Object Detection in Equirectangular Panorama

Wenyan Yang, Yanlin Qian, Joni-Kristian Kämäräinen
Laboratory of Signal Processing
Tampere University of Technology, Finland
First.Surname@tut.fi

Francesco Cricri, Lixin Fan
Nokia Technologies
Tampere, Finland

***Abstract*—We introduce a high-resolution equirectangular panorama (aka 360-degree, virtual reality, VR) dataset for object detection and propose a multi-projection variant of the YOLO detector. The main challenges with equirectangular panorama images are i) the lack of annotated training data, ii) high-resolution imagery and iii) severe geometric distortions of objects near the panorama projection poles. In this work, we solve the challenges by I) using training examples available in the "conventional datasets" (ImageNet and COCO), II) employing only low resolution images that require only moderate GPU computing power and memory, and III) our multi-projection YOLO handles projection distortions by making multiple stereographic sub-projections. In our experiments, YOLO outperforms the other state-of-the-art detector, Faster R-CNN, and our multi-projection YOLO achieves the best accuracy with low-resolution input.**

I. Introduction

360-degree (360°) video and image content has recently gained momentum due to wide availability of consumer-level video capture and display devices - "Virtual Reality (VR) gear". Equirectangular panorama (ERA, Figure 1) has quickly become the main format to store and transmit VR video. ERA images create new challenges for computer vision and image processing as i) we lack annotated 360 datasets for many problems, ii) imagery are often of high-resolution to cover the viewing sphere with reasonable resolution and iii) equirectangular projection creates severe geometric distortions for objects away from the central horizontal line.

In computer vision community, there are several recent works on processing 360 video and images, for example, "compression" of wide angle VR video to conventional narrow angle video [1], [2], equirectangular super-resolution [3] and 360-degree object tracking [4]. In this work, we focus on visual object class detection in equirectangular panorama and provide a novel 360-degree dataset and evaluate state-of-the-art object detectors in this novel problem setting.

We train two state-of-the-art detectors, Faster R-CNN [5] and YOLO (version 2) [6], with conventional examples available in the existing datasets (ImageNet and COCO in our case) and test them with 360-degree data. In our experiments, the YOLO detector performs better than Faster R-CNN, but for dense scenes of many objects YOLO needs high-resolution input (Sec. IV-C). To adapt the YOLO detector for less computation power, we propose a multi-projection variant of the original YOLO detector. Our m-p YOLO employs stereographic projection and post-processing with soft non-maximum suppression (soft-NMS) and outperforms both Faster R-CNN and YOLO. Our novel contributions in this work are:

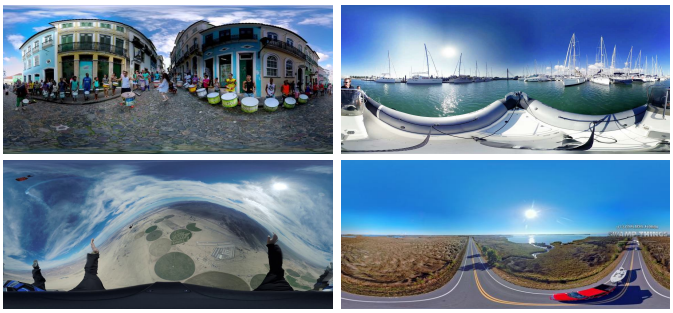

Figure 1. Frames from 360° videos captured by users and uploaded to Youtube (distortions are due to the equirectangular panorama projection). In the bottom-left the projection horizon does not match to the scene horizon and needs to be fixed (automatically done in our dataset).

- We introduce a high-resolution equirectangular panorama dataset for evaluating and benchmarking object detectors with 360-degree data (Sec. II).
- We compare two state-of-the-art object detectors, Faster R-CNN [5] and YOLO (version 2) [6] trained with conventional data (ImageNet and COCO) in 360-degree object detection (Sec. IV).
- We propose a multi-projection variant of YOLO that achieves high accuracy with moderate GPU power (Sec. III).

Our dataset and code will be made publicly available.

