at hand but location in latitude and longitude as well as yaw are rather inaccurate. Since we do not have any information regarding C's intrinsic or extrinsic properties, we rely on the GMD to use in our projection functions which plays an important role as we will present in the upcoming parts of the paper. GMD is also contains IDs of other images in vicinity.

3.1. Multi-view object detection

Our core object detection network component is based on the single shot detector (SSD) [19]. Our architecture is generally detector-agnostic and any detector could replace SSD if desired. We chose SSD over other prominent methods like Faster R-CNN [25] because SSD provides an easy implementation that allows intuitive modifications and it performs faster with fewer classes, like in our case, while achieving good accuracy [12]. We chose SSD512 [19] as our preferred architecture, which sacrifices a bit of computational speed for better accuracy.

Our network is composed of two identical blocks denoted as $\boldsymbol{X}$ and $\boldsymbol{Y}$ (Fig. 1). As shown in Fig. 4, each block receives an image, camera pose information (geometric meta

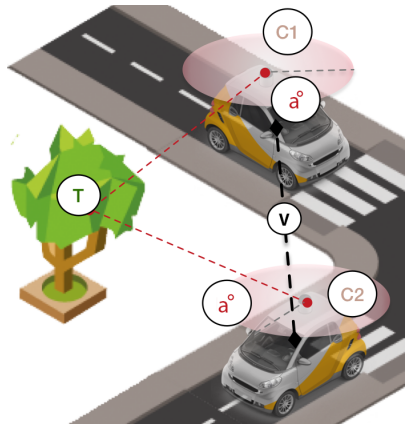

Figure 2. C^*: Camera with geo-position. T: The tree has its actual geographic coordinates, and location within the panorama. a°: heading angle inside panorama. v: Distance between cameras.

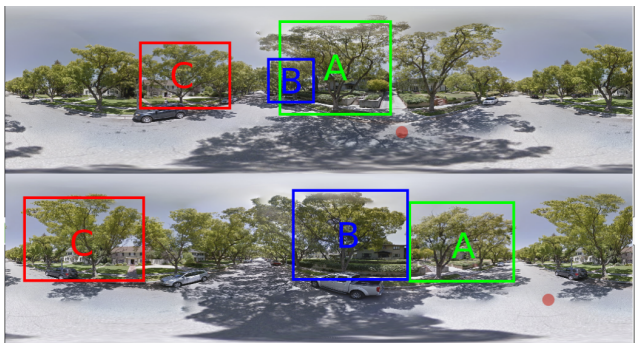

Figure 3. Tree instance re-identification problem (color indicates matches): each tree is photographed from multiple different views, changing its size, perspective, and background. Note that many trees look alike.

data, GMD), and the ground truth during training. Note that the camera's pose information C only contains its location $\ell = (\text{lat}, \text{lng})$, yaw, and height h, which is passed to the network denoted as GMD in Fig. 4. From the GMD data we are able to calculate the distance between the cameras, and the heading angle inside the panorama toward the object, see Eq. (1). Ground truth is composed of two types

of bounding boxes: (i) regular object bounding boxes and (ii) bounding boxes that carry instance IDs labeled and geo-coordinates. Each image passes first through the SSD base network composed of ResNet-50 modules [11]. It is then subject to the convolutional feature layers that provide us with detections at multiple scales. In order to prepare for instance re-identification, each individual object detection is given a *local ID*, which will play a role in the multi-view instance matching stage later on. All detections per network block (i.e., panorama) are then projected during training into the other block's space using our geometric projection, see Eq. (1) & (2). Predictions generated from $\boldsymbol{X}$ and $\boldsymbol{Y}$ are passed through a projection function that estimates their real world geographic position. From this position it is again projected into pixels into the corresponding view. These projection functions assume that the local terrain is flat to simplify the problem. Objects T are represented inside street view images in local East, North, Up (ENU) coordinates that are calculated by providing C_l, C_h and T_l using Eq. (1). To obtain the pixel location of the object $O_{x,y}$, Eq. (2) is used given R as the Earth's radius, W and H the image's width and height respectively, and z the estimated distance from C calculated by $z = \sqrt{e_x^2 + e_y^2}$.

