BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

DEEP LEARNING ALGORITHMS FOR BACKGROUND SUBTRACTION AND PEOPLE DETECTION

by

M. OZAN TEZCAN

B.S., Koç University, 2016
M.S., Boston University, 2021

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

2021

© 2021 by
M. OZAN TEZCAN
All rights reserved

Approved by

First Reader

Janusz Konrad, Ph.D.
Professor of Electrical and Computer Engineering

Second Reader

Prakash Ishwar, Ph.D.
Professor of Electrical and Computer Engineering
Professor of Systems Engineering

Third Reader

Brian Kulis, Ph.D.
Associate Professor of Electrical and Computer Engineering
Associate Professor of Systems Engineering
Associate Professor of Computer Science

Fourth Reader

Pierre-Marc Jodoin, Ph.D.
Professor of Computer Science
Université de Sherbrooke

Acknowledgments

First, I would like to express my gratitude and thanks to my research advisor, Prof. Janusz Konrad, for his help and support during my Ph.D. This work would not have been possible without his expertise and guidance. He is also a kindhearted mentor who is always open to discussing research- and career-related problems. Secondly, I would like to thank Prof. Prakash Ishwar for his help and support. His mathematical clarity and attention to detail helped me a lot to shape my research. Thirdly, my thanks go to my other dissertation committee members, Prof. Pierre-Marc Jodoin and Prof. Brian Kulis, for providing feedback on my research.

Apart from my committee members, I would also like to express my thanks to my research collaborators, Josh Bone, Mertcan Cokbas, Zhihao Duan, Shengye Li, Qili Zeng and Prof. Jordana Muroff. It was an honor to work with all of them. My thanks also go to Ragib Ahsan, John Bolognino, Benjamin Chan and Nancy Zheng for their tremendous help with video annotations, which were critical for part of my research.

I would like to acknowledge my friends from BU as well for their support and encouragement: Mertcan Cokbas, Dr. Jiawei Chen, Dr. Jinyuan Zhao, Alp Acar, Burak Aksar, Dr. Atakan Ari, Dr. Tolga Bolukbasi, Sadullah Canakci, Kubra Cilingir, Dr. Andrew Cutler, Furkan Eris, Cagatay Karakan, Ahmet Can Kirlioglu, Christy Lin, Dr. Onur Sahin, Verda Saygin, Dr. Ozan Tuncer, Dr. Celalettin Yurdakul and Onur Zungur. They brought fun and joy to this challenging journey.

My warmest thanks go to my parents, Dr. Hanife Tezcan and Dr. Harun Tezcan; sister, Elif Beste Tezcan and significant other, Beste Ozer, for their endless love and support. They were always with me on my good and bad days. I would not be able to finish this work without their support.

Finally, I would like to thank the Advanced Research Projects Agency – Energy (ARPA-E) and the College of Engineering, Boston University for financial support of my research. Also, I would like to acknowledge in-kind support from NVIDIA Corporation through a donation of Titan GPUs that I used for some of this research. The views in this document are those of the author and do not represent those of the funding sources.

DEEP LEARNING ALGORITHMS FOR BACKGROUND SUBTRACTION AND PEOPLE DETECTION

M. OZAN TEZCAN

Boston University, College of Engineering, 2021

Major Professor: Janusz Konrad, PhD
Professor of Electrical and Computer Engineering

ABSTRACT

Video cameras are commonly used today in surveillance and security, autonomous driving and flying, manufacturing and healthcare. While different applications seek different types of information from the video streams, detecting changes and finding people are two key enablers for many of them. This dissertation focuses on both of these tasks: change detection, also known as background subtraction, and people detection from overhead fisheye cameras, an emerging research topic.

Background subtraction has been thoroughly researched to date and the top-performing algorithms are data-driven and supervised. Crucially, during training these algorithms rely on the availability of some *annotated* frames from the video being *tested*. Instead, we propose a novel, supervised background-subtraction algorithm for *unseen videos* based on a fully-convolutional neural network. The input to our network consists of the current frame and two background frames captured at different time scales along with their semantic segmentation maps. In order to reduce the chance of overfitting, we introduce novel temporal and spatio-temporal data-augmentation methods. We also propose a cross-validation training/evaluation strategy for the largest change-detection dataset, CDNet-2014, that allows a fair and video-agnostic performance comparison of supervised algorithms. Overall, our algorithm achieves significant performance gains over state of the art in terms of F-measure, recall and precision. Furthermore, we develop a real-time variant of our algorithm with performance close to that of the state of the art.

Owing to their large field of view, *fisheye* cameras mounted overhead are becoming a surveillance modality of choice for large indoor spaces. However, due to their top-down viewpoint and unique optics, standing people appear radially oriented and radially distorted in fisheye images. Therefore, traditional people detection, tracking and recognition algorithms developed for standard cameras do not perform well on fisheye images. To address this, we introduce several novel people-detection algorithms for overhead fisheye cameras. Our first two algorithms address the issue of radial body orientation by applying a rotating-window approach. This approach leverages a state-of-the-art object-detection algorithm trained on standard images and applies additional pre- and post-processing to detect radially-oriented people. Our third algorithm addresses both the radial body orientation and distortion by applying an end-to-end neural network with a novel angle-aware loss function and training on fisheye images. This algorithm outperforms the first two approaches and is two orders of magnitude faster. Finally, we introduce three spatio-temporal extensions of the end-to-end approach to deal with intermittent misses and false detections. In order to evaluate the performance of our algorithms, we collected, annotated and made publicly available four datasets composed of overhead fisheye videos. We provide a detailed analysis of our algorithms on these datasets and show that they significantly outperform the current state of the art.

Contents

1 Introduction **1**

1.1 Background Subtraction 2

1.2 People Detection From Overhead Fisheye Cameras 7

1.3 Organization 12

2 Related Work **14**

2.1 Background Subtraction 14

2.1.1 Unsupervised BGS Algorithms 14

2.1.2 Video- or Video-Group-Optimized Supervised BGS Algorithms 16

2.1.3 Video-Agnostic Supervised BGS Algorithms 17

2.1.4 Post-Processing Methods 18

2.2 People Detection Using Fisheye Cameras 20

2.2.1 People Detection in Images from Side-View, Standard-Lens Cameras 20

2.2.2 Object Detection Using Rotated Bounding Boxes 21

2.2.3 People Detection in Images from Overhead, Fisheye Cameras . 22

2.2.4 Video Object Detection 23

3 Background Subtraction for Unseen Videos Using Deep Learning **24**

3.1 Spatio-Temporal Inputs at Multiple Time Scales 25

3.2 Leveraging Semantic Segmentation 26

3.3 Notation 28

3.4 Improving Resilience to Illumination Changes by Temporal Data Augmentation 30
3.5 Network Architecture 32
3.6 Loss Function 32
3.7 Video-Agnostic Evaluation Strategy for Supervised Algorithms 33
3.8 Experimental Results 38
3.8.1 Dataset and Evaluation Metrics 38
3.8.2 Training and Evaluation Details 38
3.8.3 Quantitative Results 40
3.8.4 Visual Results 43
3.8.5 Ablation Study 45
3.9 Discussion 47
4 Spatio-Temporal Data Augmentations for Supervised Background Subtraction 49
4.1 Spatio-Temporal Data Augmentations 50
4.1.1 Notation 50
4.1.2 Spatio-Temporal Crops 52
4.1.3 Post-Crop Augmentations 54
4.1.4 Combining Spatio-Temporal Augmentations 56
4.2 Video-Agnostic Evaluation Strategy for Supervised Algorithms 57
4.3 Experimental Results 57
4.3.1 Dataset and Evaluation Details 57
4.3.2 Training Details 60
4.3.3 Ablation Study 63
4.3.4 Fast BSUV-Net 2.0 64
4.3.5 Comparison with State of the Art 67

4.3.6 Cross-Dataset Evaluation 70
4.3.7 In-the-Wild Results 72
4.4 Discussion 76

5 Datasets for People Detection from Fisheye Cameras 77

5.1 HABBOF 78
5.2 MW-R 81
5.3 CEPDOF 84
5.4 WEPDOF 84
5.5 Evaluation Methodologies 92
5.5.1 Evaluation metrics for Staged-Scenario Datasets 92
5.5.2 Evaluation Metric for In-the-Wild Dataset 92

6 People Detection from Overhead Fisheye Cameras Using Rotating Windows 95

6.1 Activity-Blind Application of YOLO v3 97
6.1.1 Focus Window and Image Rotation 98
6.1.2 Initial People Detection and Post-Processing 99
6.1.3 Reverse Mapping of Detections 100
6.1.4 NMS, Verification, and Final People Detection 100
6.2 Activity-Aware Application of YOLO v3 102
6.2.1 Background Subtraction 102
6.2.2 Background Model 104
6.2.3 Focus Window Selection 106
6.3 Experimental Results 107
6.4 Discussion 111

7 End-to-End People Detection from Overhead Fisheye Cameras 112

7.1 Notation 114

7.2 Network Architecture 114
7.2.1 Backbone 116
7.2.2 Feature Pyramid Network (FPN) 116
7.2.3 Detection Head 116
7.3 Angle-Aware Loss Function 117
7.3.1 Periodic Loss for Angle Prediction 118
7.3.2 Parameterization of Rotated Bounding Boxes 119
7.4 Inference 121
7.5 Experimental Results 121
7.5.1 Design Evaluation 124
7.5.2 Performance of RAPiD on Real-World Challenges 128
7.6 Discussion 131

8 Leveraging Temporal Information for People Detection From Overhead Fisheye Cameras 132
8.1 RAPiD+REPP 133
8.1.1 Construction of Bounding-Box Tubelets 133
8.1.2 Refinement Network 136
8.2 RAPiD+FGFA 136
8.3 Experimental Results 137
8.4 Discussion 143

9 Conclusions and Future Directions 144
9.1 Future Directions 146
9.1.1 Background Subtraction 146
9.1.2 People Detection from Overhead Fisheye Cameras 149

References 152

Curriculum Vitae 162

List of Tables

2.1 Training and evaluation methods of supervised BGS algorithms tested on CDNet-2014. 19

3.1 Training/testing splits used for algorithm evaluation. (✓ – video is used in training, ✗ – video is used in testing). 35
3.2 Performance comparison of BSUV-Net against state-of-the-art methods for **unseen videos** on CDNet-2014. For fairness, we separated the post-processing and self-contained algorithms. 41
3.3 Performance comparison of BSUV-Net against state-of-the-art methods according to per-category *F-score* for **unseen videos** on CDNet-2014. 42
3.4 Impact of background frames, data augmentation to combat illumination changes and semantic information (FPM) on BSUV-Net performance. 46
3.5 Comparison of BSUV-Net with thresholded FPM used as a BGS result (probability threshold equals 0.5). 47
3.6 Comparison of manual and automatic selection of empty background frames in BSUV-Net. 47

4.1 Proposed sets for 4-fold cross-validation on CDNet-2014. 58

4.2 Comparison of different spatio-temporal augmentations on CDNet-2014 based on *F-score*. SAC: spatialy-aligned crop, RSC: randomly-shifted crop, PTZ: PTZ camera crop, ID: illumination difference, IOA: intermittent object addition. The values in boldface font show the best performance for each category. 61
4.3 Efficiency vs performance trade-off for BSUV-Net 2.0 on CDNet-2014. FPS is calculated using PyTorch 1.3 implementation on a node with single Nvidia Tesla P100 GPU. 64
4.4 Official comparison of top BGS algorithms evaluated on **unseen videos** from CDNet-2014. 65
4.5 Official comparison of top BGS algorithms according to the per-category *F-score* on **unseen videos** from CDNet-2014. 66
4.6 *F-score* comparison of BSUV-Net 2.0 with *video-agnostic* supervised BGS algorithms that are not reported in `changedetection.net`. Each column shows test performance of the algorithm by using the training/testing split provided in the respective paper. 68
4.7 Per-category *F-score* comparison of the cross-dataset performance of BSUV-Net 2.0 with the top-performing unsupervised BGS algorithms on LASIESTA 71
4.8 Per-video *F-score* comparison of BSUV-Net 2.0 with the top-performing supervised BGS algorithms on unseen videos from LASIESTA. BSUV-Net 2.0 is trained only on CDNet-2014, as explained in Section 4.3.6. BSUV-Net 2.0* is trained on videos from LASIESTA that are not from the test set as described in [Mandal et al., 2021]. 73

4.9 *F-score* comparison of cross-dataset performance of different spatio-temporal augmentations on *Moving camera* and *Simulated motion* videos of LASIESTA. SAC: spatialy-aligned crop, RSC: randomly-shifted crop, PTZ: PTZ camera crop. 74

5.1 Statistics of 4 new datasets for people detection from overhead fisheye cameras. "people" refers to the annotated people and "# of people" refers to the number of annotated people in a single frame. 78
5.2 Scenario details and statistics of individual videos in HABBOF. . . . 79
5.3 Description and statistics of individual videos in MW-R. 82
5.4 Scenario details and statistics of individual videos in CEPDOF. . . . 85
5.5 Scenario details and statistics of individual videos in WEPDOF. . . . 88
5.6 The number of bounding boxes assigned to small, medium and large class (see text for the definition) for all WEPDOF videos. 94
5.7 Proposed sets for 2-fold cross-validation on WEPDOF 94

6.1 Performance comparison of people detection methods on MW-R, HABBOF and CEPDOF datasets. P, R and F denote *Precision*, *Recall*, and *F-measure*, respectively. 107
6.2 In-the-wild comparison of people-detection algorithms on WEPDOF. Average run-times are computed on a node with a single NVIDIA Tesla V100 GPU. 108

7.1 Performance comparison of people-detection algorithms on three *staged* test datasets (MW-R, HABBOF, CEPDOF). P, R, and F denote *Precision*, *Recall*, and *F-measure*, respectively. 122

7.2 *In-the-wild* comparison of people-detection algorithms on WEPDOF. The average runtimes are computed on a node with a single NVIDIA Tesla V100 GPU. 123
7.3 Ablation study of RAPiD. The first row corresponds to the baseline algorithm. Fine-tuning is applied using the MW-R dataset. 127
7.4 Comparison of RAPiD's performance for different angle ranges and loss functions. 128
7.5 Per-video performance comparison of fisheye people-detection algorithms on WEPDOF. 130

8.1 *In-the-wild* comparison of people-detection algorithms on WEPDOF. The average run-times are computed on a node with a single NVIDIA Tesla V100 GPU. 138
8.2 Per-video performance comparison of fisheye people-detection algorithms on WEPDOF. 142

List of Figures

1·1 Scene-optimized and video-agnostic training/testing regimes illustrated using 4 videos from CDNet-2014 [Goyette et al., 2014]. 5

1·2 (a) Sample frame from Penn-Fudan database for pedestrian detection and segmentation [Wang et al., 2007] with image-axis-aligned bounding boxes. (b) Sample frame from one of our own datasets for people detection from overhead fisheye cameras (CEPDOF) with human-body-aligned bounding boxes. 9

3·1 Examples of spatio-temporal inputs to BSUV-Net for three videos from CDNet-2014 27

3·2 Examples of semantic segmentation and foreground probability map (FPM) for the current frames from Figure 3·1. Different colors in (b) represent 150 object categories of ADE20K [Zhou et al., 2017a] and the color coding in (c) is from 0 (black) to 1 (white). 29

3·3 Network architecture of BSUV-Net. BN stands for batch normalization and SD stands for spatial dropout. Grayscale images at the network input show foreground probability maps (FPM) of the corresponding RGB frames. 31

3·4 Visual comparison of sample results produced by BSUV-Net, SWCD and WisenetMD on **unseen videos** from CDNet-2014. 44

4·1 Image augmentation examples. Each row shows an example for one of the augmentations: (a) original input, (b) spatially-aligned crop, (c) randomly-shifted crop, (d) PTZ camera crop, (e) illumination difference, (f) intermittent-object addition. 51
4·2 Visual comparison of different spatio-temporal augmentations on sample frames. Columns 1 and 2 show the current frame and its ground truth. The subsequent columns show predictions of our network trained with augmentations listed at the top of each column. The last column shows predictions of BSUV-Net 2.0 trained using a combination of all proposed augmentations. The dark-gray areas represent pixels outside of CDNet-2014 regions of interest. 62
4·3 Qualitative comparison of top BGS algorithms on sample frames from different categories of CDNet-2014. 69
4·4 Sample results of in-the-wild application of BSUV-Net 2.0. The videos have been captured by a live camera during the day and at night to test different illumination conditions. Shown are the input frame (left column), foreground predictions computed by BSUV-Net 2.0 (middle column) and a combination of the input frame and foreground predictions (right column) – darker pixels represent the background while brighter ones show the foreground. 75

5·1 Sample frames with rotated bounding-box annotations from all 4 videos in HABBOF. 80
5·2 Sample frames with rotated bounding-box annotations from MW-R. . 83
5·3 Sample frames with rotated bounding-box annotations from all 8 videos in CEPDOF. 86

5·4 Sample frames with rotated bounding-box annotations from all 14 videos in WEPDOF. 90
5·5 (a) Sample frame from WEPDOF's "IT Office", (b) its ROI map, (c) its ROI map overlayed on top of the frame. 91
5·6 Histogram of bounding-box areas of the annotations in WEPDOF. All areas are normalized to image size of 1024 × 1024. 93

6·1 Examples of overhead fisheye images (top row) and their dewarped versions (last 3 rows). 96
6·2 Block diagram of the proposed activity-blind (AB) people detection method. 98
6·3 (a) Placement of the 1,300×800-pixel focus window in a 2,048×2,048 image. The faint green area is the margin area defined in Section 6.1.2. (b) Focus window after reverse rotations. 99
6·4 Comparison of the two reverse mapping approaches: (a) naïve approach here reverse-rotated bounding boxes are not radially oriented, and (b) improved approach where reverse-rotated bounding boxes are radially oriented and are better aligned with human bodies. 101
6·5 Block diagram of the proposed AA algorithm. Blocks within the blue-shaded rectangle are common to both AA and AB methods, while the other parts are unique to the AA algorithm. 103
6·6 Examples of: (a) reference background; (b) current frame; (c) focus windows for the current frame overlaid on the RGB image; and (d) the same focus windows overlaid on the connected components of the final RoI. 105

6·7 Qualitative comparison of people-detection results on sample frames from each video (one per column) in the HABBOF dataset. Columns show the ground truth annotations as well as the AB and AA people-detection results. 109

7·1 Illustration of *typical* people-detection results on overhead, fisheye images (one quarter shown) for algorithms using various bounding-box orientation constraints; the human-aligned bounding boxes fit bodies most accurately. These are not outputs from any algorithms. See the text for discussion. 113

7·2 RAPiD architecture. Following the paradigm of one-stage detectors, our model contains a backbone, FPN, and detection head (bounding-box regression network). In the diagram, each arrow represents multiple convolutional layers and the colored rectangles represent multi-dimensional matrices, i.e., feature maps, whose dimensions correspond to input image of size $h \times w = 1024 \times 1024$. 115

7·3 Periodic loss function with $L2$ norm as regressor and its derivative . . 119

7·4 Illustration of the necessity to expand the predicted-angle value range. Gradient descent applied to the predicted angle $\hat{\theta}$ (red arrow) may rotate it clockwise and away from the ground truth angle θ (green arrow). Since a bounding box at angle $\theta + \pi$ is the same as the one at θ, we need to extend the angle range to include $\theta + \pi$ (dashed green arrow) otherwise $\widehat{\theta}$, pushed by the gradient, will stop at $\pi/2$. 120

7·5 Comparison of AA, AB and RAPiD on WEPDOF in terms of: (a) *F-score* versus confidence-score threshold; and (b) *Precision-Recall* curves.125

7·6 Qualitative results of RAPiD on videos from MW-R (a, b), CEPDOF (c–f) and WEPDOF (g, h). Green boxes are true positives, red boxes are false positives, and yellow boxes are false negatives. 126