II. Equirectangular Panorama Dataset

360° virtual reality images and video frames are stored as 2D projections of the captured 3D world on the surface of a *viewing sphere*. The most popular projection is *equirectangular panorama* (ERA) in which each sphere point is uniquely defined by two angles [7]: *latitude* $\varphi \in [-90°, +90°]$ and *longitude* $\lambda \in [-180°, +180°]$. In this work, we employ equirectangular panorama images.

For our dataset we selected 22 4k-resolution (3840×1920) VR videos captured and uploaded by users to Youtube. From each video we selected a number of frames and annotated a set of visual classes in each frame. Our dataset consists of the total of 903 frames and 7,199 annotated objects. For the experiments, we selected objects that are also available in the Microsoft COCO dataset [8] to experiment training

with examples in conventional images. The selected videos represent dynamic activities, for example, skateboarding or riding bicycles, are all real world scenes, and contain both moving and static objects.

A. Object Annotations

An important difference between equirectangular panorama and conventional images is that the panorama projection changes object's appearance depending on its spatial location (Figure 1). Therefore for consistent annotation, we need to select a "canonical pose" where bounding box coordinates are valid and has the shape of a box. We implemented an annotation tool where the user is shown a perspective projection of the full panorama. User is allowed to change the center of projection and field-of-view (FOV). During the annotation workshop, annotators were asked to move an object to the central view where the object center approximately matches the projection axis and then annotate a bounding box. As for the ground truth we store the following attributes: object label l^i; the bounding box center as angular coordinates φ^i, λ^i; and the bounding box angular dimensions $\Delta\varphi^i$, $\Delta\lambda^i$. We refer such annotation as BFOV (Boundig FOV, see Figure 2).

It should be noted that the above annotation protocol has problems with objects that are close to the VR camera. Annotating these objects requires a very wide FOV ($\geq 120°$) which makes annotation sensitive to the selection of the projection center. See Figure 3 for examples of the annotation ambiguity problem with objects too close to the camera. We manually identified these objects and re-annotated them directly in the ERA image. Hu et al. [9] avoided this problem in their recent work by selecting images where is only one "dominating object", but in our dataset images contain multiple objects which are all annotated. All experiments are conducted using the original or corrected bounding boxes in the original equirectangular panorama frames.

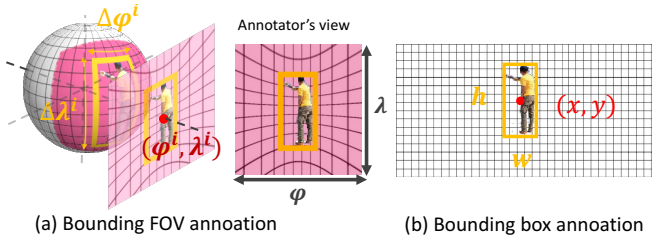

Figure 2. Example of our annotation protocol: a) user centers the object and annotates "Bounding FOV" (BFOV) in center and FOV angles. b) BBox (bounding box) coordinates are directly annotated on the original panorama.

B. 360-degree dataset vs. COCO dataset

One target in our experiments is to leverage the pre-trained detectors (YOLO [6] and Faster R-CNN [5] on Microsoft COCO [8] and ILSVRC2012 [10]) on equirectangular images. However, it is unclear how well object appearance match between the VR and conventional datasets. Our annotated bounding boxes in our and COCO datasets are plotted in Figure 4 and Figure 5. BBox dimensions (width and height normalized to $[0, 1]$) are plotted in Figure 4 to compare the aspect ratios of annotated bounding boxes. Figure 5 shows how BBoxes are distributed on images. It is clear that COCO bounding box aspect ratios span wider range than our dataset and locations of bounding boxes for our datasets are more limited. This verifies that appearance of COCO objects match to sufficient degree the appearance of objects in our dataset (except objects near the VR camera).