$$(e_x, e_y, e_z) = \big(R\cos[\mathrm{C}_{lat}]\sin[\mathrm{T}_{lng} - \mathrm{C}_{lat}], \\ R\sin[\mathrm{T}_{lat} - \mathrm{C}_{lat}], -\mathrm{C}_h\big) \tag{1}$$

$$x = (\pi + \arctan(e_x, e_y) - \mathrm{C}_{yaw})\, W/2\pi \\ y = (\pi/2 - \arctan(-h, z))\, H/\pi \tag{2}$$

Blindly projecting bounding boxes between panoramas would, however, ignore any scale difference between different images of the same instance. Since the mapping vehicle is moving along while acquiring images, objects close to the camera in one image will likely be further away in the next. In addition, detected bounding boxes may sometimes be fitting an object inaccurately due to partial occlusions or simply poor detector performance. Using the above mentioned equations that assume flat terrain, these errors would results in projections meters away from the true position. We thus add a dense regression network to regress the predicted bounding boxes to the ground truth of the other block once projected. For example, $\boldsymbol{X}$'s projected predictions are regressed to $\boldsymbol{Y}$'s ground truth, and vice versa. This component (Geo Regression Net) aims at taking the predicted boxes, and projected boxes location, and regress them to their real world geo-coordinates through a densely layered network.

Geo Regression Net: Inspired by [23, 28], this network component estimates the geo-coordinates of the detected

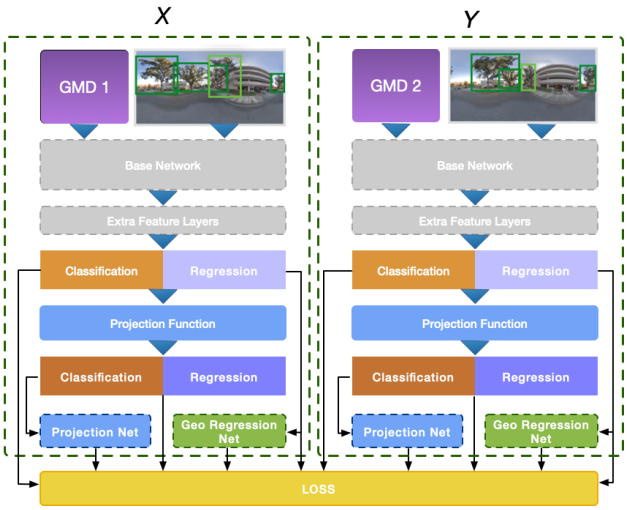

Figure 4. Our network design: Images along with their GMD are inputs to the network. Object bounding boxes and scores for each class are computed via extra feature layers (i.e., Conv4_3 [19]). These are projected into the other image's space (i.e. to the other network block, from X to Y and from Y to X), and input to a dense *Geo Regression Net* to estimate geographic coordinates. Finally, projected predictions are input to the *Projection Net* network component that does some fine-tuning of the projections.

bounding boxes. Note that our "Projection Function" component (based on Eq. (1) and (2)) provides initial estimates for geo-coordinates, which are improved with this component. The Geo Regression Net consists of two dense layers with ReLU activations.

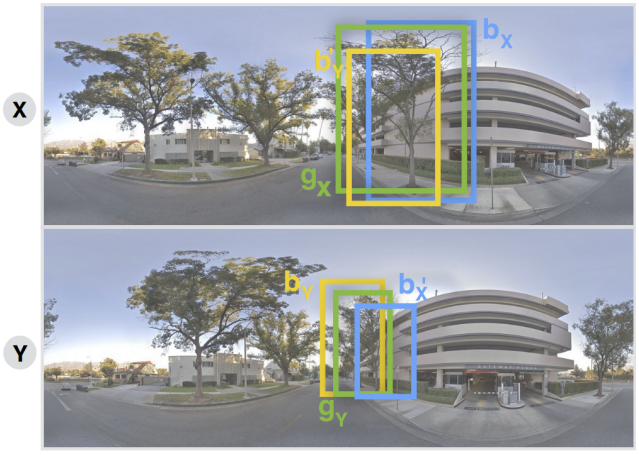

Figure 5. Illustration of predictions (b_*) projected (b'_*) in other views and their ground truth (g_*).