7·7 Histogram of the height-to-width ratio of the predicted bounding boxes. 129

8·1 An example of greedy bounding-box matching. Frame i has 3 bounding boxes and frame $i+1$ has 2. Similarity scores between the rotated bounding boxes of frame i and frame $i+1$ are given in the matrix on the left. The greedy algorithm will match bb_i^2 with bb_{i+1}^1 since they have the largest similarity score. Then, the first column and the second row will be eliminated and the similarity matrix will be reduced to a 2×1 matrix as shown on the right. Finally, bb_i^3 will be matched with bb_{i+1}^2 and the matching algorithm will terminate since there are no more boxes left in frame $i+1$. 135

8·2 Comparison of people-detection algorithms on WEPDOF in terms of: (a) *F-score* versus confidence-score thresholds; and (b) *Precision* versus *Recall*. Tamura *et al.* [Tamura et al., 2019] is omitted since its performance is significantly lower. 139

8·3 Qualitative results of RAPiD and its spatio-temporal extensions on videos from WEPDOF. Green boxes are true positives, red boxes are false positives, and yellow boxes are false negatives. 141

List of Abbreviations

BGS	....	Background Subtraction
BSUV-Net	....	Background Subtraction for Unseen Videos
CEPDOF	....	Challenging Events for Person Detection from Overhead Fisheye Cameras
CNN	....	Convolutional Neural Network
FGFA	....	Flow-Guided Feature Aggregation
FOV	....	Field of View
HABBOF	....	Human-Aligned Bounding Boxes from Overhead Fisheye Cameras
IOU	....	Intersection over Union
MW	....	Mirror Worlds Dataset
MW-R	....	Mirror Worlds-Rotated Dataset
OHF	....	Overhead Fisheye
RAPiD	....	Rotation-Aware People Detection
REPP	....	Robust and Efficient Post-Processing for Video Object Detection
ROI	....	Region of Interest
SVS	....	Side-View Standard-Lens
WEPDOF	....	In-the-Wild Events for People Detection from Overhead Fisheye Cameras
YOLO	....	You Only Look Once

Chapter 1

Introduction

While surveillance and security cameras have been prevalent for decades, with the latest advances in big data and machine learning their usefulness has grown tremendously leading to diverse applications ranging from public safety to animal-behavior analysis. In most of such applications, the data are recorded continuously for long periods of time, even 24/7 in many cases, and making sense of these vast visual data is a huge challenge. One of the commonly-used mechanisms to filter out unnecessary information is to detect changes in video data. This is usually accomplished by the so-called *background subtraction* (BGS). In BGS, the aim is to separate foreground areas of a video frame from the background. Since most of the critical information is associated with the foreground (e.g., people, cars), BGS turns out to be a very useful pre-processing tool for many applications. For example, in surveillance and security, one may be interested in detecting suspicious activities (e.g., a person entering a restricted area); a detected foreground is useful for this task. Similarly, in autonomous driving, it is critical to detect the nearby pedestrians and cars; both are usually appear in the foreground. A less known application of BGS is animal-behavior analysis where several cameras help monitor animal activity patterns by detecting changes between video frames.

Another critical video-analysis task is the detection of people. It is often the very first step in recognition, counting and tracking of people and their actions. In outdoor environments, the detection and tracking of pedestrians is critical for autonomous driving; a successful tracking algorithm can locate pedestrians and even predict their near-future locations thus helping avoid accidents. In indoor environments, counting people throughout a building is essential for next-generation heating, ventilation, and air conditioning (HVAC), safety and security as well as space management. Today, HVAC systems operate in a binary fashion – they provide a minimum air flow when a room is empty and a maximum air flow even if a single person enters. Obviously, this results in huge energy waste. This waste can be significantly reduced by automatically counting people in a room and controlling the HVAC system as a function of the occupancy level. Knowing how many people are in a building and where is critical for emergency situations, such as fire, chemical hazard, active-shooter scenario, etc. Finally, a long-term analysis of occupancy patterns in a building can help optimize space usage and reduce rental costs.

In this dissertation, we focus on both problems – background subtraction and people detection – using supervised algorithms and provide solutions suitable for unseen videos with various real-life challenges.

1.1 Background Subtraction

Background subtraction aims to segment an input video frame into regions corresponding to either foreground (e.g., motor vehicles) or background (e.g., highway surface). It is frequently used as a pre-processing step for higher-level tasks such as object tracking, people and motor-vehicle recognition, human activity recognition, etc. Since BGS is often the first pre-processing step, the accuracy of its output has an overwhelming impact on the overall performance of subsequent steps. Therefore, it is critical that BGS produce as accurate a foreground/background segmentation as possible.

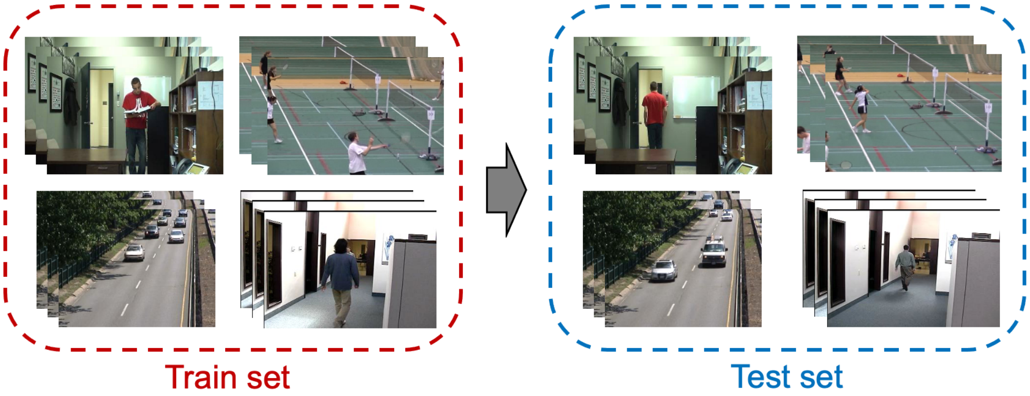
Traditional BGS algorithms are unsupervised and rely on a background model to predict foreground regions [Stauffer and Grimson, 1999, Elgammal et al., 2002, Zivkovic, 2004, Mittal and Paragios, 2004, Barnich and Van Droogenbroeck, 2011, St-Charles et al., 2015a, St-Charles et al., 2015b, Işık et al., 2018, Lee et al., 2018]. Pixel-based Adaptive Word Consensus Segmenter (PAWCS) [St-Charles et al., 2015a], Sliding Window-based Change Detection (SWCD) [Işık et al., 2018] and WisenetMD [Lee et al., 2018] are currently considered to be state-of-the-art *unsupervised* BGS algorithms. However, since they rely on the accuracy of the background model, they encounter difficulties when applied to complex scenes. Ensemble methods combine the results produced by several BGS algorithms by means of genetic programming [Bianco et al., 2017] or convolutional neural networks (CNNs) [Zeng et al., 2019b] and have been shown to significantly outperform traditional algorithms.

The success of deep learning in computer vision did not bypass BGS research [Bouwmans et al., 2019]. A number of *supervised* deep-learning BGS algorithms have been developed [Braham and Van Droogenbroeck, 2016, Wang et al., 2017, Sakkos et al., 2018, Babaee et al., 2018, Bakkay et al., 2018, Zeng and Zhu, 2018, Lim and Keles, 2018a, Lim and Keles, 2018b] with performance easily surpassing that of the traditional methods. However, these algorithms have been tuned to either one specific video or to a group of videos similar to the test video, and their performance drops significantly when applied to *unseen videos*. Clearly, they are not suitable for real-world applications.

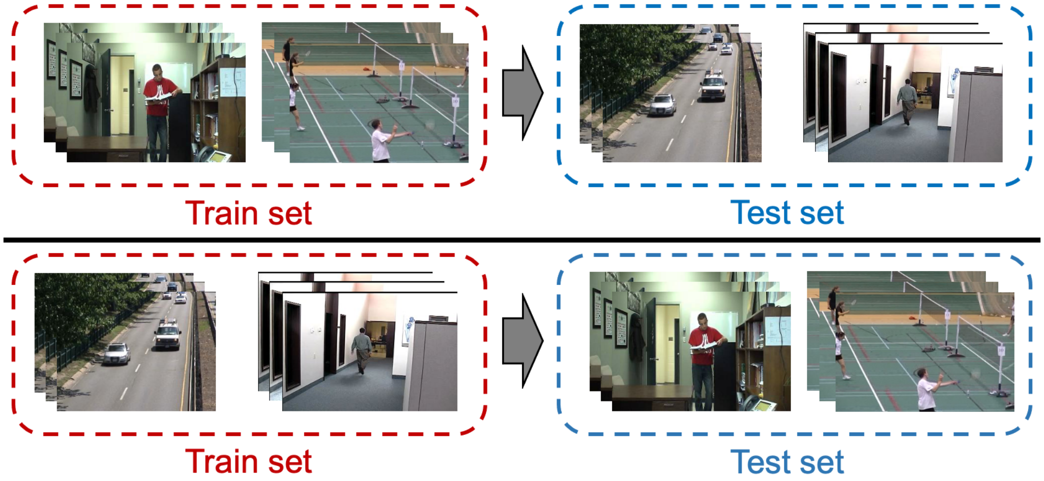
To address this problem, we introduce *Background Subtraction for Unseen Videos* (BSUV-Net). BSUV-Net is a *video-agnostic* supervised BGS algorithm that can be applied to unseen videos with no or little loss of performance. A key feature of BSUV-Net is that the training and test sets are composed of frames originating from different videos. This guarantees that no ground-truth data from the test videos have been shown to the network in the training phase. Figure 1·1 depicts the training/testing regimes used in *video-* or *video-group-optimized* algorithms versus *video-agnostic* algorithms for 4 videos from CDNet-2014 [Goyette et al., 2014]. Clearly, the training and test sets used by *video-* or *video-group-optimized* algorithms share lots of similarities (e.g., very similar background and foreground objects in Figure 1·1a) that the network can memorize. However, memorization can reduce the performance of these algorithms significantly on *unseen* videos. On the other hand *video-agnostic* algorithms use completely different videos in their training and test sets (see Figure 1·1b) which forces them to generalize better to *unseen* videos.

Another key feature of BSUV-Net is the composition of network input. By employing two reference backgrounds at different time scales, BSUV-Net addresses two challenges often encountered in BGS: (i) varying scene illumination and (ii) intermittently-static objects that tend to get absorbed into the background. We also propose a novel *temporal* data augmentation strategy that further improves the method's performance under varying illumination. Furthermore, motivated by the work of Braham *et al.* on the use of semantic segmentation in BGS [Braham et al., 2017], we improve our method's accuracy by complementing the reference backgrounds and the current frame on input with a semantic segmentation for each of them.

One of the most successful approaches for increasing the generalization capacity of computer vision algorithms trained with limited data is the use of data augmentation. Spatial data augmentations, such as random crops, rotations, color changes, noise etc. have proved very successful in image-related tasks [Taylor and Nitschke, 2017,Shorten and Khoshgoftaar, 2019]. The simple *temporal* data augmentation mechanism, that we introduced in BSUV-Net to handle illumination differences between videos, resulted in a significant performance improvement. Motivated by this, we propose a comprehensive suite of *spatio-temporal* data augmentation methods and adapt them to BSUV-Net. The proposed augmentations address some key BGS challenges, such



(a) *Scene-optimized* (also referred as *video-* or *video-group-optimized*) training



(b) *Video-agnostic* training via cross-validation

Figure 1·1: Scene-optimized and video-agnostic training/testing regimes illustrated using 4 videos from CDNet-2014 [Goyette et al., 2014].

as pan-tilt-zoom (PTZ) operation, camera jitter and presence of intermittently-static objects. We conduct a *video-agnostic* performance analysis of the new BSUV-Net 2.0 and show that these data augmentations significantly increase algorithm's performance for targeted categories without any significant loss of performance in other categories. In extensive experiments on the CDNet-2014 dataset [Goyette et al., 2014], we show that both BSUV-Net and the improved BSUV-Net 2.0 outperform state-of-the-art BGS algorithms evaluated on *unseen* videos.

Furthermore, to demonstrate the versatility of the proposed methodology we apply our best-performing BGS algorithm, BSUV-Net 2.0, to several video clips recorded at different times of the day by a live high-resolution surveillance camera. The captured scene is of a very busy street intersection in Tokyo, Japan, with a multitude of people and motor vehicles either in motion or intermittently stopped. Since no ground-truth annotations are available for these clips, we only provide visual results demonstrating that BSUV-Net 2.0 is a very promising BGS algorithm suitable for real-life applications.

The main contributions of this part of the dissertation can be summarized as follows:

1. **Supervised BGS algorithms for unseen videos:** Although supervised algorithms, especially neural networks, have significantly improved BGS performance, most of them are tuned to a specific video and thus their performance on *unseen* videos deteriorates dramatically. We introduce two BGS algorithms that are both truly generalizable to *unseen* videos.

2. **Leveraging multiple-time-scale and semantic information:** The proposed BGS algorithms improve foreground-background segmentation accuracy by using the current frame (to be segmented) and two background frames from different time scales. While one background frame, based on *distant* history,

helps with the discovery of intermittently-static objects, the other frame, based on *recent* history, is key for handling dynamic factors such as illumination changes. Each of these inputs is complemented by its semantic segmentation that helps shape the foreground regions as typical objects. This is unlike in an earlier BGS method [Braham et al., 2017] which used semantic information as a *post-processing* step.

3. **Fair evaluation strategy for CDNet-2014:** Although CDNet-2014 is an extensive BGS dataset, it lacks a training/testing split for consistent evaluation of supervised learning approaches. We introduce a split of CDNet-2014 videos into 4 groups to be used for cross-validation. In this way, one can easily evaluate any supervised BGS algorithm on all CDNet-2014 videos in a video-agnostic manner. This will simplify algorithm performance comparisons in the future.

4. **Spatio-temporal data augmentation:** We introduce spatio-temporal data augmentation methods for BSUV-Net to mimic challenging BGS scenarios, such as PTZ operation, camera jitter, illumination variations and presence of intermittently-static objects (e.g., cars stopped at a streetlight). Our experimental results show that these augmentations significantly improve the performance on unseen videos of corresponding categories.

1.2 People Detection From Overhead Fisheye Cameras

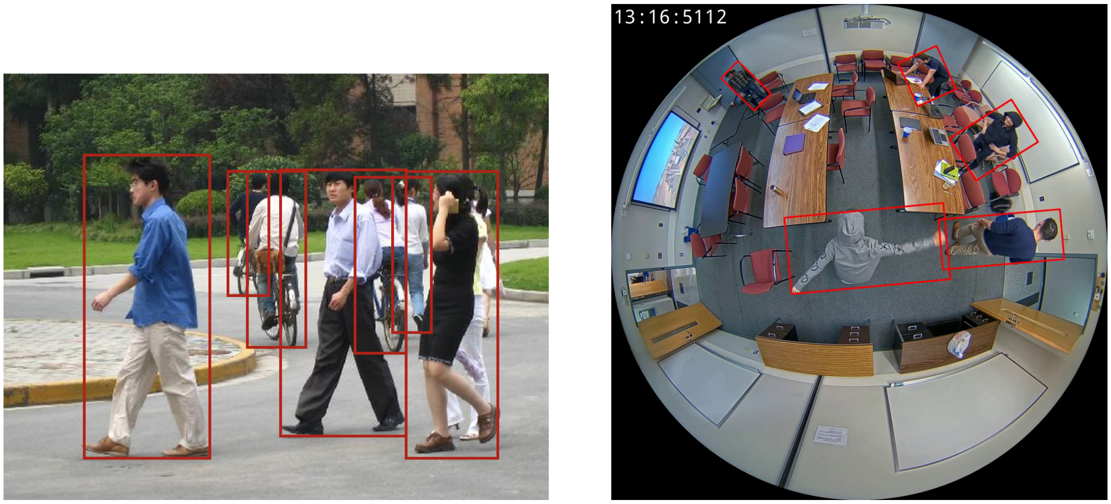
People detection is a key first step in many video-analysis tasks, such as people counting and tracking, action recognition, etc. To date, most of the research related to people detection has been focused on side-view standard-lens (SVS) images [Enzweiler and Gavrila, 2009, Nguyen et al., 2016, Brunetti et al., 2018]. Typically, a wide-angle, standard-lens camera is side-mounted above the scene. The main challenges in using

SVS cameras are: significant occlusions, potential blind spots and the need to use multiple cameras to fully cover a large space.

In the last decade, interest has grown in detecting and tracking people from overhead fisheye (OHF) images and videos [Saito et al., 2011, Chiang and Wang, 2014, Wang et al., 2017, Krams and Kiryati, 2017, Tamura et al., 2019]. As shown in Figure 1·2, a single high-resolution, overhead, fisheye camera with a 360° horizontal field of view (FOV) can monitor a much larger space than a single SVS camera and, owing to its viewpoint, captures images with vastly reduced occlusions. However, people detection algorithms developed for side-view, standard-lens images do not perform well on overhead, fisheye images due to their unique radial geometry and barrel distortions. In this thesis, we propose three different solutions to these challenges:

- a rotating-window approach based on a state-of-the-art deep-learning object-detection algorithm developed for SVS images and augmented by novel pre- and post-processing,
- a new end-to-end deep-learning approach designed specifically to tackle the radial-geometry challenge,
- an extension of the end-to-end approach to videos by leveraging the temporal information alongside the spatial information.

Due to the scarcity of datasets composed of OHF images or videos, we needed to collect and annotate our own datasets to evaluate our methods. We recorded 12 videos using overhead fisheye cameras mounted in 4 different rooms on Boston University campus. With the help from our research-group members and several undergraduate students, we annotated these videos by drawing bounding boxes aligned with the body of each person (Figure 1·2b), unlike in other datasets where bounding boxes



(a) **(b)**

Figure 1·2: (a) Sample frame from Penn-Fudan database for pedestrian detection and segmentation [Wang et al., 2007] with image-axis-aligned bounding boxes. (b) Sample frame from one of our own datasets for people detection from overhead fisheye cameras (CEPDOF) with human-body-aligned bounding boxes.

are aligned with image axes [Demiröz et al., 2012, del Blanco et al., 2021, Ma et al., 2018b]. In addition to our own videos, we also annotated a subset of the videos collected for the Mirror Worlds Challenge [Ma et al., 2018b] and 16 OHF video clips that we collected from YouTube. In total, we annotated 47 videos with more than 50,000 frames.

In SVS images, standing people usually appear in an upright position and algorithms that detect bounding boxes aligned with image axes, such as You Only Look Once (YOLO) [Redmon et al., 2016, Redmon and Farhadi, 2017, Redmon and Farhadi, 2018], Single Shot MultiBox Detector (SSD) [Liu et al., 2016] and Regions with CNN Features (R-CNN) [Ren et al., 2015], work well. However, these algorithms perform poorly on OHF images, usually missing non-upright bodies. In such images, standing people appear along image radius (Figure 1·2b), due to the overhead placement of the camera, so bounding-box rotations must be allowed. To accommodate such rotations, we introduce a rotating-window approach that leverages a state-of-the-art object detection algorithm. Among the object-detection algorithms for SVS images, YOLO v3 [Redmon and Farhadi, 2018] achieves a very competitive performance in real time (using a desktop GPU). Therefore, we propose two methods to leverage YOLO v3 for people detection from OHF images. In one approach, we apply YOLO v3 only to a window extracted from the upper central part of a fisheye image where the orientation of people should be close to upright. To cover the whole image, we create 24 rotations of the image and apply YOLO v3 to the same window after each rotation. Then, we rotate the results back to the original angles and apply post-processing to prune multiple detections of the same person (the results from neighboring rotations may overlap). In an alternative approach, we first identify regions of interest (ROIs) where activity takes place and then we rotate each ROI to the upper central part of the image and apply YOLO v3. In order to identify areas of activity, we apply a simple background subtraction algorithm. Experimental results demonstrate that both rotating-window approaches significantly improve the people detection performance over state-of-the-art.