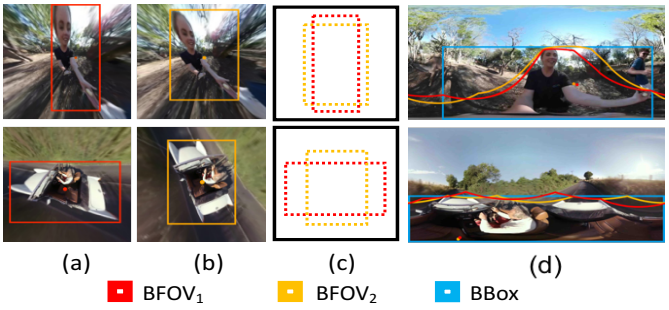

Figure 3. Example of annotation ambiguities of objects near the VR camera. (a) and (b) show bounding boxes annotated by two different annotators who both set annotation tool field-of-view to $\varphi = 150°$, $\lambda = 150°$. (c) illustrates bounding box ambiguity due to sensitivity to the bounding box center that defines the projection angle (cf. (a) and (b)). (d) illustrates the annotated bbs in the ERA image and blue box is the corrected bounding box.

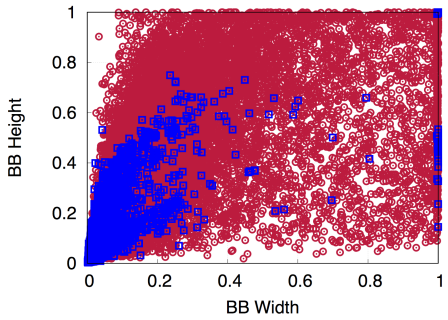

Figure 4. Bounding box aspect ratios (normalized width and height) in the COCO (red circles) and our (blue squares) dataset.

The main differences in our dataset as compared to COCO can be summarized as:

- In our dataset there are more small objects. This can be explained by the nature of equirectangular panorama projection where scale quickly changes with respect to object distance. Therefore accurate object detection also requires high-resolution. This is evident in Figure 4 where our objects span clearly smaller areas in images.
- Geometric distortion due to equirectangular projection. However, in many cases people prefer to capture main targets near the central horizontal line where distortions are small if the objects are not too close to the VR camera (Figure 5).

For our experiments we selected $6,431$ annotated objects that are also available in the COCO dataset: *person* (3720), *car*

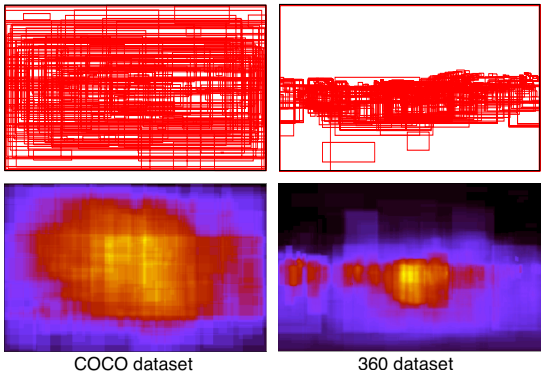

Figure 5. Bounding boxes location distribution in COCO (left) and our dataset (right) plotted to a single image. Heat maps below.

(1610), *boat* (963). Another annotated classes are skateboard, bicycle, truck, sign, ball, penguin and lion, but since a) there were not enough annotated objects in them or b) they are not available in COCO these were omitted in our experiments.

III. Multi-projection YOLO

In our experiments we report results for the recent state-of-the-art object detector YOLO [6] which is pre-trained on ImageNet data and fine-tuned with COCO. However, processing full panorama with sufficient accuracy requires high-resolution input. It requires high-end GPU (e.g. TITAN-X) to fit all data to memory and therefore we propose a multi-projection YOLO detector. Multi-projection YOLO can be computed on consumer GPUs and provides competitive accuracy.

As discussed in Section II-A, One of the main challenges of object detection in equirectangular panorama is severe geometric distortions caused by panorama projection. A straightforward solution to remove distortions is to project sub-widows of the full 360-degree sphere onto 2D image plane for which detection methods trained on conventional images should perform well. However, the smaller is the sub-window field-of-view (FOV) the more windows are needed and therefore more processing is needed. Some recent works argue that this approach is not optimal [11], [12], but methods using re-projections have also been proposed [13].