Projection Net: This network component fine-tunes projected predictions b'_* by learning to regress the discrepancy between them and the other block's ground truth as illustrated in Fig 5. Projection Net is constructed similar to the

extra feature layers (Fig. 4), but uses only box regression layers. Our model was trained for 13 epochs, using a single NVIDIA GeForce GTX 1080Ti, with each epoch being trained for approximately 4.5 hours.

3.2. Loss function

We formulate a multi-task loss in Eq. (3) to train our network. Similarly to SSD [19], we use a softmax log loss L_{conf} for classification and a smooth-L1 loss L_{loc} for bounding box regression. As shown in Fig. 5, our predictions b are projected into b' using the projection function. We use again L_{loc} but this time using the ground truth g of the other image g', since it contains the actual bounding boxes we are trying to regress to in order for the *"Projection Network"* to regress the projected boxes. However, mapping which predicted bounding boxes correspond to which default boxes x in g' is not a straightforward task due to how the default boxes are generated systematically:

- boxes inside g and g' are filtered (f_g and f'_g) by keeping only the identified objects (ID'd) to ensure that we are regressing each instance to its corresponding box in the other image,
- b is matched using IoU with f_g to estimate which boxes are our target identities,
- indices of the boxes targeted are then selected to be used as inputs into our loss function, with f'_g as ground truth.

The Geo Regression Net network component is trained using a RMSE L_{RMSE} loss. For the re-identification task, we train both base networks $\boldsymbol{X}$ and $\boldsymbol{Y}$ using a contrastive loss L_{cont} by feeding features from x and x' that are of identified objects as input to learn discriminative features and pull them close if similar. Our complete, multi-task loss function is:

$$\begin{aligned} L_{com}(x,c,b,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,b,g) \\ +\alpha L_{loc'}(x,b',g,g') + L_{cont}(x,g) + L_{RMSE}(b,g)) \end{aligned} \quad (3)$$

During inference, predicted boxes b_* are combined in each view creating a large number of candidate boxes. As in the original implementation of SSD [19], we use a classification confidence threshold of 0.01 to filter redundant boxes. Afterwards non-maximum suppression (NMS) with Jaccard index (IoU) is employed using a 0.5 overlap. As mentioned in Sec. 3.1 the *local IDs* assigned are used to find which remaining candidate bounding boxes when projected, overlap's with the other view's candidate boxes (i.e. $b_X \cap b'_Y$), from which we can identify the corresponding boxes. Simultaneously, by calculating the distance between the candidate boxes from each view using Euclidean distance, we are able to match corresponding boxes.

4. Experiments

We validate our method with experiments on two different data sets with street-level imagery. The first data set consists of GSV panoramas, meta data, and tree object instance labels across multiple views. The second data set contains sequences of Mapillary images acquired with dash cams where object instances are labeled across multiple consecutive image acquisitions. In addition to presenting final results of our end-to-end learnable multi-view object instance detection and re-identification approach, we also do a thorough ablation study to investigate the impact of each individual component.

4.1. Data sets

Pasadena Multi-View ReID: We build a new multi-view data set of street-trees, which is used as a test-bed to learn simultaneous object detection and instance re-identification with soft geometric constraints. The original Pasadena Urban Trees data set [30] contains 1,000 GSV images labeled using Mechanical Turk without explicit instance labels across multiple views. We construct a new *Multi-View ReID* data set for our purpose where each tree appears in those four panoramas that are closest to a particular tree location. In total, we label 6,020 individual tree objects in 6,141 panoramas with each tree appearing in four different panoramas. Each panorama image is of size 2048 x 1024 px. This creates a total of 25,061 bounding boxes, where each box is assigned a particular tree ID to identify the different trees across panoramas. The annotations per image include the following: (i) bounding boxes identified (ID'd) and unidentified, (ii) ID'd bounding boxes include the geo coordinate position, distance from camera v, heading angle a, (iii) image's dimensions, and geo-coordinates. For validating our method experimentally, we split the data set into 4298 images for training, 921 for validation, and 922 for testing. Since we are not aware of any existing multi-view instance labeling tool for geo-located objects, we created a new one described in the sequel.