However, the proposed algorithms' inference speed is sub-par due to the multiple applications of YOLO v3 to a single image. Therefore, we introduce Rotation-Aware People Detection (RAPiD), an end-to-end supervised algorithm for people detection from OHF images. RAPiD is a single-stage convolutional neural network that predicts arbitrarily-rotated bounding boxes around people in a fisheye image. It extends the model proposed in YOLO v3 [Redmon and Farhadi, 2018]. In addition to predicting the center and size of a bounding box, RAPiD also predicts its angle. This is accomplished by a periodic loss function based on an extension of a common regression loss. This allows us to predict the exact rotation of each bounding box in an image without any assumptions and additional computational complexity. Since

RAPiD is an end-to-end algorithm, we can train or fine-tune its weights on annotated fisheye images. Indeed, we show that such fine-tuning of a model trained on standard images significantly increases the method's performance. An additional aspect of this work, motivated by its focus on people detection, is the replacement of the common regression-based loss function used in multi-class object detection algorithms [Girshick, 2015, Ren et al., 2015, Redmon et al., 2016, Liu et al., 2016] with single-class object detection. The inference speed of RAPiD is nearly identical to that of YOLO v3 since it is applied to each image only once without the need for pre- or post-processing.

Both the rotating-window approach and RAPiD perform people detection independently in each frame. However, if a camera's acquisition frame rate is sufficiently high, the locations of people in consecutive frames do not change much. This temporal coherence has been well-known and used in video processing and computer vision for decades. Recently, temporal information has been successfully leveraged in deep-learning methods for video-object detection [Zhu et al., 2017, Zhang et al., 2018, Wu et al., 2019, Lin et al., 2019, Liu et al., 2019, Han et al., 2020, Sabater et al., 2020, Chen et al., 2020]. Inspired by these methods, we introduce 3 extensions of RAPiD that combine temporal information with spatial information to improve people-detection performance. Our approach combines RAPiD, with some of the best-performing video-object detection algorithms, Robust and Efficient Post-Processing for video object detection (REPP) [Sabater et al., 2020] and Flow-Guided Feature Aggregation (FGFA) [Zhu et al., 2017]. We show that the improved versions of RAPiD achieve significantly better performance on our most challenging dataset collected from YouTube. However, due to additional post-processing steps and increased complexity of the network, the inference speed of the improved versions is reduced compared to that of RAPiD.

The main contributions of this part of the dissertation can be summarized as follows:

1. **Extensive datasets for training and evaluation:** We introduce 3 *staged* datasets and 1 *in-the-wild* dataset for people detection from overhead fisheye cameras. All off our datasets are labeled with rotated-bounding boxes and they include various challenging scenarios.

2. **Algorithms for people detection from overhead fisheye cameras:** We develop 6 algorithms that significantly outperform the state of the art. We discuss the trade-offs of these algorithms in detail using multiple evaluation metrics and performance trade-off plots.

3. **Angle-aware loss function and end-to-end algorithm:** We propose a continuous, periodic loss function for bounding-box angle that, unlike in previous methods, facilitates arbitrarily-oriented bounding boxes capable of handling a wide range of human-body poses. Using this loss function, we introduce a new end-to-end people detection algorithm from overhead fisheye cameras that outperforms the state-of-the-art algorithms.

4. **Leveraging spatio-temporal information:** We introduce extensions to our end-to-end algorithm by leveraging spatial *and* temporal information simultaneously. The proposed extensions outperform the spatial-only version significantly.

1.3 Organization

This dissertation is organized as follows. In Chapter 2, we discuss the related work on background subtraction and people detection from overhead fisheye cameras. In Chapters 3 and 4, we introduce our supervised background subtraction algorithms designed for unseen videos. We show the effectiveness and generalizability of our proposed models by extensive quantitative and qualitative analysis. In Chapter 5, we introduce four people-detection datasets from overhead fisheye cameras that we annotated with the help of our lab members and undergraduate students. In Chapters 6-8, we introduce novel people-detection algorithms that we developed for OHF images and videos via different design paradigms. We show the effectiveness of the introduced algorithms on the datasets introduced in Chapter 5. Chapter 9 summarizes the contributions of this dissertation, draws conclusions and offers new directions to explore in the future.

Chapter 2

Related Work

2.1 Background Subtraction

A wide range of BGS algorithms have been developed in the past, each having some advantages and disadvantages over others. In this section, we divide these algorithms into 4 categories: (i) unsupervised BGS algorithms, (ii) video- or video-group-optimized supervised BGS algorithms (iii) video-agnostic supervised BGS algorithms and (iv) post-processing methods that improve the results of BGS algorithms.

2.1.1 Unsupervised BGS Algorithms

Nearly all traditional BGS algorithms first compute a background model, and then use it to predict the foreground. While a simple model based on the mean or median of a subset of preceding frames offers only a single background value per pixel, a probabilistic Gaussian Mixture Model (GMM) [Stauffer and Grimson, 1999] allows a range of background values. This idea was improved by creating an online procedure for the update of GMM parameters in a pixel-wise manner [Zivkovic, 2004]. Kernel Density Estimation (KDE) was introduced into BGS [Elgammal et al., 2002] as a non-parametric alternative to GMMs and was subsequently improved [Mittal and Paragios, 2004]. The probabilistic methods achieve better performance compared to single-value models for dynamic scenes and scenes with small background changes.

Barnich and Droogenbroeck introduced a sample-based background model [Barnich and Van Droogenbroeck, 2011]. Instead of implementing a probabilistic model, they modeled the background by a set of sample values per pixel and used a distance-based model to decide whether a pixel should be classified as background or foreground. Since color information alone is not sufficient for complex cases, such as illumination changes, Bilodeau *et al.* introduced Local Binary Similarity Patterns (LBSP) to compare the current frame and background using spatio-temporal features instead of color [Bilodeau et al., 2013]. St-Charles *et al.* combined color and texture information, and introduced a word-based approach, PAWCS [St-Charles et al., 2015a]. They considered pixels as background words and updated each word's reliability by its persistence. Similarly, Self-Balanced SENsitivity SEgmenter (SuBSENSE) by St-Charles *et al.* [St-Charles et al., 2015b] combines LBSP and color features, and employs pixel-level feedback to improve the background model.

Recently, Isik *et al.* introduced SWCD, a pixel-wise, sliding-window approach leveraging a dynamic control system to update the background model [Işık et al., 2018], while Lee *et al.* introduced WisenetMD, a multi-step algorithm to eliminate false positives in dynamic backgrounds [Lee et al., 2018]. In another approach, Sultana *et al.* introduced an unsupervised background estimation method based on a generative adversarial network (GAN) [Sultana et al., 2019]. They used optical flow to create a motion mask and then in-painted the pixels with significant motion with background values estimated by a GAN. The foreground is then computed by subtracting the estimated background from the current frame followed by morphological operations. They, however, did not achieve state-of-the-art results. Zeng *et al.* introduced real-time semantic segmentation (RTSS) [Zeng et al., 2019a] which uses deep learning-based semantic segmentation predictions to improve the background model used in SubSENSE [St-Charles et al., 2015b].

2.1.2 Video- or Video-Group-Optimized Supervised BGS Algorithms

A supervised BGS algorithm estimates the foreground in two steps. First, it learns the parameters (e.g., neural-network weights) of a complex function through minimization of a loss function dependent on labeled training frames. Then, using the learned parameters, it is applied to a separate set of test frames to assess its performance. Several recently-developed algorithms use some frames from a test video for training and *all* frames of the *same* video for evaluating performance on that video. In such algorithms, parameter values are optimized separately for each video. We will refer to this class of algorithms as *video-optimized* BGS algorithms. In another family of algorithms, randomly-selected frames from a *group* of test videos are used for training and *all* the frames of the *same* videos are used for testing. Since some frames from *all* test videos are used for training, we will refer this class of algorithms as *video-group-optimized* algorithms. Note that, in both of these scenarios the algorithms are neither optimized for nor evaluated on *unseen* videos.

In recent years, supervised learning algorithms based on CNNs have been widely applied to BGS. The first CNN-based BGS algorithm was introduced in [Braham and Van Droogenbroeck, 2016]. This is a *video-optimized* algorithm which produces a single foreground probability for the center of each 27×27 patch of pixels. A method proposed in [Wang et al., 2017] uses a similar approach, but with a modified CNN which operates on patches of size 31×31 pixels.

Instead of using a patch-wise algorithm, Zeng and Zhu introduced the Multiscale Fully-Convolutional Neural Network (MFCN) which can predict the foreground of the entire input image frame in one step [Zeng and Zhu, 2018]. Lim and Keles proposed Foreground Segmentation Network (FgSegNet), a triplet CNN which uses Siamese networks to create features at three resolution scales and combines these features within a transposed CNN [Lim and Keles, 2018a]. In a follow-up work, they removed the triplet networks and used dilated convolutions to capture the multiscale information [Lim and Keles, 2018b]. Bakkay *et al.* used generative adversarial networks for BGS [Bakkay et al., 2018]. The generator performs the BGS task, whereas the discriminator tries to classify the BGS map as real or fake. Although all these algorithms perform very well on various BGS datasets, it is important to note that they are all *video-optimized*, thus they will suffer a performance loss when tested on unseen videos.

Babae *et al.* designed a *video-group-optimized* CNN for BGS [Babaee et al., 2018]. They randomly selected 5% of CDNet-2014 frames [Goyette et al., 2014] as a training set and developed a single network for all of the videos in this dataset. Sakkos *et al.* used a 3D CNN to capture the temporal information in addition to the color information [Sakkos et al., 2018]. Similarly to [Babaee et al., 2018], they trained a single algorithm using 70% of frames in CDNet-2014 and then used it to predict the foreground in all videos of the dataset. Note that even these approaches do not generalize to other videos since some ground-truth data from *each* video exists in the training set.

2.1.3 Video-Agnostic Supervised BGS Algorithms

In the last two years, several supervised background-subtraction algorithms have been developed with the goal of improving their performance on *unseen* videos. Such *video-agnostic* algorithms, use frames from a set of training videos to learn the network parameters, but a completely different set of videos – for evaluation. The idea of *video-agnostic* or *scene-independent* background subtraction was simultaneously, but independently, developed in 2019 by us and by Mandal *et al.* We introduced BSUV-Net in 2019 [Tezcan et al., 2019,Tezcan et al., 2020] and BSUV-Net 2.0 in 2021 [Tezcan et al., 2021b]. On the other hand Mandal *et al.* introduced 3D Feature Reductionist framework (3DFR) in 2019 [Mandal et al., 2019], as well as ChangeDet [Mandal and Vipparthi, 2020] and 3D-CNN based Change Detection network (3DCD) in 2020 [Mandal et al., 2021]. These are end-to-end convolutional neural networks for BGS that use both spatial and temporal information based on previous frames and a simple median-based background model.

Similarly, Kim *et al.* [Kim and Ha, 2020] introduced a U-Net-based [Ronneberger et al., 2015] neural network that uses a concatenation of the current frame and several background models generated at different time scales as the input.

During evaluation, the methods developed by Mandal *et al.* and by Kim *et al.* divide videos of a popular BGS dataset, CDNet-2014 [Goyette et al., 2014], into a training set and a testing set, and report results for the test videos, unseen by the algorithm during training. Although all of these algorithms outperform unsupervised algorithms on their own test sets, their true performance is unknown since no results were reported for the full dataset. Furthermore, these algorithms cannot be compared with each other since each used a different train/test split. Table 2.1 compares and summarizes the landscape of supervised BGS algorithms and the methodology used for training and evaluation.

A more detailed comparison of the *video-optimized*, *video-group-optimized* and *video-agnostic* supervised BGS algorithms can be found in a recent survey paper from Mandal *et al.* [Mandal and Vipparthi, 2021].

2.1.4 Post-Processing Methods

Over the last few years, many deep-learning-based algorithms were developed for the problem of semantic segmentation and they achieved state-of-the-art performance. Braham and Droogenbroeck introduced a post-processing step for BGS algorithms based on semantic segmentation predictions [Braham et al., 2017]. Given an input frame, they predicted a segmentation map using Pyramid Scene Parsing Network (PSPNet) [Zhao et al., 2017] and obtained pixel-wise probability predictions for se-

Table 2.1: Training and evaluation methods of supervised BGS algorithms tested on CDNet-2014.

Algorithm	**Are some frames from test videos used in training?**		**Training and evaluation methodology**
[Braham and Van Droogenbroeck, 2016]	Yes	First half of the labeled frames of the test video	*video-optimized*
[Zeng and Zhu, 2018]	Yes	Randomly selected 200 frames from the first 3000 labeled frames of the test video	*video-optimized*
[Wang et al., 2017] [Lim and Keles, 2018a] [Lim and Keles, 2018b] [Bakkay et al., 2018]	Yes	Hand picked 200 labeled frames of the test video	*video-optimized*
[Babaee et al., 2018]	Yes	5% of the labeled frames of all videos	*video-group-optimized*
[Babaee et al., 2018]	Yes	70% of the labeled frames of all videos	*video-group-optimized*
[Tezcan et al., 2019] [Mandal et al., 2019] [Tezcan et al., 2020] [Mandal and Vipparthi, 2020] [Mandal et al., 2021] [Kim and Ha, 2020] [Tezcan et al., 2021b]	No	No frame from test videos is used in training	*video-agnostic*

mantic labels such as person, car, animal, house etc. Then, they manually grouped these labels into two sets – foreground and background labels, and used this information to improve any BGS algorithm's output in a post-processing step. They obtained very competitive results by using SubSENSE [St-Charles et al., 2015b] as the BGS algorithm.

Bianco *et al.* introduced an algorithm called In Unity There Is Strength (IUTIS) which combines the results produced by several BGS algorithms [Bianco et al., 2017]. They used genetic programming to determine how to combine several BGS algorithms' outputs using a sequence of basic binary operations, such as logical "and/or", majority voting and median filtering. Their best result was achieved by using 5 top-performing BGS algorithms on the CDNet-2014 dataset at the time of publication. Zeng *et al.* followed the same idea, but instead of genetic programming, used a fully-convolutional neural network to fuse several BGS results into a single output [Zeng et al., 2019b], and outperformed IUTIS on CDNet-2014.

2.2 People Detection Using Fisheye Cameras

2.2.1 People Detection in Images from Side-View, Standard-Lens Cameras

Among traditional people-detection algorithms for standard cameras, the most popular ones are based on the histogram of oriented gradients (HOG) [Dalal and Triggs, 2005] and aggregate channel features (ACF) [Dollár et al., 2014]. Recently, deep learning algorithms have achieved outstanding performance in object and people detection [Girshick, 2015, Ren et al., 2015, Redmon et al., 2016, Liu et al., 2016, Fu et al., 2017, He et al., 2017]. These algorithms can be divided into two categories: two-stage methods and one-stage methods. The two-stage methods, such as R-CNN and its variants [Girshick, 2015, Ren et al., 2015, He et al., 2017], consist of a Region Proposal Network (RPN) which predicts regions of interest (RoIs) and a network head that refines them to produce the final bounding boxes. One-stage methods, such as variants of SSD [Liu et al., 2016,Fu et al., 2017] and YOLO [Redmon et al., 2016,Redmon and Farhadi, 2017,Redmon and Farhadi, 2018], could be viewed as independent RPNs. Given an input image, one-stage methods directly regress bounding boxes through CNNs. Recently, attention has focused on fast one-stage detectors [Zhao et al., 2019,Tan et al., 2020] and anchor-free detectors [Tian et al., 2019,Zhang et al., 2020].

2.2.2 Object Detection Using Rotated Bounding Boxes

Detection of rotated bounding boxes has been widely studied in text detection and aerial image analysis [Ma et al., 2018a,Ding et al., 2019,Yang et al., 2019b,Qian et al., 2019]. Rotation Region Proposal Network (RRPN) is a two-stage object detection algorithm which uses rotated anchor boxes and a rotated region-of-interest (RRoI) layer. RoI-Transformer [Ding et al., 2019] extended this idea by first computing a horizontal region of interest (HRoI) and then learning the warping from HRoI to RRoI. Refined Rotation RetinaNet (R^{3}Det) [Yang et al., 2019b] proposed a single-stage rotated bounding box detector by using a feature refinement layer to solve feature misalignment occurring between the region of interest and the feature, a common issue in single-stage methods. In an alternative approach, Nosaka *et al.* [Nosaka et al., 2018] used orientation-aware convolutional layers [Zhou et al., 2017b] to handle the bounding box orientation and a smooth $L1$ loss for angle regression. All of these methods use a 5-component vector for rotated bounding boxes (coordinates of the center, width, height and rotation angle) with the angle defined in $[\frac{-\pi}{2}, 0]$ range and a traditional regression loss. Due to symmetry, a rectangular bounding box having width b_w, height b_h and angle θ is indistinguishable from one having width b_h, height b_w and angle $(\theta - \pi/2)$. Hence, a standard regression loss, which does not account for this, may incur a large cost even when the prediction is close to the ground truth, e.g., if the ground-truth annotation is $(b_x, b_y, b_h, b_w, -4\pi/10)$, a prediction $(b_x, b_y, b_w, b_h, 0)$ may seem far from the ground truth, but it is not so since the ground truth is equivalent to $(b_x, b_y, b_w, b_h, \pi/10)$. Rotation Sensitive Detector (RSDet) [Qian et al., 2019] addresses this by introducing a modulated rotation loss.

2.2.3 People Detection in Images from Overhead, Fisheye Cameras

People detection using overhead, fisheye cameras is an emerging area with sparse literature. In some approaches, traditional people-detection algorithms such as HOG and ACF have been applied to fisheye images with slight modifications to account for fisheye geometry [Saito et al., 2011,Chiang and Wang, 2014,Wang et al., 2017,Krams and Kiryati, 2017]. For example, Chiang and Wang [Chiang and Wang, 2014] rotated each fisheye image in small angular steps and extracted HOG features from the top-center part of the image. Subsequently, they applied Support Vector Machines (SVM) classifier to detect people. In another algorithm, Krams and Kiryati [Krams and Kiryati, 2017] trained an ACF classifer on side-view images and dewarped the ACF features extracted from the fisheye image for person detection.

Recently, CNN-based algorithms have been applied to this problem as well. Tamura *et al.* introduced a rotation-invariant version of YOLO [Redmon et al., 2016] by training the network on a rotated version of the Common Objects in Context (COCO) dataset [Lin et al., 2014]. The inference stage in their method relies on the assumption that bounding boxes in a fisheye image are aligned with the image radius. Another YOLO-based algorithm [Seidel et al., 2018] applies YOLO to dewarped versions of overlapping windows extracted from a fisheye image.

In this dissertation, we introduce two YOLO-based rotating-window approaches for people detection from OHF cameras [Li et al., 2019] that apply geometric pre- and post-processing to realize significant performance gains. In follow-up work, we introduce Rotation-Aware People Detection (RAPiD) – an end-to-end algorithm. RAPiD uses a novel angle-aware loss function to predict the exact angle of bounding boxes without any additional assumptions. We also change the commonly-used representation of rotated bounding boxes to overcome the symmetry problem [Duan et al., 2020].

2.2.4 Video Object Detection

With the introduction of ImageNet VID challenge for object detection from video [Russakovsky et al., 2015], new algorithms leveraging both spatial and temporal information have been developed. Several algorithms apply post-processing to the output of object detection algorithms, which consider video frames as still images [Han et al., 2016, Belhassen et al., 2019, Sabater et al., 2020]. For example, Robust and Efficient Post-Processing for video object detection (REPP) [Sabater et al., 2020] links the bounding box predictions between consecutive frames using a learning-based approach. Then, it creates and re-scores bounding box tubelets in temporal dimension. Another approach is to design an end-to-end video object detection algorithm that leverages both the temporal and spatial information [Zhu et al., 2017, Zhang et al., 2018, Wu et al., 2019, Lin et al., 2019, Liu et al., 2019, Chen et al., 2020]. For example, Flow-Guided Feature Aggregation (FGFA) [Zhu et al., 2017] uses optical flow to warp the feature maps of past and future frames and then aggregates the warped feature maps to detect objects in the current frame. Similarly, SELSA [Wu et al., 2019] aggregates the feature maps of frames from the whole video based on their semantic similarities with the current frame.