In our work, we investigate the multi-projection approach with a wide FOV and adopt soft selection as an alternative to the standard non-maximum suppression (NMS) to select detections produced by multiple windows (see Figure 8 for overview of our approach).

A. Sphere-to-plane projection

There are various popular projections to map a sphere to a plane and these are mainly used in cartography applications [14]. The standard approach is to consider a tangent plane on a sphere and adjust projection center and FOV in order to produce sphere-to-plane projection (see Figure 6). In the following we assume that the sphere has radius $r = 1$, the projection direction is toward the positive z-axis, and the tangent plane center is located at $z = 1$. Now, a point $p(\theta, \phi)$ on the sphere is projected onto point $P(x_p, y_p, 1)$ as

$$\begin{aligned} y_p &= \frac{d+1}{d + Rcos(\theta)} Rsin(\theta) \\ x_p &= \frac{d+1}{d + Rcos(\phi)} Rsin(\phi) \end{aligned} \tag{1}$$

If the projection center d in (1) is set to 0, the projection corresponds to the popular *perspective projection* and for $d = 1$ the projection is *stereographic projection*. These two projections provide different images as illustrated in Figure 7.

Many projection models are proposed to minimize object appearance distortions. Chang et al. [15] proposed *rectangling stereo projection* which makes projection in two steps. At first, they project original sphere onto a swung sphere, and then in the second step they generate perspective projection from the swung sphere. Their method preserves linear structures. Another distortion minimizing projection is *automatic content-aware projection* by Kim et al. [16]. In their work, model interpolation between local and global projections is used to adjust distortions in optimized Pannini projection.

In our experiments, we adopted stereographic projection model due to its simplicity. But it is easy to replace it with other models in our framework.

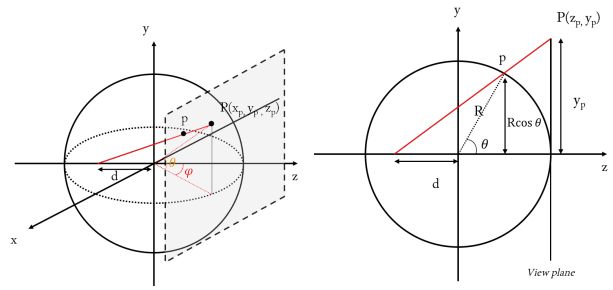

Figure 6. Illustration of the general sphere-to-plane projection in Eq. (1).

Figure 7. Perspective (left) and stereographic (right) projections of the same spherical image. The FOV for perspective projection is set as h = 90°, w = 175° ; The FOV for stereoprojection is set as h = 90°, w = 180°. For wide-angle projection, stereographic model preserves better information although straight lines are not preserved.

B. Bounding box selection

Detection – Two stereographic projections with horizontal and vertical span of 180° cover the whole sphere. However, such wide angle projections still produce large geometric distortions for objects near the projection edges. To compensate that, we adopt four sub-windows with overlap of 90°. In our