Mapillary: In order to verify if our method generalizes across different data sets, we run experiments on a data set provided by Mapillary [1]. Note that this particular data set is different from the well-known Mapillary Vistas dataset [22], which provides images and semantic segmentation. Our data set at hand is composed of 31,442 traffic signs identified in 74,320 images and carries instance IDs across views in an area of approximately $2km^2$. On average, two traffic signs appear per image. This data set comes as GeoJSON "FeatureCollection" where each "feature" or identity contains the following properties that were used: (i) the object's geo-coordinate that is achieved by using 3D

[1] www.mapillary.com

structure from motion techniques, therefore it is affected by the GPS, and the density of the images, (ii) the distance in meters from the camera position, (iii) image keys in which the object appears in and which is used to retrieve the image using their API, (iv) geo-coordinates of the image location, (v) the object's altitude, and (vi) an annotation in polygon form of the sign.

The Mapillary data set significantly differs from our tree data set in several aspects. Images were crowd-sourced with forward looking dash cams attached to moving vehicles, and by walking humans using smart phones. Image sizes and image quality are thus inconsistent across the data set. Viewpoint changes between consecutive frames are only of a few meters, and the field of view per image is much smaller than a GSV panorama as shown in Fig. 8. Consequently, the distribution of relative poses between viewpoints is very different as well as the change in appearance of the same object instance across views. Because the camera is forward-looking, each object is viewed more or less from the same viewpoint, only scale changes. However, objects are generally smaller because unlike GSV, no orthogonal views perpendicular to the driving direction exist.

Figure 6. Consecutive frames of two example scenes of the Mapillary data set.

4.2. Multi-view object annotation tool

Labeling object instances across multiple panoramas is a difficult task (Fig. 3) because many trees look alike and significant variations in scale and viewpoint occur. Our annotation tool aims at making multi-view instance labeling more efficient by starting from an aerial view of the scene. To begin labeling, the annotator first selects an individual object from the aerial image. The four closest panoramas are presented to the annotator and in each view a marker appears that roughly points at the object location inside each panorama. This projection from aerial view to street-view panorama approximates the object's position in each of the panoramas and is calculated using Eq. (1) and (2). This initial, approximate object re-identification significantly helps