We adapt the feature map warping and feature aggregation ideas introduced in FGFA and the post-processing method introduced in REPP into RAPiD to improve its performance.

Chapter 3

Background Subtraction for Unseen Videos Using Deep Learning

In Section 2.1, we proposed to group BGS algorithms into 4 categories:

- unsupervised BGS algorithms,
- video- or video-group-optimized supervised BGS algorithms,
- video-agnostic supervised BGS algorithms,
- post-processing methods that improve the results of BGS algorithms.

Prior to 2019, all of the supervised BGS algorithms were either video-optimized or video-group-optimized and their performance on unseen videos was unknown. In July 2019, we introduced *Background Subtraction for Unseen Videos* (BSUV-Net) [Tezcan et al., 2019, Tezcan et al., 2020], the first supervised *video-agnostic* BGS algorithm. In this chapter[1], we describe BSUV-Net[2] in detail and analyze its performance on a widely-used BGS dataset, CDNet-2014 [Goyette et al., 2014]. Furthermore, we present a detailed ablation study which demonstrates the effectiveness of individual components of BSUV-Net. BSUV-Net is a fully-convolutional neural network for predicting foreground of an *unseen* video. A key feature of this approach is that the training and test sets are composed of frames originating from different videos.

---

[1] This work was published in the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV) [Tezcan et al., 2020].

[2] The source code of BSUV-Net is publicly available at `github.com/ozantezcan/BSUV-Net-inference`

This guarantees that no ground-truth data from the test videos have been shown to the network in the training phase. By employing two reference backgrounds at different time scales, BSUV-Net addresses two challenges often encountered in BGS: varying scene illumination and intermittently-static objects that tend to get absorbed into the background. We also propose a novel data augmentation method which further improves BSUV-Net's performance under varying illumination. Furthermore, motivated by recent work on the use of semantic segmentation in BGS [Braham et al., 2017], we improve our method's accuracy by inputting semantic information along with the reference backgrounds and the current frame.

3.1 Spatio-Temporal Inputs at Multiple Time Scales

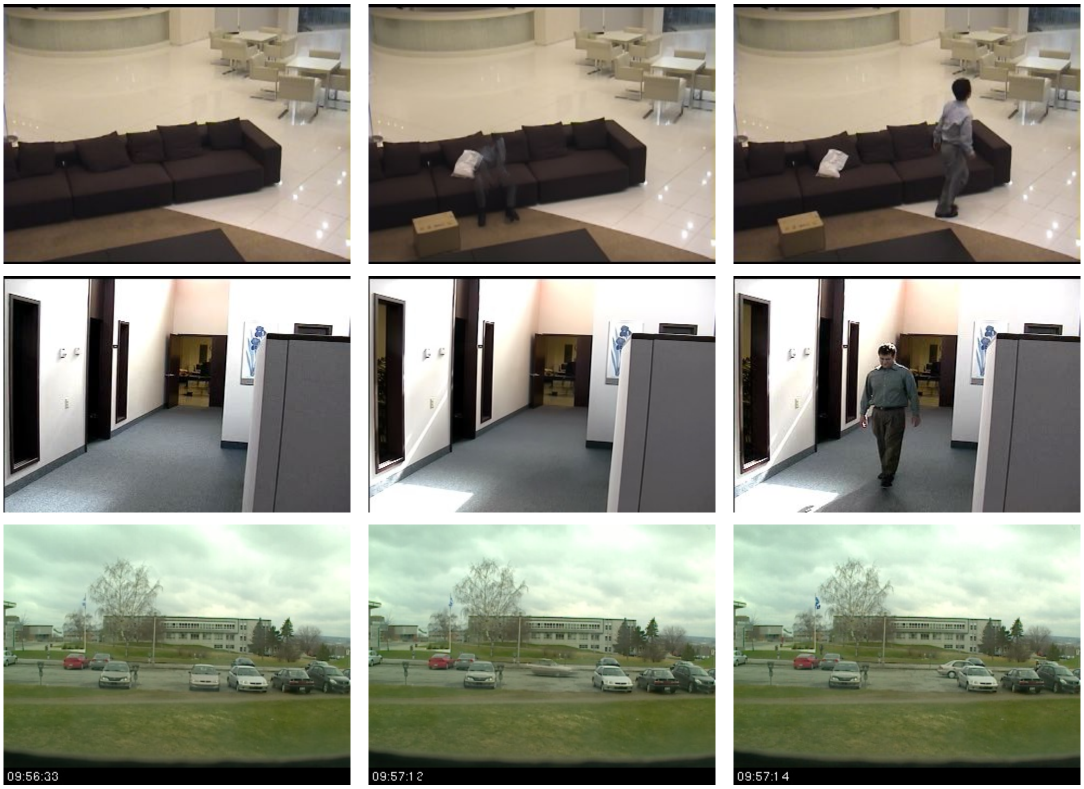
Segmenting an unseen video frame into foreground and background regions without using any information about the background would be an ill-defined problem. In BSUV-Net, we use two reference frames to characterize the background. One frame is an *empty* background frame, with no people or other objects of interest, which can be identified manually (e.g., at camera installation), captured using side information (e.g., door sensor in indoor scenarios) or computed (e.g., median filtering over a long time span, such as hours). This provides an accurate reference that is very helpful for segmenting intermittently-static objects in the foreground. However, due to dynamic factors, such as illumination variations, this reference frame may not be valid after some time. To counteract this, we use another reference frame that characterizes *recent* background, for example by computing the median of 100 frames immediately preceding the frame being processed. However, this frame might not as accurately represent the background as the first reference frame since we cannot guarantee that there will be no foreground objects in it (if such objects are present for less than 50 frames, the temporal median will suppress them). Figure 3·1 shows three examples of

BSUV-Net input for videos from CDNet-2014: empty and recent background frames and a current frame. In the top row, the empty-background frame is clearly a better estimate of the true background than the recent-background frame which is distorted by the appearance of intermittently static foreground objects (the box on the floor and the package on the sofa). On the other hand, the empty background in the middle row is affected by illumination change (bottom left corner), which makes the recent background a better background estimate for the current frame. In the bottom row, neither the empty nor recent background are free of foreground objects (the empty background includes the moving car as parked and the recent background includes a ghosting effect of the same car). By using two reference frames captured at different time scales, we aim to leverage benefits of each frame type.

3.2 Leveraging Semantic Segmentation

Braham *et al.* [Braham et al., 2017] have shown that leveraging the results of semantic segmentation in a post-processing step significantly improves the performance of a BGS algorithm. In BSUV-Net, we follow a different idea and use semantic information as an additional input channel to our neural network. In this way, we let our network learn how to use this information. To extract semantic segmentation information, we use a state-of-the-art CNN called DeepLabv3 [Chen et al., 2017] trained on ADE20K [Zhou et al., 2017a], an extensive semantic-segmentation dataset with 150 different class labels and more than 20,000 images with dense annotations.

Let us denote the set of object classes in ADE20K as $C = \{c_0,\ c_1,\ \ldots,\ c_{149}\}$. Following the same procedure as in [Braham et al., 2017], we divide these classes into two sets: foreground and background objects. As foreground objects, we use person, car, cushion, box, book, boat, bus, truck, bottle, van, bag and bicycle. The rest of the classes are used as background objects. The softmax layer of DeepLabv3



(a) Empty background **(b)** Recent background **(c)** Current frame

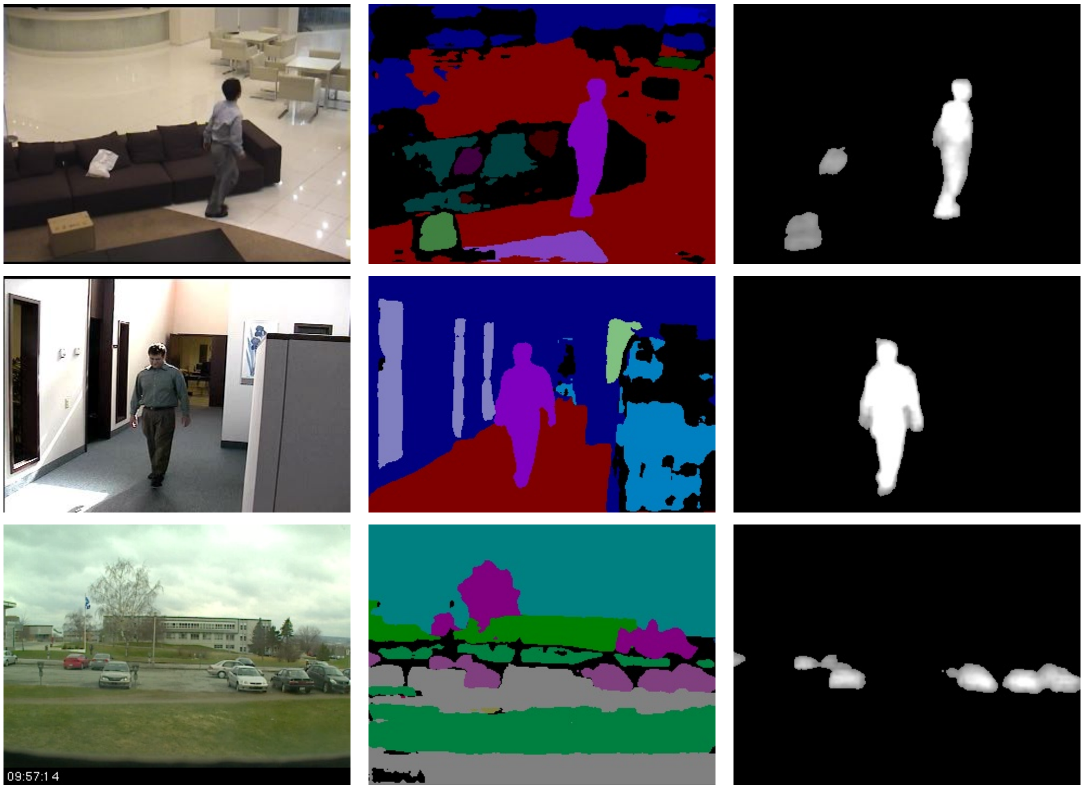
Figure 3·1: Examples of spatio-temporal inputs to BSUV-Net for three videos from CDNet-2014 produces pixel-wise class probabilities. Let us denote DeepLabv3 as a function $\mathcal{F} : \mathbf{I} \in \mathbb{R}^{w\times h\times 3} \rightarrow \mathbf{S} \in \mathbb{R}^{w\times h\times 150}$ where w and h denote the width and height of the image with 3 color channels (RGB). $\mathbf{S}[m, n, j]$ represents an estimate of the probability that pixel $\mathbf{I}[m, n, :]$ belongs to class c_j. Then, we compute a foreground probability map (FPM) as follows:

$$\mathbf{FPM}[m, n] = \sum_{c_j \in Fr} \mathbf{S}[m, n, j] \tag{3.1}$$

where Fr is the set of foreground classes. Figure 3·2 shows examples of semantic segmentation and FPM for the current frames from Figure 3·1. Clearly, FPM is very successful in extracting semantic details, but itself is not enough to estimate foreground since some objects in Fr can appear in the background as well (e.g., a box in the top row and parked cars in the bottom row). By using FPM as an additional input channel, we hope to leverage semantic information for improved foreground estimation.

3.3 Notation

Let us introduce mathematical notation for input-label pair used in BSUV-Net. The input is denoted as $\mathbf{X} \in \mathbb{R}^{w\times h\times 12}$ and computed as the channel-wise concatenation of $\mathbf{I_E}, \mathbf{I_R}, \mathbf{I_C} \in \mathbb{R}^{w\times h\times 4}$ an empty background, a recent background and the current frame, respectively, where w, h are the width and height of each image. Each image has 4 channels: three colors (R, G, B) plus FPM discussed in Section 3.2. Similarly, let $\mathbf{I_{FG}} \in \{0, 1\}^{w\times h}$ be the corresponding foreground label field where 0 represents the background and 1 – the foreground. We will use this notation to formulate BSUV-Net, its loss function and a temporal data augmentation.



(a) Current frame **(b)** Semantic segmentation **(c)** FPM

Figure 3·2: Examples of semantic segmentation and foreground probability map (FPM) for the current frames from Figure 3·1. Different colors in (b) represent 150 object categories of ADE20K [Zhou et al., 2017a] and the color coding in (c) is from 0 (black) to 1 (white).

3.4 Improving Resilience to Illumination Changes by Temporal Data Augmentation

Since neural networks have millions of parameters, they are very prone to overfitting. A widely-used method for reducing overfitting in computer-vision problems is to enlarge the dataset by applying several data augmentations such as random crops, rotations and noise addition. Since we are dealing with videos in this paper, we can also add augmentation in the temporal domain.

In real-life BGS problems, there might be a significant illumination difference between an empty-background frame acquired at an earlier time and the current frame. However, only a small portion of videos in CDnet-2014 capture significant illumination changes which limits BSUV-Net's generalization performance. Therefore, we introduce a new data-augmentation technique to account for global illumination changes between the empty reference frame and the current frame. Using the notation introduced in Section 3.3, an augmented version of the input images can be formulated as follows:

$$\widehat{\mathbf{I}}_k[i,\ j,\ c] = \mathbf{I}_k[i,\ j,\ c] + \mathbf{d}_k[c] \text{ for } k \in \{\mathbf{E},\ \mathbf{R},\ \mathbf{C}\}, c = 1, 2, 3$$

where $\mathbf{d_E}, \mathbf{d_R}, \mathbf{d_C} \in \mathbb{R}^3$ represent illumination offsets applied to RGB channels of the input images. By choosing $\mathbf{d}_k$ randomly for each example during training (see Section 3.8.2 for details), we can make the network resilient to illumination variations. In our experiments we forced $\mathbf{d_R} = \mathbf{d_C}$ since, illumination change rarely appears in a short time window covered by the recent background. Note that, this augmentation does not alter the FPM channels of the inputs.

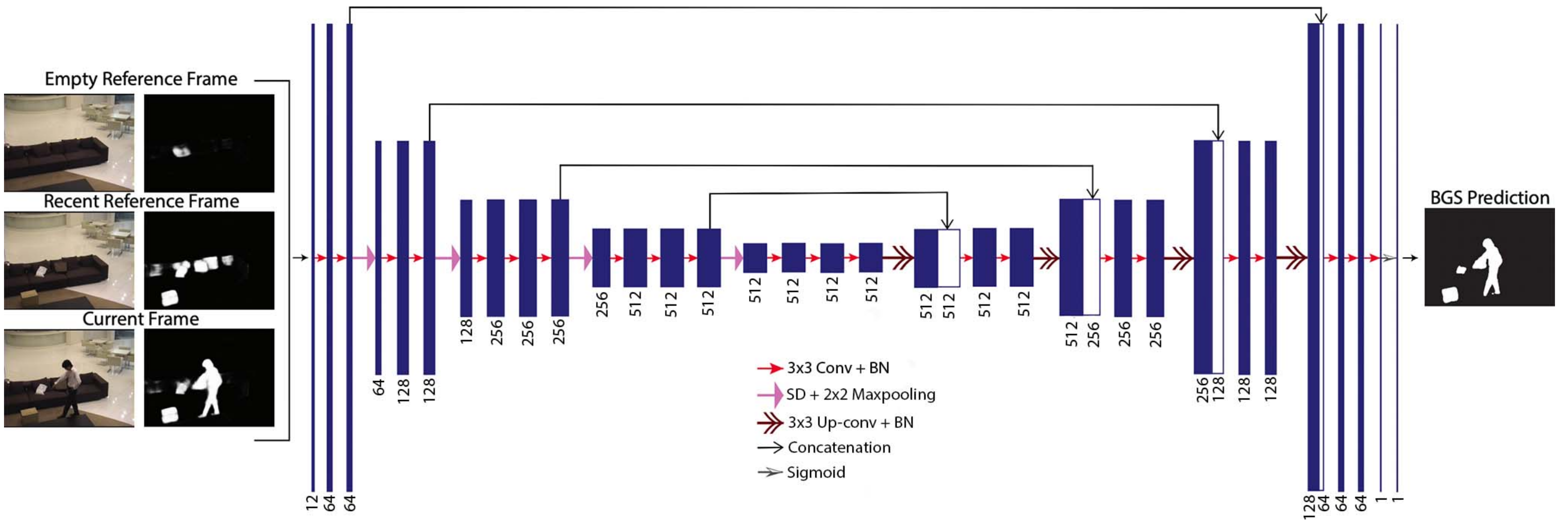


Figure 3·3: Network architecture of BSUV-Net. BN stands for batch normalization and SD stands for spatial dropout. Grayscale images at the network input show foreground probability maps (FPM) of the corresponding RGB frames.

3.5 Network Architecture

We use a UNET-type [Ronneberger et al., 2015] fully-convolutional neural network (FCNN) with residual connections. The architecture of BSUV-Net has two parts: encoder and decoder, and is shown in Figure 3·3. In the encoder network, we use 2×2 max-pooling operators to decrease the spatial dimensions. In the decoder network, we use up-convolutional layers (transposed convolution with a stride of 2) to increase the dimensions back to those of the input. In all convolutional and up-convolutional layers, we use 3×3 convolutions as in VGG [Simonyan and Zisserman, 2014]. The residual connections from the encoder to the decoder help the network combine low-level visual information gained in the initial layers with high-level visual information gained in the deeper layers. Since our aim is to increase the performance on unseen videos, we use strong batch normalization (BN) [Ioffe and Szegedy, 2015] and spatial dropout (SD) [Tompson et al., 2015] layers to increase the generalization capacity. Specifically, we use a BN layer after each convolutional and up-convolutional layer, and an SD layer before each max-pooling layer. Since our task can be viewed as a binary segmentation, we use a sigmoid layer as the last layer in BSUV-Net. The operation of the overall network can be defined as a nonlinear map $\mathbf{G}(\mathbf{W}) : \mathbf{X} \rightarrow \widehat{\mathbf{I}}_{\mathbf{FG}}$ where $\mathbf{X}$ is defined in Section 3.3, $\mathbf{W}$ represents the parameters of neural network $\mathbf{G}$, and $\widehat{\mathbf{I}}_{\mathbf{FG}} \in [0,1]^{w \times h}$ is a pixel-wise foreground probability prediction. Note that since this is a fully-convolutional neural network, it does not require fixed input size; any frame size can be used, but some padding may be needed to account for max-pooling operations.

3.6 Loss Function

In most BGS datasets, the number of background pixels is much larger than the number of foreground pixels. Due to this class imbalance, the commonly-used loss functions, such as cross-entropy and mean-squared error tend to favor the dominant class. A good alternative for unbalanced binary datasets is the Jaccard index. Since the network output is a probability map, we used a relaxed form of the Jaccard index in the loss function, defined as follows:

$$J_R(\mathbf{I_{FG}}, \widehat{\mathbf{I}}_{\mathbf{FG}}) = \frac{T + \sum\limits_{m,n} (\mathbf{I_{FG}}[m,n] \widehat{\mathbf{I}}_{\mathbf{FG}}[m,n])}{T + \sum\limits_{m,n} \Big(\mathbf{I_{FG}}[m,n] + \widehat{\mathbf{I}}_{\mathbf{FG}}[m,n] - \mathbf{I_{FG}}[m,n] \widehat{\mathbf{I}}_{\mathbf{FG}}[m,n] \Big)} \tag{3.2}$$

where T is a smoothing parameter and m, n are the spatial locations. The loss function is computed as $1 - J_R$. Since J_R is a region-based similarity metric, it will not be affected by the data imbalance. The numerator of J_R forces BSUV-Net to increase the number of true positives, whereas the denominator forces it to produce as few false positives as possible.