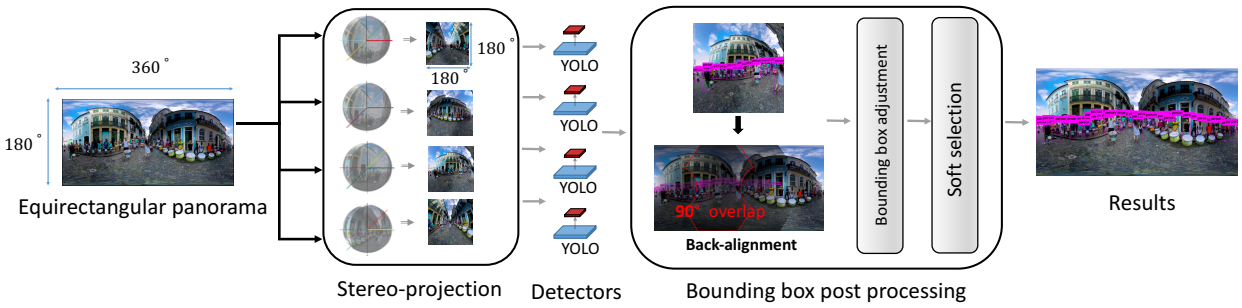

Figure 8. Overall processing pipeline of our multi-projection YOLO. In the first step, we generate four stereographic projections for which their horizontal and vertical spans are 180°. The horizontal overlap between two neighbor projections is 90°. Then each sub-projection is separately processed by the YOLO detector. In the post-processing part, bounding box adjustment part fixes distorted detection boxes (Figure 9) and soft-selection re-scores redundant boxes in overlapping areas (Section III-B)

experiments, we use YOLOv2 [6] and it produces a set of object detections with bounding boxes and detection scores for each sub-window.

Re-aligning bounding boxes – The YOLO detections are

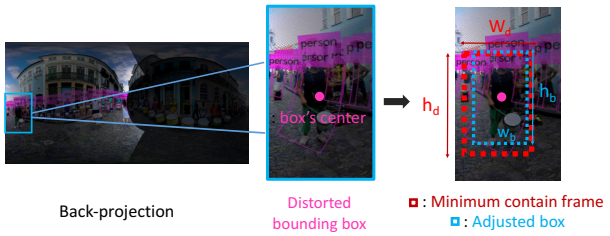

Figure 9. Re-aligning YOLO detector bounxing boxes. Sub-window detected bounding boxes are projected to the panorama (left) where they are distorted (middle). Distortions are removed by fitting a "minimum frame" and then using a soft penalty to fix the bounding box size.

back-projected to the equirectangular panorama. Due to projection model's feature (Sec III-A), this process severely bends the lines at the edges and make evaluation difficult. Thus the BBoxes need to be re-aligned. We exploit the fact that bounding box center remains unaffected by the distortion and we therefore re-adjust the edges around the center. We first find each bounding box a "minimum frame" that contains the back-projected detection box. The minimum frame has its width and height w_d, h_d. The final width and height of the adjusted box w_b and h_b are set based on the minimal frame size w_d and h_d and the distance d to the box center. Our adjustment is based on [17] and the final bounding box size is set as (Fig. 9). The penalty parameter is σ and is manually set in experiments.

$$w_b = w_d e^{-(\frac{d^2}{\sigma})}, \quad h_b = h_d e^{-(\frac{d^2}{\sigma})} \quad . \tag{2}$$

Soft selection – Since two neighbor sub-windows overlap by 90° we need to post-process the boxes in those areas. Non-maximum suppression (NMS) is the standard technique, but yields to problems in our case since we have wide FOV detectors that are more reliable at the center of the sub-window. Therefore, we do NMS on every sub-window separately and for the remaining objects, we keep them all, but re-score them based on their location in the detection window and their overlap. This approach is inspired by [17]. We do re-scoring by a soft-penalty function using the following rule:

$$s'_i = s_i e^{-(\frac{IoU(\hat{b},b_i)^2}{\sigma_1} + \frac{d_i^2}{\sigma_2})} \tag{3}$$

where s_i is the original YOLO score of the bounding box detection b_i with the same label as $\hat{b}$, but futher away from the sub-window center than $\hat{b}$. Penalty is affected by the amount of intersection measured by Intersection-over-Union $IoU(\hat{b}, b_i)$ and the distance d_i from the center of the sub-window producing the detection. The balance between the two terms is set by the parameters σ_1 and σ_2 and the distance d_i is normalized to $[0, 1]$. Intuitively, the larger is the overlap and further the detection is away from the detection window center the more the score will be suppressed.

IV. Experiments

In this section, we first study performance of the two state-of-the-art detectors, *YOLOv2* by Redmon et al. [6] and *Faster R-CNN* by Ren et al. [5] with various input resolutions. In the second set of experiments we investigate our multi-projection YOLO with low-resolution inputs.