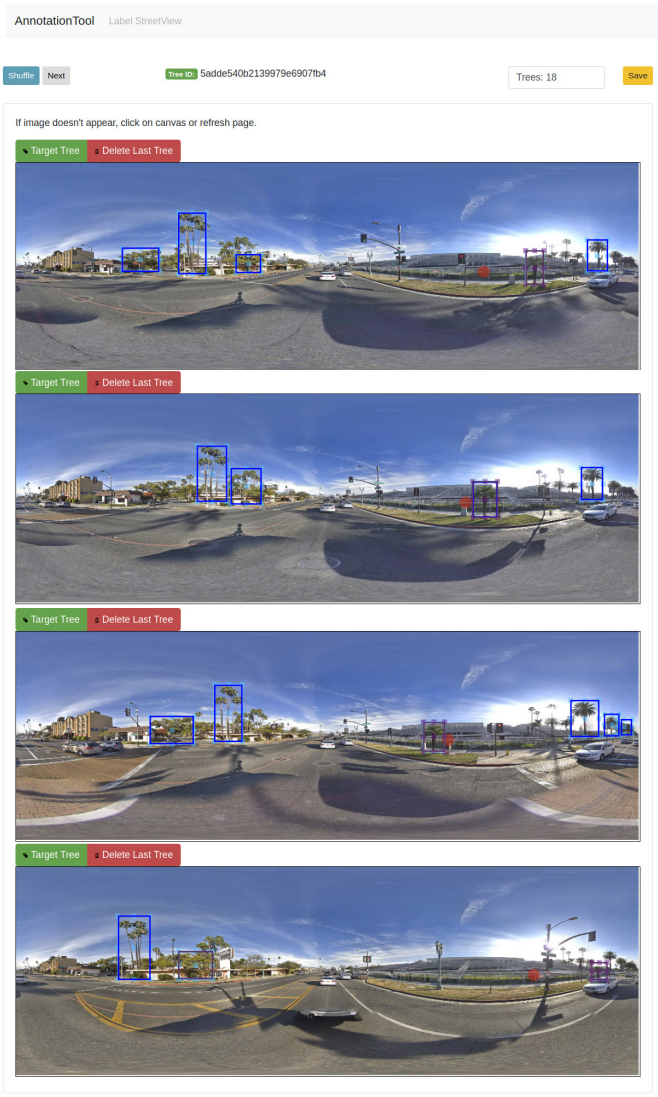

Figure 7. Our annotation tool provides 4 multi-view panoramas from GSV. Initial bounding boxes for the target object are predicted, in which annotators can then refine, or annotate the missing object. To help identifying the object in multiple views a red circle is drawn to estimate the location of the target object in all views to guide the annotator. Images in the figure are from our Multi-View ReID dataset in Pasadena.

a human observer to identify the same tree across different images despite large scale and viewpoint changes. Moreover, SSD predicts bounding boxes around the objects of interest (here: trees) such that the annotator can simply refine or resize an existing bounding box in most cases (or create a new bounding box if the object remains undetected). Identity labeling or correspondence matching is done by selecting the best fitting bounding box (i.e., there may sometimes be more than one bounding box per object) per object per panorama. All multi-view instance annotations are stored in a MongoDB, which enables multiple annotators to work on

Method	w/o Pose [mAP]	w/ Pose [mAP]
FaceNet [26]	0.808	0.842
ResNet-50 [11]	0.828	0.863
MatchNet [9]	0.843	**0.871**

Table 1. Re-identification results without (w/o Pose) and with (w/ Pose) camera pose information (C_l^*, v, a), fed to the Siamese network architectures FaceNet, ResNet-50, and MatchNet.

the same data set at the same time. The database is designed to store annotated bounding boxes to each image, with a separate document storing which bounding boxes are identities. This reduces the effort of having to reannotate each image again for every identity. Our labeling tool is generic in terms of object category and can easily be adapted to any category by re-training the detector component for a different class. Also the detector can be exchanged for any other object detector implemented/wrapped in Python. As output the labeling tool provides annotations through its API in VOC and JSON format.

4.3. Detection

A significant benefit of projecting bounding boxes between blocks of our architecture is that it makes object detection much more robust against occlusions and missed detections caused by missing image evidence in individual images. In order to validate the improvement due to simultaneously detecting objects across multiple views, we compare object detector results on individual panoramas (Monocular) with results from our model that combine object evidence from multiple views via projecting detections between blocks X and Y. Results are shown in Tab. 2. *Ours* improves detection mAP on the Pasadena tree data set by 8.5 percent points, while improving by 2.7 percent points on the Mapillary data set.

4.4. Re-identification with pose information

We verify if learning a joint distribution across camera poses and image evidence supports instance re-identification (regardless of the chosen architecture) with three popular Siamese architectures, namely FaceNet [26], ResNet-50 [11], and MatchNet [9]. Results shown in Tab. 1 indicate that *Ours* with camera pose information consistently outperforms all baseline methods regardless of the base network architecture. Any architecture with added geometric cues does improve performance. Learning soft geometric constraints of typical scene configurations helps differentiating correct from wrong matches in intricate situations. Overall, *Ours* with the MatchNet [9] architecture performs best.

We evaluate the Re-ID mAP for our multi-view setting, which measures the amount of correct instance re-identifications if projecting detections between panoramas in the right column of Tab. 2. To measure the similarity between tree detections across multiple views projected onto one another, we use the distance between overlapping bounding boxes as explained in the inference stage. 73% of all tree instances labeled with identities are matched correctly, which is a high number given the high similarity between neighboring trees and the strong variation in scale and perspective. As for Mapillary's dataset, 88% of the traffic signs were re-identified. In comparison to tree objects, neighboring traffic signs (with different purposes) are easier to discriminate, but much smaller in size.