3.7 Video-Agnostic Evaluation Strategy for Supervised Algorithms

The most commonly used BGS datasets with a variety of scenarios and pixel-wise ground-truth annotations are CDNet-2014 [Goyette et al., 2014], LASIESTA [Cuevas et al., 2016] and SBMI2015 [Maddalena and Petrosino, 2015]. Among these 3 datasets, only CDNet-2014 has a well-maintained evaluation server, that keeps a cumulative performance record of the uploaded algorithms. Moreover, it has been the most widely-used dataset for BGS in recent years with publicly-available evaluation results for nearly all of the published BGS algorithms.

Since one of our aims is to compare the performance of BSUV-Net with state-of-the-art *video-agnostic* BGS algorithms on unseen videos, the availability of public results for these algorithms is critical. Therefore, we use CDNet-2014 as our evaluation dataset. Unfortunately, CDNet-2014 [Goyette et al., 2014] does not provide different videos for training and testing. Instead, it provides some frames from each video as training data and the remaining ones – as test data. However, this type of division is not useful for evaluating the performance on *unseen* videos.

For comparing the performance of different models on *unseen* videos, we split the dataset into 18 different sets of training/testing videos as shown in Tables 3.1. The splits are structured in such a manner that every video appears in the test set of exactly one split, but when it does so, it does not appear in the training set for that split. When training a supervised algorithm, the main assumption is that the training set is diverse enough to cover a wide range of test scenarios. For example, if there are no examples that include shadow in the training set, then it is impossible for the network to learn how to classify shadow regions. Therefore, we designed the splits so that the training set for each split contains some videos from the same category as the test videos. We did not perform a full "leave-k-videos-out" cross-validation due to prohibitive time needed to train BSUV-Net. In all of the tests, we used videos from "baseline", "bad weather", "intermittent object motion", "low frame rate" and "shadow" categories during training since they span most of the common scenarios. For videos from more difficult scenarios, we progressively added additional categories into the training set. In particular, we considered "PTZ", "thermal" and "turbulence" categories as the most difficult ones since they have substantially different data characteristics from other categories. "PTZ" is the only category with significant camera movement and zoom in/out, while "thermal" and "turbulence" categories capture different scene properties than the remaining categories (far- and near-infrared spectrum instead of RGB, respectively). For these 3 categories, we used more videos in the training set, than in the other categories.

Table 3.1: Training/testing splits used for algorithm evaluation. (✓ – video is used in training, ✗ – video is used in testing).

		Split Number																	
Category	**Video**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
baseline	highway	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	pedestrians	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	office	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	PETS2006	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
bad weather	skating	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	blizzard	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	snowFall	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	wetSnow	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
intermittent object motion	abandonedBox	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	parking	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	sofa	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	streetLight	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	tramstop	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	winterDriveway	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
low framerate	port 0.17fps	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	tramCrossroad 1fps	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	tunnelExit 0.35fps	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	turnpike 0.5fps	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

		Split Number																	
Category	**Video**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
shadow	backdoor	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	bungalows	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	busStation	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	copyMachine	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	cubicle	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	peopleInShade	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
camera jitter	badminton							✗	✓					✓	✓	✓	✓	✓	✓
	traffic							✗	✓					✓	✓	✓	✓	✓	✓
	boulevard							✓	✗					✓	✓	✓	✓	✓	✓
	sidewalk							✓	✗					✓	✓	✓	✓	✓	✓
dynamic background	boats									✓	✗			✓	✓	✓	✓	✓	✓
	canoe									✗	✓			✓	✓	✓	✓	✓	✓
	fall									✗	✓			✓	✓	✓	✓	✓	✓
	fountain01									✓	✗			✓	✓	✓	✓	✓	✓
	fountain02									✗	✓			✓	✓	✓	✓	✓	✓
	overpass									✓	✗			✓	✓	✓	✓	✓	✓

		Split Number																	
Category	**Video**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
night videos	bridgeEntry											✗	✓						
	busyBoulvard											✓	✗						
	fluidHighway											✗	✓						
	streetCornerAtNight											✓	✗						
	tramStation											✗	✓						
	winterStreet											✓	✗						
PTZ	continuousPan													✓	✗				
	intermittentPan													✓	✗				
	twoPositionPTZCam													✗	✓				
	zoomInZoomOut													✗	✓				
thermal	corridor															✓	✗	✓	✓
	diningRoom															✓	✗	✓	✓
	lakeSide															✗	✓	✓	✓
	library															✗	✓	✓	✓
	park															✗	✓	✓	✓
turbulence	turbulence0															✓	✓	✗	✓
	turbulence1															✓	✓	✗	✓
	turbulence2															✓	✓	✓	✗
	turbulence3															✓	✓	✓	✗

3.8 Experimental Results

3.8.1 Dataset and Evaluation Metrics

In order to evaluate the performance of BSUV-Net, we used CDNet-2014 [Goyette et al., 2014], the largest BGS dataset with 53 natural videos from 11 categories including challenging scenarios such as shadows, night videos, dynamic background, etc. The spatial resolution of videos varies from 320×240 to 720×526 pixels. Each video has a region of interest labelled as either 1) foreground, 2) background, 3) hard shadow or 4) unknown motion. When measuring an algorithm's performance, we ignored pixels with unknown motion label and considered hard-shadow pixels as background. Our treatment of hard shadows is consistent with what is done in CDNet-2014 for the change-detection task.

In CDNet-2014 [Goyette et al., 2014], the authors proposed seven binary performance metrics to cover a wide range of BGS cases: *Recall* (Re), specificity (Sp), false positive rate (FPR), false negative rate (FNR), percentage of wrong classifications (PWC), *Precision* (Pr) and *F-score* (F_1). They also introduced two ranking-based metrics namely "average ranking" (R) and "average ranking accross categories" (R_{cat}) which combine all 7 metrics into ranking scores. The details of these rankings can be found on the dataset website at `changedetection.net`. In our evaluations, we omitted FPR and FNR since they are equal to $(1 - Sp)$ and $(1 - Re)$ respectively.

3.8.2 Training and Evaluation Details

As discussed in Section 3.7, we applied a *video-agnostic* evaluation methodology in all experiments using 18 different combinations of training/testing video sets. During training on each set, we used 200 frames suggested in [Zeng and Zhu, 2018] for each video in that training set.

When training BSUV-Net on different sets, we used *exactly the same* hyperparameter values across all sets to make sure that we are not tuning our network to specific videos. In all of our experiments, we used ADAM optimizer with a learning rate of 10^{-4}, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. The minibatch size was 8 and we trained for 50 epochs. As the empty background frame, we used the median of all foreground-free frames within the first 100 frames. In a few videos containing highway traffic, the first 100 frames did not contain a single foreground-free frame. For these videos, we hand-picked empty frames (e.g., in groups) and used their median as the empty reference. Although this may seem like a limitation, in practice one can randomly sample several hundred frames at the same time of the day across several days (similar illumination) and median filter them to obtain an empty background frame (due to random selection, a moving object is unlikely to occupy the same location in more than 50% of frames). Since there is no single empty background frame in videos from the pan-tilt-zoom (PTZ) category, we slightly changed the inputs. Instead of "empty background + recent background" pair we used "recent background + more recent background" pair, where the recent background is computed as the median of 100 preceding frames and the more recent background is computed as the median of 30 preceding frames.

Although BSUV-Net can accept frames of any spatial dimension, we used a *fixed* size of 224×224 pixels (randomly cropped from the input frame) so as to leverage parallel GPU processing in the training process. We applied random data augmentation at the beginning of each epoch. For illumination resilience, we used the data augmentation method of Section 3.4 with $\mathbf{d_R}[k] = \mathbf{d_C}[k] = I + I_k$ where $I \sim \mathcal{N}(0, 0.1^2)$ and $I_k \sim \mathcal{N}(0, 0.04^2)$ for $k \in \{1, 2, 3\}$. Similarly, $\mathbf{d_E}[k] = \mathbf{d_C}[k] + I^E + I_k^E$ where $I^E \sim \mathcal{N}(0, 0.1^2)$ and $I_k^E \sim \mathcal{N}(0, 0.04^2)$ for $k \in \{1, 2, 3\}$. We also added random Gaussian noise from $\mathcal{N}(0, 0.01^2)$ to each pixel in each color channel. For pixel values, we used double precision numbers that lie between 0 and 1.

In the evaluation step, we did not apply any scaling or cropping to the inputs. To obtain binary maps, we applied thresholding with threshold $\theta = 0.5$ to the output of the sigmoid layer of BSUV-Net.

3.8.3 Quantitative Results

Table 3.2 compares BSUV-Net against state-of-the-art BGS algorithms in terms of the five metrics and two rankings discussed in Section 3.8.1. All quantitative results shown are computed by the CDNet-2014 evaluation server at `changedetection.net` to reflect the real performance on test data. Since BSUV-Net is *video-agnostic*, comparing it with *video-optimized* or *video-group-optimized* algorithms would not be fair and we omit them here. Instead, we compare BSUV-Net with state-of-the-art unsupervised algorithms, namely SWCD [Işık et al., 2018], WisenetMD [Lee et al., 2018] and PAWCS [St-Charles et al., 2015a] which, by definition, are *video-agnostic*. In Table 3.2, we group these algorithms under *self-contained algorithms* category. We exclude RTSS [Zeng et al., 2019a] and $3DFR$ [Mandal et al., 2019] from Table 3.2 since their results are not available at `changedetection.net` for the test frames.

However, we include the results of IUTIS-5 [Bianco et al., 2017] and SemanticBGS [Braham et al., 2017], but we list them separately because these are *post-processing* algorithms. Note that, both IUTIS-5 and SemanticBGS can be applied to any BGS algorithm from Table 3.2, including BSUV-Net, to improve its performance. To show this, we report the result of BSUV-Net post-processed by SemanticBGS.

We include FgSegNet v2 [Lim and Keles, 2018b] in the *self-contained algorithms* category since it is currently the best performing algorithm on CDNet-2014. However, since FGSegNet v2's performance reported at `changedetection.net` has been obtained in a *video-optimized* manner, we trained it anew in a *video-agnostic* manner using the same methodology that we used for BSUV-Net. As expected, this caused a huge performance decrease of FgSegNet v2 compared to it's *video-optimized* training.

Table 3.2: Performance comparison of BSUV-Net against state-of-the-art methods for **unseen videos** on CDNet-2014. For fairness, we separated the post-processing and self-contained algorithms.

Method	R	R_{cat}	Re	Sp	PWC	Pr	F_1
Post-processing algorithms							
BSUV-net + SemanticBGS*	**9.57**	14.27	**0.8179**	0.9944	1.1326	**0.8319**	**0.7986**
IUTIS-5* + SemanticBGS*	9.71	11.91	0.7890	**0.9961**	**1.0722**	0.8305	0.7892
IUTIS-5*	12.14	**10.91**	0.7849	0.9948	1.1986	0.8087	0.7717
Self-contained algortihms							
BSUV-net	**9.71**	**14.00**	**0.8203**	0.9946	**1.1402**	**0.8113**	**0.7868**
SWCD	16.43	20.00	0.7839	0.9930	1.3414	0.7527	0.7583
WisenetMD	17.29	15.82	0.8179	0.9904	1.6136	0.7535	0.7668
PAWCS	14.71	16.09	0.7718	**0.9949**	1.1992	0.7857	0.7403
FgSegNet v2	45.57	45.09	0.5119	0.9411	7.3507	0.4859	0.3715

As is clear from Table 3.2, BSUV-Net outperforms its competitors on almost all of the metrics. The *F-score* performance demonstrates that BSUV-Net achieves excellent results without compromising either *Recall* or *Precision*. Table 3.2 also shows that the performance of BSUV-Net can be improved even further by combining it with SemanticBGS. The combined algorithm outperforms all of the *video-agnostic* algorithms that are available at `changedetection.net`.

Table 3.3 compares the per-category *F-score* performance of BSUV-Net against state-of-the-art BGS algorithms. For RTSS [Zeng et al., 2019a], the values of performance metrics shown in Table 3.3 are as reported in their paper. Individual columns report the *F-score* for each of the 11 categories from `changedetection.net`, while the last column reports the mean *F-score* across all categories. Similarly to Table 3.2, we divided this table into *post-processing* and *self-contained algorithms*. It can be observed that BSUV-Net achieves the best performance in 5 out of 11 categories, but it is outperformed by RTSS in terms of the overall performance. BSUV-Net performs

Table 3.3: Performance comparison of BSUV-Net against state-of-the-art methods according to per-category *F-score* for **unseen videos** on CDNet-2014.

Method	Bad weather	Low framerate	Night	PTZ	Thermal	Shadow
Post-processing algorithms						
BSUV-net + SemanticBGS*	**0.8730**	0.6788	**0.6815**	**0.6562**	**0.8455**	**0.9664**
IUTIS-5* + SemanticBGS*	0.8260	0.7888	0.5014	0.5673	0.8219	0.9478
IUTIS-5*	0.8248	**0.7743**	0.5290	0.4282	0.8303	0.9084
Self-contained algortihms						
BSUV-net	**0.8713**	0.6797	**0.6987**	**0.6282**	**0.8581**	0.9233
RTSS	0.8662	0.6771	0.5295	0.5489	0.8510	**0.9551**
SWCD	0.8233	**0.7374**	0.5807	0.4545	**0.8581**	0.8779
WisenetMD	0.8616	0.6404	0.5701	0.3367	0.8152	0.8984
PAWCS	0.8152	0.6588	0.4152	0.4615	0.8324	0.89133
FgSegNet v2	0.3277	0.2482	0.2800	0.3503	0.6038	0.5295

Method	Int. obj. motion	Camera jitter	Dynamic backgr.	Base-line	Turbu-lence	Overall
Post-processing algorithms						
BSUV-net + SemanticBGS*	0.7601	0.7788	0.8176	**0.9640**	0.7631	**0.7986**
IUTIS-5* + SemanticBGS*	**0.7878**	**0.8388**	**0.9489**	0.9604	0.6921	0.7892
IUTIS-5*	0.7296	0.8332	0.8902	0.9567	**0.7836**	0.7717
Self-contained algortihms						
BSUV-net	0.7499	0.7743	0.7967	**0.9693**	0.7051	0.7868
RTSS	**0.7864**	**0.8396**	**0.9325**	0.9597	0.7630	**0.7917**
SWCD	0.7092	0.7411	0.8645	0.9214	0.7735	0.7583
WisenetMD	0.7264	0.8228	0.8376	0.9487	**0.8304**	0.7535
PAWCS	0.7764	0.8137	0.8938	0.9397	0.6450	0.7403
FgSegNet v2	0.2002	0.4266	0.3634	0.6926	0.0643	0.3715

significantly poorer than RTSS and some other algorithms in “intermittent object motion”, “camera jitter”, “dynamic background” and “turbulence” categories. We believe this is related to the supervised nature of BSUV-Net. Since it is a data-based algorithm and the representation of these categories in CDNet-2014 is limited, BSUV-Net is not able to capture the necessary details to solve these challenges. In Chapter 4, we will introduce an improved version of BSUV-net that addresses these challenges.

Note, that BSUV-Net has a striking performance advantage in the “night” category. All videos in this category are traffic-related and many cars have headlights turned on at night which causes significant local illumination variations in time. BSUV-Net’s excellent performance in this category demonstrates that the proposed model is indeed largely illumination-invariant.

3.8.4 Visual Results

A visual comparison of BSUV-Net with SWCD [Işık et al., 2018] and WisenetMD [Lee et al., 2018] is shown in Figure 3·4. Each row shows a sample frame from one of the videos in one of the 9 categories. It can be observed that BSUV-Net produces visually the best results for almost all categories.

In the “night” category, SWCD and WisenetMD produce many false positives because of local illumination changes. BSUV-Net produces better results since it is designed to be illumination-invariant. In the “shadow” category, BSUV-Net performs much better in the shadow regions. Results in the “intermittent object motion” and “baseline” categories show that BSUV-Net can successfully detect intermittently-static objects. It is safe to say that BSUV-Net is capable of simultaneously handling the discovery of intermittently-static objects and also the dynamic factors such as illumination changes.

An inspection of results in the “dynamic background” category shows that BSUV-

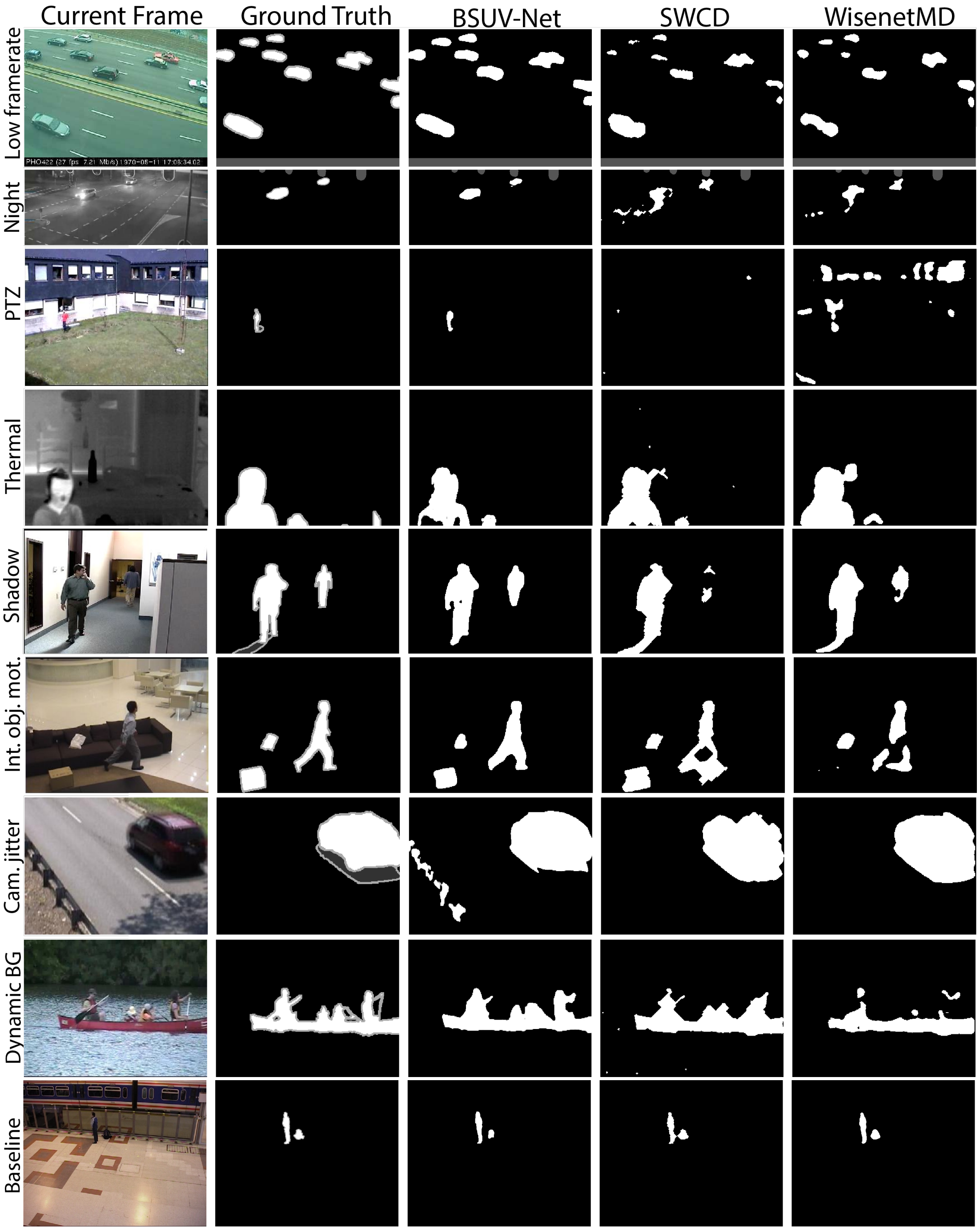


Figure 3·4: Visual comparison of sample results produced by BSUV-Net, SWCD and WisenetMD on **unseen videos** from CDNet-2014.