A. Settings

In our experiments, we used the publicly available ready-to-be-used models of YOLOv2[1] and Faster R-CNN[2] as the baselines and both were optimized for the Pascal VOC data. We fine-tuned the pre-trained baseline models using training images of all classes in the Microsoft COCO dataset [8]. Faster R-CNN uses the *VGG-16* network [18] for classification and VGG-16 is pre-trained using ILSVRC2012[3] training and validation data.

B. YOLO vs. Faster R-CNN

In the first experiment, we compared the two state-of-the-art detectors, Faster R-CNN and YOLO, to detect objects from

[1] https://pjreddie.com/darknet/yolo/
[2] https://github.com/rbgirshick/py-faster-rcnn
[3] www.image-net.org/challenges/LSVRC/2012/

Table I
AVERAGE PRECISIONS OF FASTER R-CNN [5] AND YOLO (VER 2) [6]. R-CNN RESIZES ITS INPUT SO THAT THE SMALLEST DIM. IS 600 PIXELS WHILE YOLO CAN PROCESS INPUT OF ANY SIZE. OUR DATASET'S FRAME SIZE IS 3840×1920 (ASPECT RATIO $2:1$).

	Tr size	*Person*	*Car*	*Boat*	mAP
trained with Pascal VOC					
YOLOv2 416×416	-	23.63	24.90	10.60	19.71
YOLOv2 608×608	-	30.94	26.09	**20.75**	25.93
Faster R-CNN	-	33.13	25.53	4.33	21.00
trained with COCO					
YOLOv2 416×416	416×416	24.00	17.02	3.63	14.88
YOLOv2 608×608	608×608	**39.48**	**32.19**	19.19	**30.29**
Faster R-CNN	-	27.28	15.90	13.04	18.74

full equirectangular panorama input. We conducted two kind of experiments: first with the original detectors provided by the corresponding authors and the second by re-training the same detectors with examples in the COCO dataset. As indicated by the results in Table I YOLOv2 always achieves better accuracy than Faster R-CNN and since it also faster to train we selected it for the remaining experiments.

C. YOLO input resolution

Table II
AVERAGE PRECISIONS FOR YOLOV2 TRAINED AND TESTED WITH INPUTS OF VARIOUS RESOLUTIONS (GRID SIZE IS KEPT CONSTANT IN PIXELS).

	Tr size	*Person*	*Car*	*Boat*	mAP
YOLOv2 416×416	416×416	24.00	17.02	3.63	14.88
YOLOv2 608×608	416×416	38.23	24.37	14.05	25.55
YOLOv2 864×864	416×416	40.67	24.08	15.79	26.85
YOLOv2 416×416	608×608	30.82	24.08	11.60	22.17
YOLOv2 608×608	608×608	39.48	32.19	19.19	30.29
YOLOv2 864×864	608×608	46.40	30.58	15.73	30.90
YOLOv2 864×416	864×416	43.17	29.06	15.13	29.12
YOLOv2 1696×864	864×416	**59.87**	**32.52**	**19.84**	**37.41**

Since the equirectangular panorama images are of high-resolution (3840×1920) we wanted to investigate the two important parameters of the YOLO method: size of the training data (COCO) and size of the input. During the experiments the bounding box grid size was kept constant in pixels and therefore for larger images YOLO is able to detect smaller examples and provide more dense bounding box detections. The results in Table II confirm that a higher training resolution always provides better accuracy and also higher input resolution always provides better accuracy. The maximum training resolution with our GPU card was 864×416 after which the mini batch size is enforced to a single image and training becomes inefficient. By fixing the grid cell size in pixels the YOLO improved in detection of small objects close to each other. This is beneficial as can be seen in Figure 4 which shows how our dataset contains a larger proportion of small objects as compared to COCO. The boat class performs poorly as compared to other classes, but this can be explained by the facts that many boat pictures are taken from the annotated boat itself making it highly distorted (too close to the main lenses and in periphery of the capturing device) and many boats are hydrocopters which are not well presented in COCO dataset.