4.5. Geo-localization

We finally evaluate performance of our end-to-end trainable urban object mapping architecture by comparing predicted geo-locations of trees with ground truth positions. We compare our full, learned model (*Ours*) against simply projecting each detection per single panorama (Single) to geographic coordinates as well as combining detections of multiple views (Multi) (Tab. 3). We compute the discrepancy between predicted geo-coordinates and ground truth object position using the haversine formula given in Eq. (4) with r being the Earth's radius (6,372,800 meters):

$$d = 2r \arcsin\left(\left(\sin^2\left(\frac{b_{lat} - g_{lat}}{2}\right) + \cos(g_{lat})\cos(b_{lat})\sin^2\left(\frac{b_{lng} - g_{lng}}{2}\right)\right)^{0.5}\right) \quad (4)$$

Single view geo-localization was done by applying projection functions given in Eq. (1) and (2) to the individual detections. As for multi-view experiments, we use combined detections from multiple views without learning the projection and project to geographic coordinates as before. *Ours* is our full model as depicted in Fig. 4, which takes advantage of "Projection Net" and "Geo Regression Net" components. Learning multi-view object detection and instance re-identification significantly improves performance, bringing down the MAE to 3.13 meters for the Pasadena trees data set while achieving 4.36 meters for Mapillary. Fig. 9 shows tree detection results (red) for a small example scene in comparison to ground truth locations (orange) overlaid to an aerial view.

5. Conclusion

We have presented a new, end-to-end trainable method for simultaneous multi-view instance detection and re-identification with learned geometric soft-constraints. Quantitative results on a new data set (labeled with a novel

Method	Data set	Det. mAP	Re-ID mAP
Monocular	Pasadena	0.597	-
	Mapillary	0.875	-
Ours	Pasadena	0.682	0.731
	Mapillary	0.902	0.882

Table 2. Detection and Re-identification results with individual, single-view object detections (*Monocular*) compared to our full, multi-view pipeline (*Ours*).

Method	Data set	MAE [m]
Single	Pasadena	77.41
	Mapillary	83.27
Multi	Pasadena	70.16
	Mapillary	64.0
Ours	Pasadena	**3.13**
	Mapillary	**4.36**

Table 3. Geo-localization results as mean absolute error (MAE) compared to geographic ground truth object positions.

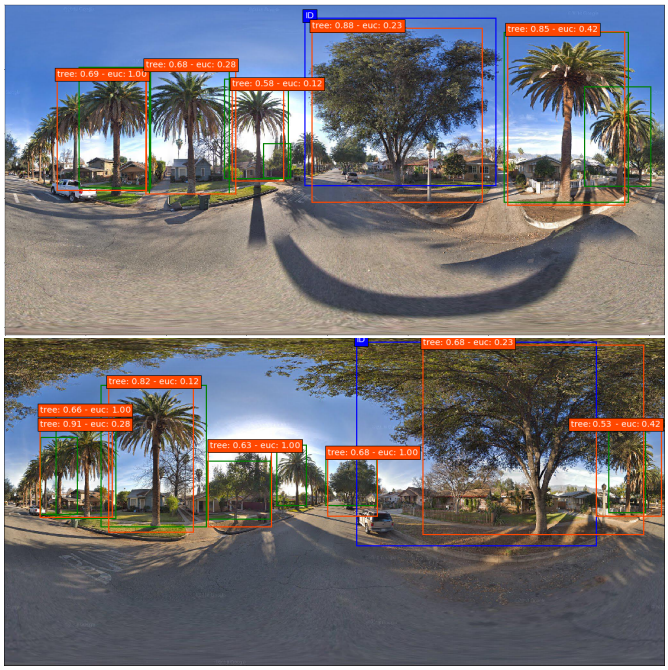

Figure 8. Detection and Re-identification using our method. Green: ground truth boxes. Blue: ground truth box of the identity instance. Orange: predictions with classification score and calculated feature distance from matching box in the other view.

multi-view instance re-identification annotation tool) with street-level panorama images are very promising. Experiments on a Mapillary data set with shorter baselines, smaller objects, narrower field of view, and mostly forward looking cameras indicate that our method generalizes to a different acquisition design, too.

In general, integrating object evidence across views improves object detection and geo-localization simultaneously. In addition, our re-identification ablation study proves that learning a joint distribution across camera poses and object appearances helps re-identification. We hope tight coupling of camera pose information and object appearance within a single architecture will benefit further research on multi-view object detection and instance re-identification in the wild. All source code, the tree detection and re-identification data set, and our new labeling tool will be made publicly available[2].