Net has detected most of the foreground pixels but failed to detect the background pixels around the foreground objects. We believe this is due to the blurring effect of the median operation that we used in the computation of background frames. Using more advanced background models as an input to BSUV-Net might improve the performance in this category.

3.8.5 Ablation Study

One of the contributions of BSUV-Net is its multi-channel input composed of two background frames from different time scales and a foreground probability map (FPM). Another contribution is a temporal data augmentation tailored to handling illumination changes. In Table 3.4, we explore their impact on *Precision*, *Recall* and *F-score*. Each column on the left represents one characteristic and each row represents a different combination of these characteristics. RGB channels of the current frame are used in all of the combinations. “Empty BG” and “Recent BG” refer to the use of empty and\or recent background frames, respectively, in addition to the current frame. “Data aug.” refers to temporal data augmentation described in Section 3.4 and “FPM” refers to the use of semantic FPM channel in addition to the RGB channels for all input frames. It is clear that all these characteristics have a significant impact on the overall performance. Using only the current frame as input results in very poor metrics. The introduction of empty or/and recent background frames leads to a significant improvement. Adding temporal data augmentation or/and FPM channels further improves the performance and the final network achieves state-of-the-art results.

Thus far, we have proposed to add semantic FPM channel as input in order to improve our algorithm’s performance. However, if the selection of background and foreground object categories were optimized for each video, FPM could be used as a BGS algorithm by itself. This optimized selection would require prior information about

Table 3.4: Impact of background frames, data augmentation to combat illumination changes and semantic information (FPM) on BSUV-Net performance.

Current frame	Empty BG	Recent BG	Data aug.	FPM	Pr	Re	F_1
✓					0.3615	0.5509	0.3476
✓	✓				0.6994	0.7686	0.6819
✓		✓			0.6976	0.7064	0.6351
✓	✓	✓			0.7658	0.7606	0.7156
✓	✓	✓	✓		0.7574	0.8159	0.7447
✓	✓	✓		✓	0.7807	0.7747	0.7450
✓	✓	✓	✓	✓	0.8113	0.8203	0.7868

objects in each video (to compute FPM) and, therefore, would not qualify as a *video-agnostic* method. In our algorithm, however, we use the same background/foreground object categories for all videos and combine the FPM channels, computed from the current frame and reference backgrounds, with the RGB channels. In particular, when applying DeepLabv3 [Braham et al., 2017] to compute FPM frames, we used background/foreground object categories suggested in the paper, which are likely incorrect for some videos. To demonstrate that our algorithm is not replicating FPM but leverages its semantic information to boost performance, we compared BSUV-Net with thresholded FPM used as a BGS result (Table 3.5). It is clear that FPM alone is not a powerful tool for BGS as it is significantly outperformed by BSUV-Net.

One could also modify DeepLabv3 to predict only two classes (foreground and background) and train it on CDNet-2014. We did not perform such a test; since DeepLabv3 is still an image-based network that does not use any temporal information, this approach can be expected to perform similarly to other *video-* or *video-group-optimized* algorithms (e.g., FgSegNet [Lim and Keles, 2018a]).

While in BSUV-Net we assume that the *empty background frame* is foreground-free, CDNet-2014 does not provide empty background frames. Therefore, in some

Table 3.5: Comparison of BSUV-Net with thresholded FPM used as a BGS result (probability threshold equals 0.5).

Method	Pr	Re	F_1
FPM	0.6549	0.6654	0.5846
BSUV-net	0.8113	0.8203	0.7868

Table 3.6: Comparison of manual and automatic selection of empty background frames in BSUV-Net.

Empty background selection	Pr	Re	F_1
Automatic	0.8207	0.7812	0.7639
Manual	0.8113	0.8203	0.7868

videos, we manually selected empty background frames from among the initial frames as explained in Section 3.8.2. In Table 3.6, we show the impact of this manual process by comparing the manual selection strategy with an automatic one, that is using the median of all frames in the test video as an *empty background frame*. Clearly, the automatic selection slightly improves *Precision* while significantly decreasing *Recall*. We believe this is due to the increase of false negatives caused by the appearance of some of the foreground objects in the empty background. Since videos in CDnet-2014 are rather short (at most 10 minutes), in some cases the median of all frames does not accurately represent an empty background. However, for stationary surveillance cameras in a real-life scenario it is often possible to compute an empty background, for example by taking the median of frames at the same time of the day (when it is expected to be empty) over many days.

3.9 Discussion

We introduced a novel deep-learning algorithm for background subtraction on unseen videos and proposed a *video-agnostic* evaluation methodology that treats each video in a dataset as unseen. The input to BSUV-Net consists of the current frame and two reference frames from different time-scales, along with semantic information for all three frames (computed using Deeplabv3 [Chen et al., 2017]). To increase the generalization capacity of BSUV-Net, we formulated a simple, yet effective, illumination-change model. Experimental results on CDNet-2014 show that BSUV-Net outperforms state-of-the-art *video-agnostic* BGS algorithms in terms of 6 out of 7 performance metrics (see Table 3.2). Its performance can be further improved by adding SemanticBGS [Braham et al., 2017] as a post-processing layer. This shows great potential for deep-learning BGS algorithms designed for unseen or unlabeled videos.

Although the overall performance of BSUV-Net is very promising, Table 3.3 shows that it suffers in certain challenging scenarios, such as "intermittent object motion", 'dynamic background", "camera jitter" or "turbulence". As discussed in Section 3.8.3, we believe this is due to the limited number of such videos in CDNet-2014. Since BSUV-Net is a supervised learning algorithm, its performance is limited by the variety of scenarios in the dataset that it is trained on. In the next chapter, we will introduce spatio-temporal data augmentations to increase the robustness of BSUV-Net to challenging scenarios such as moving cameras, camera jitter and intermittently static objects.

Chapter 4

Spatio-Temporal Data Augmentations for Supervised Background Subtraction

In Chapter 3, we introduced BSUV-Net, a novel supervised background subtraction algorithm designed for unseen videos that showed superior performance compared to its competitors. As discussed in Section 3.2, one of the key elements behind the success of BSUV-Net is its use of a temporal data augmentation to mimic illumination variations that might happen in real-world applications. However, as discussed in Section 3.8.3, BSUV-Net does not perform well on some of the challenging categories (e.g. "intermittent object motion" and "camera jitter") of CDNet-2104 that have limited number of examples. In this chapter[1], we address this by introducing spatio-temporal data augmentations designed to mimic such challenges and increase the robustness of BSUV-Net.

We introduce BSUV-Net 2.0[2] which outperforms BSUV-Net and other state-of-the-art BGS algorithms [Mandal et al., 2019, Mandal et al., 2021, Kim and Ha, 2020, Mandal and Vipparthi, 2020] with the help of several spatio-temporal data augmentations. We also introduce a real-time version of BSUV-Net 2.0 which still performs better than state-of-the-art methods and we propose a 4-fold cross-validation data split for CDNet-2014 for easier comparison of future algorithms. Finally, we demonstrate a strong generalization capacity of BSUV-Net 2.0 using cross-dataset

---

[1] This work was published in IEEE Access [Tezcan et al., 2021b]

[2] The source code of BSUV-Net 2.0 is publicly available at `github.com/ozantezcan/BSUV-Net-2.0`

evaluation on LASIESTA [Cuevas et al., 2016] in which the proposed model significantly outperforms the current state-of-the-art methods on a completely unseen dataset.

4.1 Spatio-Temporal Data Augmentations

In this section, we first introduce mathematical notation and then propose new spatio-temporal augmentations. For completeness, we include the illumination-difference augmentation proposed in BSUV-Net (Chapter 3). Figure 4·1 shows one example of each of the proposed augmentations.

4.1.1 Notation

We will use the notation for input-label pair used in BSUV-Net and introduced in Section 3.3. The input consists of $\mathbf{I_E}, \mathbf{I_R}, \mathbf{I_C} \in \mathbb{R}^{w \times h \times 4}$ and the corresponding label is denoted as $\mathbf{I_{FG}} \in \{0,1\}^{w \times h}$. Although the resolution of input images varies from video to video, it is beneficial to use a single resolution during training in order to leverage parallel processing of GPUs. Therefore, the first augmentation step we propose is *spatio-temporal cropping* that maps each video to the same spatial resolution. In the second step, we propose two additional augmentations that modify the video content but not the size.

In our two-step process, in the first step we use different cropping functions to compute $\widetilde{\mathbf{I}}_\mathbf{E}, \widetilde{\mathbf{I}}_\mathbf{R}, \widetilde{\mathbf{I}}_\mathbf{C} \in \mathbb{R}^{\widetilde{w} \times \widetilde{h} \times 4}$ and $\widetilde{\mathbf{I}}_\mathbf{FG} \in \{0,1\}^{\widetilde{w} \times \widetilde{h}}$ from $\mathbf{I_E}, \mathbf{I_R}, \mathbf{I_C}$ and $\mathbf{I_{FG}}$ where $\widetilde{w}, \widetilde{h}$ are the desired width and height after cropping. In the second step, we apply post-crop augmentations to compute $\widehat{\mathbf{I}}_\mathbf{E}, \widehat{\mathbf{I}}_\mathbf{R}, \widehat{\mathbf{I}}_\mathbf{C} \in \mathbb{R}^{\widetilde{w} \times \widetilde{h} \times 4}$ and $\widehat{\mathbf{I}}_\mathbf{FG} \in \{0,1\}^{\widetilde{w} \times \widetilde{h}}$ from $\widetilde{\mathbf{I}}_\mathbf{E}, \widetilde{\mathbf{I}}_\mathbf{R}, \widetilde{\mathbf{I}}_\mathbf{C}$ and $\widetilde{\mathbf{I}}_\mathbf{FG}$. Below, we explain these two steps in detail.

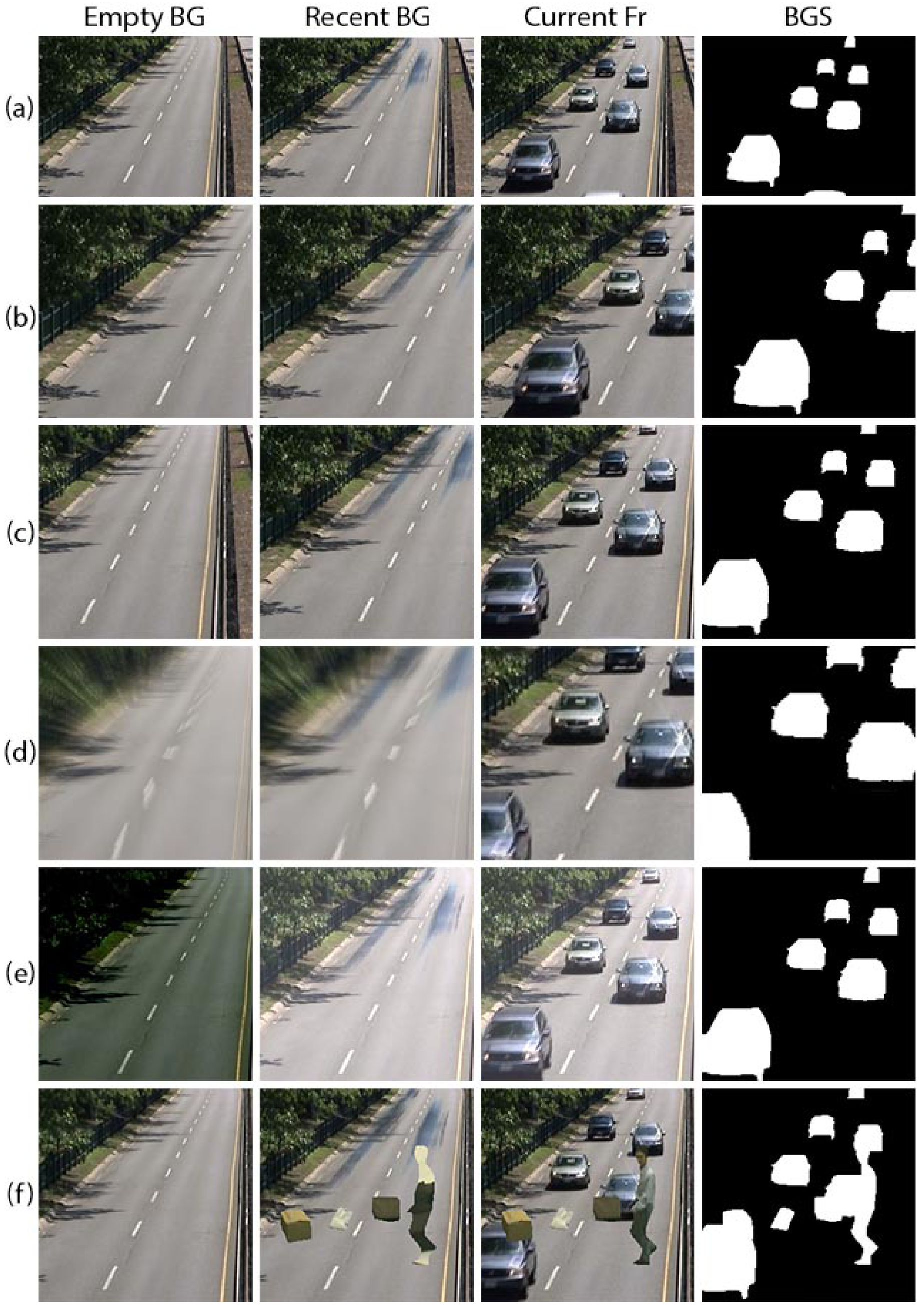


Figure 4·1: Image augmentation examples. Each row shows an example for one of the augmentations: (a) original input, (b) spatially-aligned crop, (c) randomly-shifted crop, (d) PTZ camera crop, (e) illumination difference, (f) intermittent-object addition.

4.1.2 Spatio-Temporal Crops

Here we describe 3 augmentation techniques to compute $\widetilde{\mathbf{I}}_{\mathbf{E}}, \widetilde{\mathbf{I}}_{\mathbf{R}}, \widetilde{\mathbf{I}}_{\mathbf{C}}, \widetilde{\mathbf{I}}_{\mathbf{FG}}$ from $\mathbf{I}_{\mathbf{E}}, \mathbf{I}_{\mathbf{R}}$, $\mathbf{I}_{\mathbf{C}}$, $\mathbf{I}_{\mathbf{FG}}$, each addressing a different BGS challenge. We begin by defining a cropping function, to be used in this section, as follows:

$$\mathcal{C}(\mathbf{I}, i, j, h, w) = \mathbf{I}\left[\left\lceil i - \tfrac{h}{2} \right\rceil : \left\lceil i + \tfrac{h}{2} \right\rceil, \left\lceil j - \tfrac{w}{2} \right\rceil : \left\lceil j + \tfrac{w}{2} \right\rceil, 1:4\right]$$

where i, j are the center coordinates, h, w are the height and width of the crop, $\lceil \cdot \rceil$ denotes the ceiling function and $a : b$ denotes the range of integer indices $a, a + 1, \ldots, b - 1$.

Spatially-Aligned Crop

This is an extension of the widely-used spatial cropping for individual images. Although this is straightforward, we provide a precise definition in order to clearly define steps in the subsequent sections.

The output of a spatially-aligned crop is defined follows:

$$\widetilde{\mathbf{I}}_k = \mathcal{C}(\mathbf{I}_k,\ i,\ j,\ \widetilde{h},\ \widetilde{w}) \quad \text{for all } k \in \{\mathbf{E},\ \mathbf{R},\ \mathbf{C},\ \mathbf{FG}\},$$

where i, j are randomly-selected spatial indices of the center of the crop. This formulation allows us to obtain a fixed-size, spatially-aligned crop from the input-label pair.

Randomly-Shifted Crop

One of the most challenging scenarios for BGS algorithms is camera jitter which results in random spatial shifts between consecutive video frames. However, since the variety of such videos is limited in public datasets, it is not trivial to learn the behavior of camera jitter using a data-driven algorithm. In order to address this, we introduce a new data augmentation method by simulating camera jitter. As a result, spatially-aligned inputs become randomly shifted. This is formulated as follows:

$$\widetilde{\mathbf{I}}_k = \mathcal{C}(\mathbf{I}_k,\ i_k,\ j_k,\ \widetilde{h},\ \widetilde{w}) \quad \text{for all } k \in \{\mathbf{E},\ \mathbf{R},\ \mathbf{C},\ \mathbf{FG}\},$$

where i_k, j_k are randomly-selected, but such that $i_C = i_{FG}$ and $j_C = j_{FG}$ to make sure that the current frame and foreground labels are aligned. By using different center spatial indices for background images and the current frame, we emulate camera jitter effect in the input.

PTZ Camera Crop

Another challenging BGS scenario is PTZ camera operation. While such videos are very common in surveillance, they form only a small fraction of public datasets. Therefore, we introduce another data augmentation technique specific to this challenge.

Since PTZ videos do not have a static empty background frame, BSUV-Net handled them differently than other categories (Section 3.8.2). Instead of empty and recent backgrounds, we proposed to use recent and more recent background, where the recent background was computed as the median of 100 preceding frames and the more recent background was computed as the median of 30 such frames. To simulate this kind of behavior, we introduce two types of PTZ camera crops: (i) zooming camera crop, (ii) moving camera crop.

The zooming camera crop is defined as follows:

$$\begin{aligned}
\widetilde{\mathbf{I}}_k &= \mathcal{C}(\mathbf{I}_k,\ i,\ j,\ \widetilde{h},\ \widetilde{w}) \quad \text{for all } k \in \{\mathbf{C},\ \mathbf{FG}\},\\
\widetilde{\mathbf{I}}_k &= \frac{1}{N^z} \sum_{n=0}^{N^z-1} \widetilde{\mathbf{I}}_k^n \quad \text{for all } k \in \{\mathbf{E},\ \mathbf{R}\},\ \text{where}\\
\widetilde{\mathbf{I}}_k^n &= \mathcal{R}\Big(\mathcal{C}\Big(\mathbf{I}_k,\ i,\ j,\ \widetilde{h}(1+nz_k),\ \widetilde{w}(1+nz_k)\Big),\ \widetilde{h},\ \widetilde{w}\Big)
\end{aligned}$$

where $z_{\mathbf{E}}, z_{\mathbf{R}}$ represent zoom factors for empty and recent backgrounds and N^z represents the number of zoomed in/out frames to use in averaging. In our experiments, we use $-0.1 < z_{\mathbf{E}},\ z_{\mathbf{R}} < 0.1$ and $5 < N^z < 15$ to simulate real-world camera zooming. $\mathcal{R}(\mathbf{I}, \widetilde{h}, \widetilde{w})$ is an image resizing function that changes the resolution of $\mathbf{I}$ to $(\widetilde{w}, \widetilde{h})$ using bilinear interpolation. Note, that using positive values for z_k simulates zooming-in whereas using negative values simulates zooming-out. Figure 4·1(d) shows an example of zoom-in.

Similarly, the moving camera crop is defined as follows:

$$
\begin{aligned}
\widetilde{\mathbf{I}}_k &= \mathcal{C}(\mathbf{I}_k,\ i,\ j,\ \widetilde{h},\ \widetilde{w}) \quad \text{for all } k \in \{\mathbf{C},\ \mathbf{FG}\}, \\
\widetilde{\mathbf{I}}_k &= \frac{1}{N_k^m} \sum_{n=0}^{N_k^m - 1} \widetilde{\mathbf{I}}_k^n \quad \text{for all } k \in \{\mathbf{E},\ \mathbf{R}\},\ \text{where} \\
\widetilde{\mathbf{I}}_k^n &= \mathcal{C}(\mathbf{I}_k,\ i + np,\ j + nq,\ \widetilde{h},\ \widetilde{w})
\end{aligned}
$$

where p, q are the vertical and horizontal shift amounts per frame and $N_{\mathbf{E}}^m, N_{\mathbf{R}}^m$ represent the number of empty and recent moving background crops to use for averaging. This simulates camera pan and tilt. In our experiments, we use $-5 < p,\ q < 5$ and $5 < N_{\mathbf{E}}^m, N_{\mathbf{R}}^m < 15$ to simulate real-world camera movements.