D. Multi-projection YOLO

In the last experiment, we studied processing with limited GPU power in which case we must process the input image as "windowed" version using our multi-projection YOLO (Section IV-C). We fixed the input resolution to 864×864 and used the training resolution of 608×608. The results are shown in Table IV and clearly illustrate how windowed YOLO performs better than full panorama YOLO with limited resolution (trained with the same resolution) and how our soft-selection outperforms non-maximum suppression (NMS). Fig. 10 also shows the examples of human dection. For single YOLO method, with input resolution incearsing the detection performs better. For windowed YOLO (m-p YOLO), it performs better when they have the same input resolution. Although somes bounding boxes closed to edge were enlarged due to bounding box post processing (Sec III-B).

Figure 10. Examples of human detections with different resolutions and with multi-projection YOLO. Best viewed on display.

E. Ablation studies

The results of the first ablation study are shown in Table III. In this experiment we tested the effect of the overlapping penalty parameter σ_1 and the distance penalty parameter σ_2. Moreover, we compared our results to the most popular post-processing method: non-maximum suppression (NMS) with the default threshold set to 0.3. For all penalty term values our approach achieved better accuracy than NMS and penalty term optimization by cross-validation resulted to the best accuracy with $\sigma_1 = 0.3$ and $\sigma_2 = 0.6$.

Table III
PENALTY PARAMETERS FOR MULTI-PROJECTION YOLO.

AP @ 0.5	*Person*	*Car*	*Boat*	*mAP*
$\sigma_1 = 0.3, \sigma_2 = 0.3$	49.71	34.86	12.21	32.26
$\sigma_1 = 0.3, \sigma_2 = 0.6$	**54.51**	**35.42**	12.93	**34.29**
$\sigma_1 = 0.3, \sigma_2 = 0.9$	51.35	34.83	12.20	32.79
$\sigma_1 = 0.6, \sigma_2 = 0.3$	51.35	34.79	12.20	32.78
$\sigma_1 = 0.6, \sigma_2 = 0.6$	51.26	34.76	12.17	32.73
$\sigma_1 = 0.6, \sigma_2 = 0.9$	51.10	34.60	12.16	32.62
$\sigma_1 = 0.9, \sigma_2 = 0.3$	51.30	34.72	12.16	32.73
$\sigma_1 = 0.9, \sigma_2 = 0.6$	51.08	34.52	12.16	32.34
$\sigma_1 = 0.9, \sigma_2 = 0.9$	50.88	34.20	12.19	31.41
NMS	47.47	30.96	12.60	26.31
YOLOv2 (tr. 608×608)	46.40	30.58	**15.73**	30.90
AP @ 0.4	*Person*	*Car*	*Boat*	*mAP*
$\sigma_1 = 0.3, \sigma_2 = 0.3$	62.19	40.30	24.07	42.19
$\sigma_1 = 0.3, \sigma_2 = 0.6$	**64.07**	40.26	24.05	42.79
$\sigma_1 = 0.3, \sigma_2 = 0.9$	64.03	42.04	24.05	**43.37**
$\sigma_1 = 0.6, \sigma_2 = 0.3$	64.03	40.12	24.01	42.72
$\sigma_1 = 0.6, \sigma_2 = 0.6$	62.24	**43.13**	24.09	43.15
$\sigma_1 = 0.6, \sigma_2 = 0.9$	63.64	41.36	24.00	43.00
$\sigma_1 = 0.9, \sigma_2 = 0.3$	63.96	41.65	23.99	43.20
$\sigma_1 = 0.9, \sigma_2 = 0.6$	63.63	41.25	24.20	43.02
$\sigma_1 = 0.9, \sigma_2 = 0.9$	63.34	40.89	23.88	42.70
NMS	62.66	38.27	24.43	41.79
YOLOv2 (tr. 608×608)	52.27	38.43	**30.41**	40.37

Table IV
OUR MULTI-PROJECTION YOLO FOR LOW-RES. (864×864) INPUT WITH WEIGHTS TRAINED ON (608×608) SIZE.