Acknowledgements: We thank Mapillary Research for providing the Mapillary dataset and the Hasler Foundation for funding our research presented here, which is part of the project "DeepGIS: Learning geographic information from multi-modal imagery and crowdsourcing".

[2]www.registree.ethz.ch

Figure 9. Small subset of tree predictions (red) overlaid to an aerial image (not used in our model) and compared to tree ground truth locations (orange).

References

[1] S. Agarwal, Y. Furukawa, N. Snavely, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing Rome. *IEEE Computer*, pages 40–47, 2010.

[2] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *IEEE International Conference on Computer Vision*, pages 72–79, 2009.

[3] S. Branson, J. D. Wegner, D. Hall, N. Lang, K. Schindler, and P. Perona. From google maps to a fine-grained catalog of street trees. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:13–30, 2018.

[4] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*, pages 737–744, 1994.

[5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[7] S. En, A. Lechervy, and F. Jurie. Rpnet: An end-to-end network for relative camera pose estimation. In *European Conference on Computer Vision*, pages 738–745, 2018.

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[9] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3279–3286, 2015.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[13] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[14] V. Krylov, E. Kenny, and R. Dahyot. Automatic discovery and geotagging of objects from street view imagery. *Remote Sensing*, 10(5):661, 2018.

[15] V. A. Krylov and R. Dahyot. Object geolocation using mrf based multi-sensor fusion. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2745–2749. IEEE, 2018.

[16] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, Oct 2018.

[17] S. Lefèvre, D. Tuia, J. D. Wegner, T. Produit, and A. S. Nassar. Toward seamless multiview scene analysis from satellite to street level. *Proceedings of the IEEE*, 105(10):1884–1899, 2017.

[18] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 152–159, 2014.

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[20] D. C. Luvizon, D. Picard, and H. Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5137–5146, 2018.

[21] Y. Nakajima and H. Saito. Robust camera pose estimation by viewpoint classification using deep learning. *Computational Vision Media*, 3(2):189–198, 2017.

[22] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, pages 5000–5009, 2017.

[23] S. Nilwong, D. Hossain, S.-i. Kaneko, and G. Capi. Outdoor landmark detection for real-world localization using faster r-cnn. In *6th International Conference on Control, Mechatronics and Automation*, pages 165–169. ACM, 2018.

[24] G. Poier, D. Schinagl, and H. Bischof. Learning pose specific representations by predicting different views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 60–69, 2018.

[25] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[26] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[27] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[28] S. Sun, R. Sarukkai, J. Kwok, and V. Shet. Accurate deep direct geo-localization from ground imagery and phone-grade gps. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1016–1023, 2018.

[29] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1420–1429, 2016.

[30] J. D. Wegner, S. Branson, D. Hall, K. Schindler, and P. Perona. Cataloging public objects using aerial and street-level images - urban trees. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6014–6023, 2016.

[31] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robotics: Science and Systems*, 2018.

[32] J. Xiao, Y. Xie, T. Tillo, K. Huang, Y. Wei, and J. Feng. Ian: the individual aggregation network for person search. *Pattern Recognition*, 87:332–340, 2019.

[33] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. Joint detection and identification feature learning for person search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3415–3424, 2017.

[34] C.-Y. Yang, J.-B. Huang, and M.-H. Yang. Exploiting self-similarities for single frame super-resolution. In *Asian conference on computer vision*, pages 497–510. Springer, 2010.

[35] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016.

[36] W. Zhang, C. Witharana, W. Li, C. Zhang, X. Li, and J. Parent. Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images. *Sensors*, 18(8):2484, 2018.

[37] J. Zhao, X. N. Zhang, H. Gao, J. Yin, M. Zhou, and C. Tan. Object detection based on hierarchical multi-view proposal network for autonomous driving. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2018.

[38] Q.-F. Zheng, W.-Q. Wang, and W. Gao. Effective and efficient object-based image retrieval using visual phrases. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 77–80. ACM, 2006.

[39] X. Zhou, K. Yu, T. Zhang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In *European conference on computer vision*, pages 141–154. Springer, 2010.