4.1.3 Post-Crop Augmentations

In this section, we propose several content-modifying augmentation techniques to compute $\widehat{\mathbf{I}}_{\mathbf{E}}, \widehat{\mathbf{I}}_{\mathbf{R}}, \widehat{\mathbf{I}}_{\mathbf{C}}, \widehat{\mathbf{I}}_{\mathbf{FG}}$ from $\widetilde{\mathbf{I}}_{\mathbf{E}}, \widetilde{\mathbf{I}}_{\mathbf{R}}, \widetilde{\mathbf{I}}_{\mathbf{C}}, \widetilde{\mathbf{I}}_{\mathbf{FG}}$. These augmentations can be applied after any one of the spatio-temporal crop augmentations.

Illumination Difference

Illumination variations are common, especially in long videos, for example due to changes in natural light or lights being turned on/off. We introduced a temporal data augmentation technique in BSUV-Net to handle illumination changes with the goal of increasing the network's generalization capacity for unseen videos (see Section 3.4). We use this augmentation here as well, formulated in the current notation as follows:

$$\widehat{\mathbf{I}}_k[i,\ j,\ c] = \widetilde{\mathbf{I}}_k[i,\ j,\ c] + \mathbf{d}_k[c] \text{ for } k \in \{\mathbf{E},\ \mathbf{R},\ \mathbf{C}\}, c = 1, 2, 3$$

where $\mathbf{d}_\mathbf{E}, \mathbf{d}_\mathbf{R}, \mathbf{d}_\mathbf{C} \in \mathbb{R}^3$ represent illumination offsets applied to RGB channels of the input images. This augmentation does not take the scene context into account and, therefore, can produce non-realistic results. For example, we keep the shadows intact although they strongly depend on illumination. Since the aim of data augmentations is to increase the robustness of the network to challenging scenarios, we do not attempt to make the augmented examples realistic.

Intermittent-Object Addition

Another challenge for BGS are scenarios when objects enter a scene but then stop and remain static for a long time. Even very successful BGS algorithms, after some time, predict these objects as part of the background for they rely on recent frames to estimate the background model. BSUV-Net overcomes this challenge by using inputs from multiple time scales, however it still underperforms on videos with intermittently-static objects. To address this, we propose another spatio-temporal data augmentation specific to this challenge.

We use a masking-based approach for intermittently-static objects as follows. In addition to the cropped inputs $\widetilde{\mathbf{I}}_\mathbf{E}, \widetilde{\mathbf{I}}_\mathbf{R}, \widetilde{\mathbf{I}}_\mathbf{C}, \widetilde{\mathbf{I}}_\mathbf{FG}$, we also use cropped inputs from videos with intermittently-static objects defined as $\widetilde{\mathbf{I}}^\mathbf{IO}_\mathbf{E}, \widetilde{\mathbf{I}}^\mathbf{IO}_\mathbf{R}, \widetilde{\mathbf{I}}^\mathbf{IO}_\mathbf{C} \in \mathbb{R}^{\widetilde{w} \times \widetilde{h} \times 4}$ and $\widetilde{\mathbf{I}}^\mathbf{IO}_\mathbf{FG} \in \{0, 1\}^{\widetilde{w} \times \widetilde{h}}$. We copy foreground pixels from the intermittently-static input and paste them into the original input to synthetically create an intermittent object.

This can be formulated as follows:

$$
\begin{aligned}
\widehat{\mathbf{I}}_{\mathbf{E}} &= \widetilde{\mathbf{I}}_{\mathbf{E}} \\
\widehat{\mathbf{I}}_{k} &= \widetilde{\mathbf{I}}^{\mathbf{IO}}_{\mathbf{FG}} \odot \widetilde{\mathbf{I}}^{\mathbf{IO}}_{k} + (1 - \widetilde{\mathbf{I}}^{\mathbf{IO}}_{\mathbf{FG}}) \odot \widetilde{\mathbf{I}}_{k} \text{ for } k \in \{\mathbf{C}, \mathbf{R}\}, \\
\widehat{\mathbf{I}}_{\mathbf{FG}} &= \widetilde{\mathbf{I}}^{\mathbf{IO}}_{\mathbf{FG}} + (1 - \widetilde{\mathbf{I}}^{\mathbf{IO}}_{\mathbf{FG}}) \odot \widetilde{\mathbf{I}}_{\mathbf{FG}}
\end{aligned}
$$

where $\odot$ denotes Hadamard (element-wise) product. Figure 4·1(f) shows an example of intermittent object addition. This augmentation pastes the foreground-object pixels from “intermittent object motion” videos into the original inputs without attempting to create a realistic frame (e.g., oversized cars can appear on top of the buildings). We believe this can still help the network focus on intermittently-static motion without focusing on the scene context. Also, note that this augmentation requires prior knowledge of examples with intermittently-static objects which can be found in some public datasets.

4.1.4 Combining Spatio-Temporal Augmentations

While the augmentations defined above can all be used by themselves to improve the BGS performance on related categories, combining multiple or even all of them might result in a better algorithm for a general unseen video of which the category is unknown. However, combining the crop algorithms is not trivial since it is not practical to apply more than one crop function to a single input. Thus, we use online augmentation, where we randomly augment every input while forming mini-batches. The augmentation steps are as follows:

1. randomly select one of the spatial crop augmentations and apply it to the input,
2. apply illumination-change augmentation using randomized illumination values,
3. apply intermittent object addition to $p\%$ of the inputs.

Clearly, a different combination of augmentations will be applied to the same input in different epochs. We hope this will significantly increase the variety of training examples and, consequently, the generalization capacity of our network.

4.2 Video-Agnostic Evaluation Strategy for Supervised Algorithms

In Chapter 3, we used a complicated cross-validation scheme to evaluate the performance of BSUV-Net on CDNet-2014 [Goyette et al., 2014] (see Section 3.7). In this chapter, we introduce a simpler and more intuitive 4-fold cross-validation strategy for CDNet-2014. We grouped all videos in the dataset and each category into 4 folds as evenly as possible (Table 4.1). The proposed *video-agnostic* evaluation strategy is to train any supervised BGS algorithm on three of the folds and test on the remaining fold and replicate the same process for all 4 combinations. This approach will provide results on the full CDNet-2014 dataset which can be uploaded to the evaluation server to compare against state-of-the-art algorithms. We believe this cross-validation strategy will be very beneficial for the evaluation of future BGS algorithms.

4.3 Experimental Results

4.3.1 Dataset and Evaluation Details

We evaluate the performance of our algorithm on CDNet-2014 [Goyette et al., 2014] using the evaluation strategy described in Section 4.2. In order to better understand the performance of BSUV-Net 2.0 on unseen videos, we also performed a *cross-dataset* evaluation by training our model on CDNet-2014 and testing it on a completely different dataset, LASIESTA [Cuevas et al., 2016]. LASIESTA is an extensive BGS dataset which includes 24 different videos from various indoor and outdoor scenarios. It includes a “Simulated Motion” category that is comprised of fixed-camera videos

Table 4.1: Proposed sets for 4-fold cross-validation on CDNet-2014.

Category	**Video**	$\mathbf{S_1}$	$\mathbf{S_2}$	$\mathbf{S_3}$	$\mathbf{S_4}$
Baseline	highway	✓			
	pedestrians		✓		
	office			✓	
	PETS2006				✓
Bad weather	blizzard	✓			
	skating		✓		
	wetSnow			✓	
	snowFall				✓
Intermittent object motion	sofa	✓			
	winterDriveway		✓		
	parking			✓	
	abandonedBox			✓	
	streetLight				✓
	tramstop				✓
Low framerate	port 0.17fps	✓			
	tramCrossroad 1fps		✓		
	tunnelExit 0.35fps			✓	
	turnpike 0.5fps				✓
PTZ	continuousPan	✓			
	intermittentPan		✓		
	zoomInZoomOut			✓	
	twoPositionPTZCam				✓
Thermal	corridor	✓			
	lakeSide	✓			
	library		✓		
	diningRoom			✓	
	park				✓

Category	**Video**	$\mathbf{S_1}$	$\mathbf{S_2}$	$\mathbf{S_3}$	$\mathbf{S_4}$
Camera jitter	badminton	✓			
	traffic		✓		
	boulevard			✓	
	sidewalk				✓
Shadow	copyMachine	✓			
	busStation		✓		
	cubicle			✓	
	peopleInShade			✓	
	bungalows				✓
	backdoor				✓
Dynamic background	overpass	✓			
	fountain02	✓			
	fountain01		✓		
	boats		✓		
	canoe			✓	
	fall				✓
Night videos	bridgeEntry	✓			
	busyBoulvard	✓			
	tramStation		✓		
	winterStreet		✓		
	fluidHighway			✓	
	streetCornerAtNight				✓
Turbulence	turbulence0	✓			
	turbulence1		✓		
	turbulence2			✓	
	turbulence3				✓

that are post-processed to mimic camera pan, tilt and jitter [Cuevas et al., 2016].

4.3.2 Training Details

In order to train BSUV-Net 2.0, we use similar parameters to the ones we used for BSUV-Net. We use ADAM optimizer with a learning rate of 10^{-4}, $\beta_1 = 0.9$, and $\beta_2 = 0.99$, and the mini-batch size of 8 and 200 epochs. These parameters are the same for each of the four cross-validation folds. As the empty background frame, we use manually-selected frames and as the recent background – the median of the preceding 100 frames.

In terms of spatio-temporal data augmentations, we use an online approach to randomly change the parameters under the following constraints. The random pixel shift between inputs is sampled from $\mathcal{U}(0, 5)$ where $\mathcal{U}(a, b)$ denotes uniform random variable between a to b. The zoom-in ratios are sampled from $\mathcal{U}(0, 0.02)$ and $\mathcal{U}(0, 0.04)$ for the recent and empty backgrounds, respectively, while the zoom-out ratios are sampled from $\mathcal{U}(-0.02, 0)$ and $\mathcal{U}(-0.04, 0)$. We use $N^z = 10$. The horizontal pixel shift for the moving-camera augmentation is sampled from $\mathcal{U}(0, 5)$ with $N^m_{\mathbf{E}} = 20$ and $N^m_{\mathbf{R}} = 10$. We perform no vertical-shift augmentation since CDNet-2014 does not include any videos with vertical camera movement. For illumination change, assuming $[0, 1]$ as the range of pixel values, we use $\mathbf{d_R}[k] = \mathbf{d_C}[k] = I + I_k$ where $I \sim \mathcal{N}(0, 0.1^2)$ and $I_k \sim \mathcal{N}(0, 0.04^2)$ for $k \in \{1, 2, 3\}$. Similarly, $\mathbf{d_E}[k] = \mathbf{d_C}[k] + I^E + I^E_k$ where $I^E \sim \mathcal{N}(0, 0.1^2)$ and $I^E_k \sim \mathcal{N}(0, 0.04^2)$ for $k \in \{1, 2, 3\}$. Lastly, for intermittent object addition, we always use the "intermittent object motion" inputs from the current training set and apply this augmentation to $p = 10\%$ of the inputs only. During inference, binary maps are obtained by thresholding the network output at $\theta = 0.5$.

Table 4.2: Comparison of different spatio-temporal augmentations on CDNet-2014 based on *F-score*. SAC: spatialy-aligned crop, RSC: randomly-shifted crop, PTZ: PTZ camera crop, ID: illumination difference, IOA: intermittent object addition. The values in boldface font show the best performance for each category.

SAC	RSC	PTZ	ID	IOA	Bad weather	Low framerate	Night	PTZ	Thermal	Shadow
✓					0.9442	0.7886	0.6982	0.6564	0.8960	0.9848
✓	✓				0.9439	**0.8217**	0.6941	0.6304	0.8854	0.9818
✓		✓			0.9315	0.7961	0.6557	**0.6815**	0.8905	0.9795
✓			✓		**0.9489**	0.7606	**0.7605**	0.6579	**0.9024**	**0.9855**
✓				✓	0.9456	0.7550	0.7233	0.6383	0.8997	0.9836
✓	✓	✓	✓	✓	0.9272	0.8114	0.6841	0.6725	0.8960	0.9811

SAC	RSC	PTZ	ID	IOA	Int. obj. motion	Camera jitter	Dynamic backgr.	Base-line	Turbu-lence	Overall
✓					0.7732	0.8237	0.8517	0.9878	0.7285	0.8303
✓	✓				0.7620	0.9043	0.8745	0.9865	0.7354	0.8382
✓		✓			0.7458	0.8999	0.8674	0.9838	0.7409	0.8339
✓			✓		0.7503	0.8270	0.8364	0.9874	0.7341	0.8319
✓				✓	**0.9312**	0.8359	0.8709	**0.9883**	0.7023	0.8431
✓	✓	✓	✓	✓	0.8489	**0.9163**	**0.8848**	0.9834	**0.8056**	**0.8556**

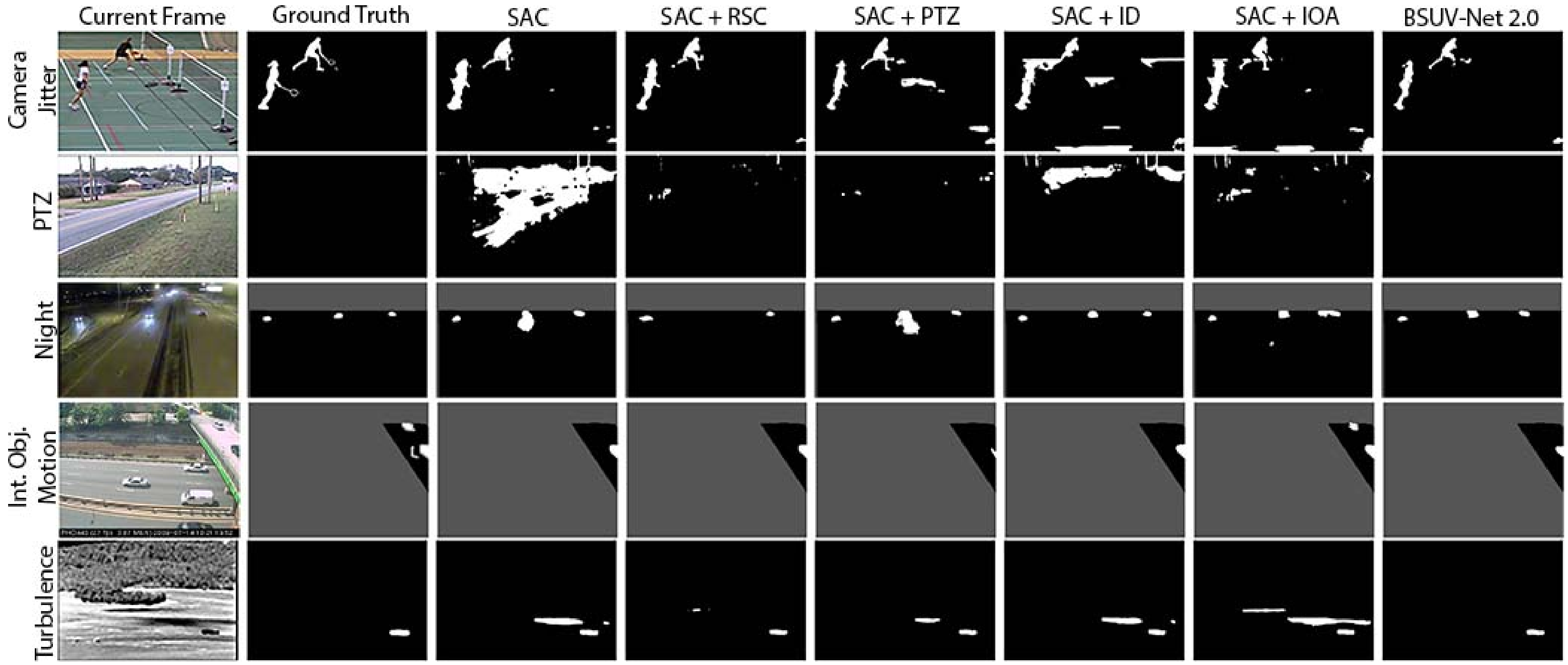


Figure 4·2: Visual comparison of different spatio-temporal augmentations on sample frames. Columns 1 and 2 show the current frame and its ground truth. The subsequent columns show predictions of our network trained with augmentations listed at the top of each column. The last column shows predictions of BSUV-Net 2.0 trained using a combination of all proposed augmentations. The dark-gray areas represent pixels outside of CDNet-2014 regions of interest.

4.3.3 Ablation Study

We assess the impact of each spatio-temporal data augmentation method defined in Section 4.1. As the baseline network, we use BSUV-Net with spatially-aligned crop augmentation and random Gaussian noise sampled from $\mathcal{N}(0, 0.01^2)$, but without the "illumination difference" augmentation. We evaluate the proposed spatio-temporal augmentations against this baseline by including the spatially-aligned crop among spatial crop augmentations, as explained in Section 4.1.4. In PTZ camera crop, for each input, we randomly select one of the following: zooming in, zooming out, moving right or moving left. Table 4.2 shows *F-score*s for each category of CDNet-2014 computed locally[3] for frames with publicly-available ground truth. We report the median of results for every 5^{th} epoch between 150^{th} and 200^{th} epochs to disregard small fluctuations in the learning process. We perform this across all four splits proposed in Table 4.1.

Figure 4·2 presents visual impact of these augmentations on 5 videos. It can be observed that each augmentation type significantly improves the performance on the related categories (randomly shifted crop – on "Camera jitter", PTZ camera crop – on "PTZ", illumination difference – on "Shadow", intermittent object addition – on "Intermittent object motion"), but combining all augmentations decreases the performance significantly on some categories (e.g., "Night videos" and "Intermittent object motion"). We believe this is due to trade-offs between the effects of different augmentations. For example, when a static background object starts moving it should be labeled as foreground, but a network trained with a randomly-shifted crop augmentation can confuse this input with an input from the "Camera jitter" category and continue labeling the object as background. Still, the overall performance (last col-

---

[3] We provide only locally-computed results because if the results of the ablation study were uploaded to the CDNet-2014 evaluation server, they would have not been made public since they all come from the same algorithm. Moreover, this simplifies corroboration of our results by independent parties by not requiring uploads to the evaluation server.

umn in Table 4.2) of BSUV-Net 2.0 that uses all augmentations handily outperforms the overall performance for individual augmentations.

4.3.4 Fast BSUV-Net 2.0

Since BGS is often applied as a pre-processing step in real-time video processing applications, computation speed is critical. As discussed in Section 3.8.5, one of the main bottlenecks of BSUV-Net is the computation of FPM for each channel – it decreases the overall computation speed significantly. On the other hand, either removing the FPM channel or predicting BGS by thresholding the FPM channel alone decreases the performance to values that are lower than that of some unsupervised algorithms. In this work, we show that the performance of our model, even without the FPM channel but with the proposed augmentations, is better than the current state-of-the-art. We call this version of BSUV-Net 2.0, which uses 9 instead of 12 channels on input, Fast BSUV-Net 2.0. Table 4.3 shows a speed and performance comparison of the two versions. Clearly, while Fast BSUV-Net 2.0 has lower performance, it can be used in real-time applications at 320 × 240 spatial resolution, which is very similar to the resolution used in training. For higher-resolution videos, one can easily feed decimated frames into Fast BSUV-Net 2.0 and interpolate the resulting BGS predictions to the original resolution.

Table 4.3: Efficiency vs performance trade-off for BSUV-Net 2.0 on CDNet-2014. FPS is calculated using PyTorch 1.3 implementation on a node with single Nvidia Tesla P100 GPU.