	Person	*Car*	*Boat*	*mAP*
YOLOv2	46.40	30.58	**15.73**	30.90
m-p YOLO (NMS)	47.47	30.96	12.60	26.31
m-p YOLO	**54.51**	**35.42**	12.93	**34.29**

Another important consideration with our dataset is that since bounding boxes are annotated in central view they distort when bounding box is moved to objects' original location (the same effect does not occur with conventional images). However, in the standard evaluation protocol the bounding box overlap limit for successful detection is 0.5 which is much harder to achieve in equirectangular panorama setting. To experiment this, we tested our penalty terms in the setting where the overlap threshold was relaxed to 0.4. The results are also shown Table III and illustrate how the performance improved significantly for all configurations.

V. CONCLUSIONS

We studied the problem of object detection in 360-degree VR images (equirectangular panorama) and proposed a novel benchmark dataset. Moreover, we compared two state-of-the-art methods and showed superior performance of the recently proposed YOLO version 2. However, good performance requires high-resolution input (testing stage) which is not suitable for low-end GPUs (out of memory). To compensate lower processing power we proposed a multi-projection YOLO method that is superior in the low-resolution setting. Our data and code will be made publicly available.

ACKNOWLEDGEMENT

This work was partially funded by the Business Finland project "360° Video Intelligence - For Research Benefit" and participating companies (Nokia Technologies, Lynx Technology, JJ-Net, BigHill Companies and Leonidas).

REFERENCES

[1] Y.-C. Su and K. Grauman, "Pano2vid: Automatic cinematography for watching 360° videos," in *ACCV*, 2016. 1
[2] ——, "Making 360 video watchable in 2d: Learning videography for click free viewing," in *CVPR*, 2017. 1
[3] V. Fakour-Sevom, E. Guldogan, and J.-K. Kämäräinen, "360 panorama super-resolution using deep convolutional networks," in *Int. Conf. on Computer Vision Theory and Applications (VISAPP)*, 2018. 1
[4] U. Kart, J.-K. Kämäräinen, L. Fan, and M. Gabbouj, "Evaluation of visual object trackers on equirectangular panorama," in *Int. Conf. on Computer Vision Theory and Applications (VISAPP)*, 2018. 1
[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Neural Information Processing Systems (NIPS)*, 2015. 1, 2, 4, 5
[6] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *CVPR*, 2017. 1, 2, 3, 4, 5
[7] J. Snyder, *Map Projections - A Working Manual*. U.S. Department of the Interior, 1987. 1
[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. Zitnick, "Microsoft COCO: Common objects in context," in *ECCV*, 2014. 1, 2, 4
[9] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun, "Deep 360 pilot: Learning a deep agent for piloting through 360 sports video," in *CVPR*, vol. 1, no. 2, 2017, p. 3. 2
[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int J Comput Vis*, vol. 115, pp. 211–252, 2015. 2
[11] R. Khasanova and P. Frossard, "Graph-based classification of omnidirectional images," *arXiv preprint arXiv:1707.08301*, 2017. 3
[12] Y.-C. Su and K. Grauman, "Flat2sphere: Learning spherical convolution for fast features from 360 imagery," in *Advances in Neural Information Processing Systems*, 2017, pp. 529–539. 3
[13] R. Carroll, M. Agrawal, and A. Agarwala, "Optimizing content-preserving projections for wide-angle images," in *SIGGRAPH*, 2009. 3
[14] J. P. Snyder, *Map projections–A working manual*. US Government Printing Office, 1987, vol. 1395. 3
[15] C.-H. Chang, M.-C. Hu, W.-H. Cheng, and Y.-Y. Chuang, "Rectangling stereographic projection for wide-angle image visualization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2824–2831. 3
[16] Y. W. Kim, D.-Y. Jo, C.-R. Lee, H.-J. Choi, Y. H. Kwon, and K.-J. Yoon, "Automatic content-aware projection for 360? videos," *CoRR*, 2017. 3
[17] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-nms–improving object detection with one line of code," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, no. 3, 2017, p. 4. 4
[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015. 4