	Re	Pr	F_1	FPS	
				320 × 240	640 × 480
BSUV-Net 2.0	0.85	0.89	0.86	~ 6	~ 2.5
Fast BSUV-Net 2.0	0.84	0.84	0.81	~ 29	~ 13

Table 4.4: Official comparison of top BGS algorithms evaluated on **unseen videos** from CDNet-2014.

Method	R	R_{cat}	Re	Sp	PWC	Pr	F_1
BSUV-Net 2.0	**7.71**	**7.73**	0.8136	**0.9979**	**0.7614**	**0.9011**	**0.8387**
Fast BSUV-Net 2.0	7.86	10.36	0.8181	0.9956	0.9054	0.8425	0.8039
BSUV-Net + SemanticBGS	9.57	14.27	0.8179	0.9944	1.1326	0.8319	0.7986
IUTIS-5 + SemanticBGS [Braham et al., 2017]	9.71	11.91	0.7890	0.9961	1.0722	0.8305	0.7892
IUTIS-5 [Bianco et al., 2017]	12.14	10.91	0.7849	0.9948	1.1986	0.8087	0.7717
BSUV-Net	9.71	14.00	**0.8203**	0.9946	1.1402	0.8113	0.7868
SWCD [Işık et al., 2018]	16.43	20.00	0.7839	0.9930	1.3414	0.7527	0.7583
WisenetMD [Lee et al., 2018]	17.29	15.82	0.8179	0.9904	1.6136	0.7535	0.7668
PAWCS [St-Charles et al., 2015a]	14.71	16.09	0.7718	0.9949	1.1992	0.7857	0.7403
FgSegNet v2 [Lim and Keles, 2018a]	45.57	45.09	0.5119	0.9411	7.3507	0.4859	0.3715

Table 4.5: Official comparison of top BGS algorithms according to the per-category *F-score* on **unseen videos** from CDNet-2014.

Method	Bad weather	Low framerate	Night	PTZ	Thermal	Shadow
BSUV-Net 2.0	0.8844	**0.7902**	0.5857	**0.7037**	**0.8932**	0.9562
Fast BSUV-Net 2.0	**0.8909**	0.7824	0.6551	0.5014	0.8379	0.8890
BSUV-Net + SemanticBGS	0.8730	0.6788	0.6815	0.6562	0.8455	**0.9664**
IUTIS-5 + SemanticBGS	0.8260	0.7888	0.5014	0.5673	0.8219	0.9478
IUTIS-5	0.8248	0.7743	0.5290	0.4282	0.8303	0.9084
BSUV-Net	0.8713	0.6797	**0.6987**	0.6282	0.8581	0.9233
RTSS	0.8662	0.6771	0.5295	0.5489	0.8510	0.9551
WisenetMD	0.8616	0.6404	0.5701	0.3367	0.8152	0.8984
FgSegNet v2	0.3277	0.2482	0.2800	0.3503	0.6038	0.5295

Method	Int. obj. motion	Camera jitter	Dynamic backgr.	Base-line	Turbu-lence	Overall
BSUV-Net 2.0	0.8263	**0.9004**	0.9057	0.9620	0.8174	**0.8387**
Fast BSUV-Net 2.0	**0.9016**	0.8828	0.7320	**0.9694**	0.7998	0.8039
BSUV-Net + SemanticBGS	0.7601	0.7788	0.8176	0.9640	0.7631	0.7986
IUTIS-5 + SemanticBGS	0.7878	0.8388	**0.9489**	0.9604	0.6921	0.7892
IUTIS-5	0.7296	0.8332	0.8902	0.9567	0.7836	0.7717
BSUV-Net	0.7499	0.7743	0.7967	0.9693	0.7051	0.7868
RTSS	0.7864	0.8396	0.9325	0.9597	0.7630	0.7917
WisenetMD	0.7264	0.8228	0.8376	0.9487	**0.8304**	0.7535
FgSegNet v2	0.2002	0.4266	0.3634	0.6926	0.0643	0.3715

4.3.5 Comparison with State of the Art

Table 4.4 shows the performance of BSUV-Net 2.0 and Fast BSUV-Net 2.0 compared to state-of-the-art BGS algorithms that are designed for and tested on unseen videos. We did not include the results of *video-* or *video-group-optimized* algorithms since it is not fair to compare them against *video-agnostic* algorithms. This table shows official results computed by CDNet-2014 evaluation server[4], so the results of our models slightly differ from those in Tables 4.2 and 4.3 (different ground-truth frames). We compare BSUV-Net 2.0 with some of the top-performing *video-agnostic algorithms* reported by this server. RTSS [Zeng et al., 2019a], 3DCD [Mandal et al., 2021], 3DFR [Mandal et al., 2019], ChangeDet [Mandal and Vipparthi, 2020] and Kim *et al.* [Kim and Ha, 2020] are not included in this table since their results are not reported. BSUV-Net 2.0 outperforms all state-of-the-art algorithms by at least ∼5% in terms of *F-score* (0.8387 versus 0.7986 in Tables 4.4, 4.5). Fast BSUV-Net 2.0 also outperforms all state-of-the-art algorithms while being ∼5 times faster than BSUV-Net 2.0 during inference (Table 4.3). Table 4.5 shows the comparison of F_1 results for each category. This table includes RTSS using results reported in the paper [Zeng et al., 2019a]. In 7 out of 11 categories, either BSUV-Net 2.0 or Fast BSUV-Net 2.0 achieve the best performance, including most of the categories that we designed the augmentations for (an exception is the "Night videos" category). However, note that the best-performing algorithm in the "Night videos" category is BSUV-Net which uses only the illumination-difference augmentation. Thus, it focuses on videos with illumination differences such as night videos.

Figure 4·3 qualitatively compares the performance of BSUV-Net 2.0 with state-of-the-art video-agnostic BGS algorithms on several videos from CDNet-2014. BSUV-Net 2.0 clearly produces the best visual results in a variety of scenarios. Results for

---

[4] `http://changedetection.net/`

Table 4.6: *F-score* comparison of BSUV-Net 2.0 with *video-agnostic* supervised BGS algorithms that are not reported in `changedetection.net`. Each column shows test performance of the algorithm by using the training/testing split provided in the respective paper.

	Training/Testing Split			
Method	3DCD	3DFR	ChangeDet	Kim
BSUV-Net 2.0	**0.89**	**0.89**	**0.90**	**0.88**
Fast BSUV-Net 2.0	0.84	0.84	0.86	**0.88**
3DCD [Mandal et al., 2021]	0.86	-	-	-
3DFR [Mandal et al., 2019]	-	0.86	-	-
ChangeDet [Mandal and Vipparthi, 2020]	-	-	0.84	-
Kim [Kim and Ha, 2020]	-	-	-	0.86

"Camera jitter" and "PTZ" categories show the effectiveness of BSUV-Net 2.0 in removing false positives resulting from camera motion. In the example from "Intermittent object motion" category, the car on the left is starting to back-up from the driveway and most of the algorithms produce false positives at the location where the car was parked whereas BSUV-Net 2.0 successfully eliminates these false positives. Results for the "Dynamic background" category show that BSUV-Net 2.0 is very effective in accurately delineating the boundary between foreground objects and the background.

As discussed in Section 2.1.3, 3DCD [Mandal et al., 2021], 3DFR [Mandal et al., 2019], ChangeDet [Mandal and Vipparthi, 2020] and Kim *et al.* [Kim and Ha, 2020] are also among the best *video-agnostic* supervised algorithms, however each reports performance on a different subset of CDNet-2014, with the algorithm trained on the remaining videos. Table 4.6 shows the comparison of BSUV-Net 2.0 with these algorithms using the training/testing splits provided in respective papers in each column. BSUV-Net 2.0 clearly outperforms all four competitors, while Fast BSUV-Net 2.0 beats 2 out of 4, and does so with real-time performance.

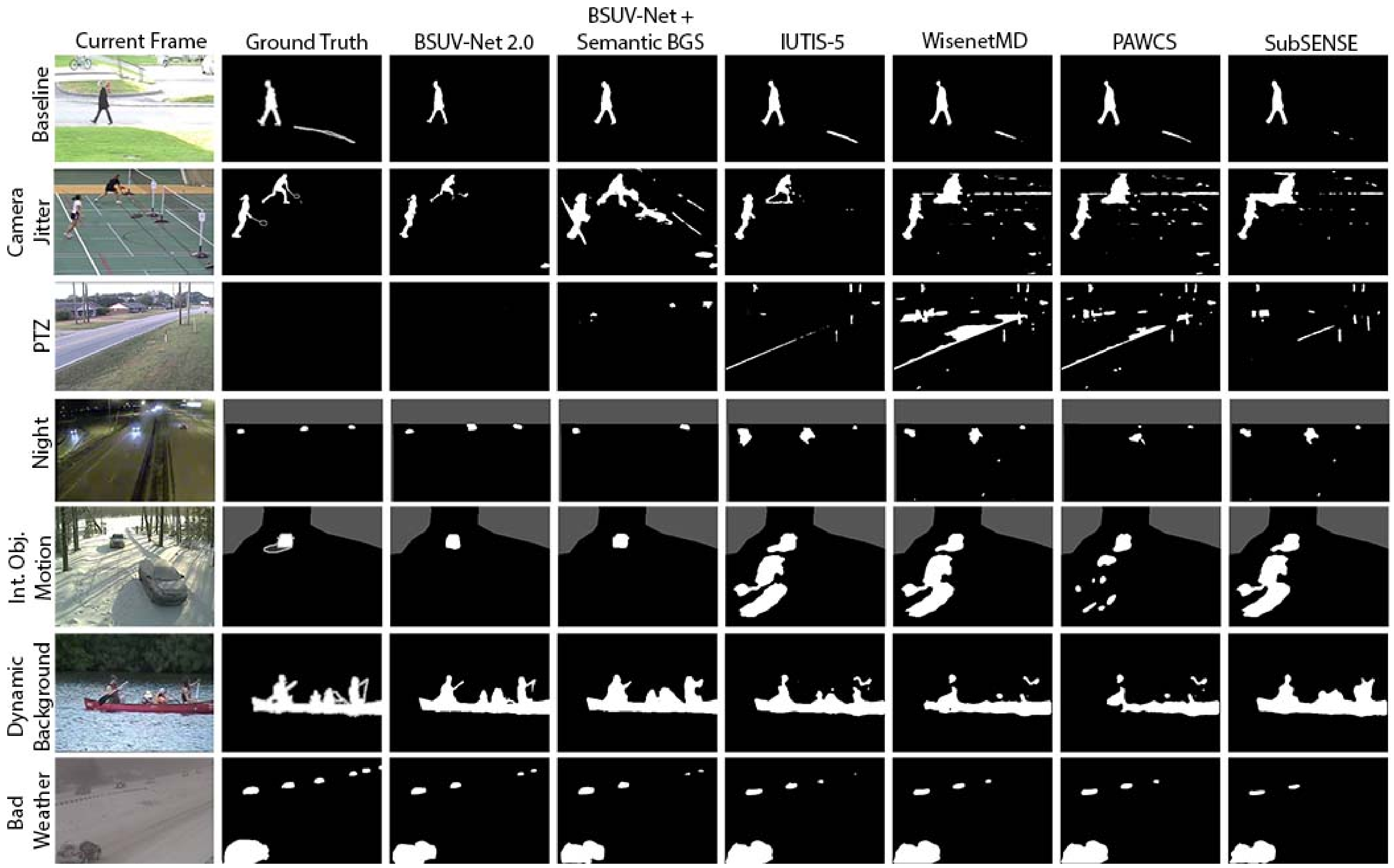


Figure 4·3: Qualitative comparison of top BGS algorithms on sample frames from different categories of CDNet-2014.

4.3.6 Cross-Dataset Evaluation

In this section, we perform a cross-dataset evaluation to show the generalization capacity of BSUV-Net 2.0. We train BSUV-Net 2.0 using CDNet-2014 videos from $\mathbf{S_2}$, $\mathbf{S_3}$, $\mathbf{S_4}$ sets shown in Table 4.1 and use $\mathbf{S_1}$ as a validation set to select the best performing epoch. Then, we evaluate the results on a completely different dataset, LASIESTA [Cuevas et al., 2016][5]. Table 4.7 shows the comparison of BSUV-Net 2.0 with top-performing unsupervised algorithms reported in [Cuevas et al., 2016]. Since the authors reported results only for categories of LASIESTA recorded with static cameras, we report results only on these categories. Clearly, BSUV-Net 2.0 outperforms its competitors on a completely unseen dataset by a significant margin.

In [Mandal et al., 2021], Mandal *et al.* performed a *video-agnostic* evaluation of some supervised learning algorithms by training with 10 of the LASIESTA videos and evaluating on 10 unseen videos from LASIESTA. Table 4.8 shows a comparison of BSUV-Net 2.0 with unseen video performance of the algorithms reported in [Mandal et al., 2021]. We show the results of BSUV-Net 2.0 trained with two different datasets. BSUV-Net 2.0 row shows the results of cross-dataset training whereas BSUV-Net 2.0* row shows the results of using the same training set as used in [Mandal et al., 2021], for a fair comparison. BSUV-Net 2.0 achieves significantly better results than state of the art even if the training set does not include any videos from LASIESTA. Since we train BSUV-Net 2.0* with videos from LASIESTA, it performs even better than BSUV-Net 2.0. This shows that the proposed spatio-temporal data augmentations are not specific to CDNet-2014 and can be very effective on other datasets as well. Note that the performance of BSUV-Net 2.0 is significantly better than that of BSUV-Net 2.0* on OSN-2, an outdoor video recorded in heavy snow. This is due to the fact

---

[5] The empty backgrounds of LASIESTA videos are computed automatically as the median of all frames in the video. The recent backgrounds are computed similarly to CDNet-2014, as the median of previous 100 frames.

Table 4.7: Per-category *F-score* comparison of the cross-dataset performance of BSUV-Net 2.0 with the top-performing unsupervised BGS algorithms on LASIESTA

Method	ISI	ICA	IOC	IIL	IMB	IBS	OCL	ORA	OSN	OSU	Overall
BSUV-Net 2.0	0.92	0.68	**0.96**	**0.88**	0.81	**0.77**	0.93	**0.94**	**0.84**	0.79	**0.85**
Cuevas [Berjón et al., 2018]	0.88	0.84	0.78	0.65	**0.93**	0.66	0.93	0.87	0.78	0.72	0.81
Haines [Haines and Xiang, 2014]	0.89	**0.89**	0.92	0.85	0.84	0.68	0.83	0.89	0.17	0.86	0.78
Maddalena2 [Maddalena and Petrosino, 2012]	**0.95**	0.86	0.95	0.21	0.91	0.40	**0.97**	0.90	0.81	**0.88**	0.78
Maddalena1 [Maddalena and Petrosino, 2008]	0.87	0.85	0.91	0.61	0.76	0.42	0.88	0.84	0.58	0.80	0.75

that the training videos of LASIESTA do not include a heavy-snow video, however the training set of CDNet-2014 does. This also shows the importance of scene variety in the training dataset. Both Tables 4.7 and 4.8 clearly show that BSUV-Net 2.0 is not specific to a dataset that it was trained on, but can successfully predict BGS of an unseen video.

In addition to the videos reported in Table 4.7, LASIESTA includes several videos that are either recorded with a moving camera or post-proccessed to look like they were recorded with a moving camera. We group these videos under 4 categories:

1. *Indoor pan & tilt videos* (IMC-1, ISM-1, ISM-2, ISM-3),
2. *Outdoor pan & tilt videos* (OMC-1, OSM-1, OSM-2, OSM-3),
3. *Indoor jitter videos* (IMC-2, ISM-4, ..., ISM-12),
4. *Outdoor jitter videos* (OMC-2, OSM-4, ..., OSM-12).

Table 4.9 shows the *F-score* comparison of BSUV-Net 2.0 trained with different combinations of spatio-temporal data augmentations on these 4 categories. As expected, the randomly shifted crop augmentation achieves the best performance for videos with camera jitter whereas the PTZ augmentation achieves the best results for PTZ category. This further shows that the impact of spatio-temporal data augmentations is generalizable to different datasets.

4.3.7 In-the-Wild Results

In Sections 4.3.5 and 4.3.6, we showed that BSUV-Net 2.0 significantly outperforms its competitors on two of the largest BGS datasets, CDNet-2014 and LASIESTA. Although these datasets cover a wide range of challenging scenarios such as moving cameras, thermal videos, dynamic background, etc., their video clips are generally short (typically, less than 5 minutes) and each clip focuses on a single challenge.

Table 4.8: Per-video *F-score* comparison of BSUV-Net 2.0 with the top-performing supervised BGS algorithms on unseen videos from LASIESTA. BSUV-Net 2.0 is trained only on CDNet-2014, as explained in Section 4.3.6. BSUV-Net 2.0* is trained on videos from LASIESTA that are not from the test set as described in [Mandal et al., 2021].

Method	ISI-2	ICA-2	IOC-2	IIL-2	IMB-2	IBS-2	OCL-2	ORA-2	OSN-2	OSU-2	Overall
BSUV-Net 2.0	0.89	0.60	0.95	0.89	0.76	0.69	0.89	0.93	**0.70**	0.91	0.82
BSUV-Net 2.0*	**0.98**	**0.99**	**0.97**	**0.97**	**0.88**	**0.95**	**0.97**	**0.97**	0.55	**0.95**	**0.92**
3DCD [Mandal et al., 2021]	0.86	0.49	0.93	0.85	0.79	0.87	0.87	0.87	0.49	0.83	0.79
FgSegNet v2 [Lim and Keles, 2018b]	0.53	0.58	0.25	0.41	0.63	0.25	0.54	0.54	0.05	0.29	0.41
FgSegNet-M [Lim and Keles, 2018a]	0.56	0.55	0.65	0.42	0.56	0.19	0.28	0.18	0.01	0.33	0.37

Table 4.9: *F-score* comparison of cross-dataset performance of different spatio-temporal augmentations on *Moving camera* and *Simulated motion* videos of LASIESTA. SAC: spatialy-aligned crop, RSC: randomly-shifted crop, PTZ: PTZ camera crop.

SAC	RSC	PTZ	Indoor pan & tilt	Outdoor pan & tilt	Indoor jitter	Outdoor jitter
✓			0.48	0.56	0.81	0.75
✓	✓		0.52	0.42	**0.88**	**0.85**
✓		✓	**0.58**	**0.58**	0.84	0.50

In most real-world applications (e.g., city-traffic surveillance), BGS predictions need to be computed continuously (24/7). Since the scene composition (e.g., background objects) is likely to change over time, an application of BSUV-Net 2.0 would require a periodic update of the empty-background frame. Furthermore, the algorithm's performance "in the wild" will be affected by camouflage, crowded scenes and presence of small objects, all insufficiently represented in public BGS datasets.

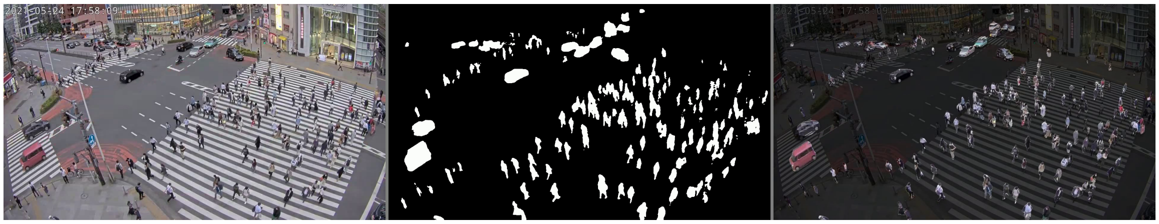
In order to evaluate BSUV-Net 2.0 performance *in the wild*, we used a live recording of street crossing in Tokyo, Japan (publicly available on YouTube[6]). We applied BSUV-Net 2.0 to long clips (several hours) taken from that live recording either during the day or at night. The empty background and recent background frames were computed for each current frame as follows:

- the empty background is updated hourly by using the average of all frames captured during the preceding 5 minutes,
- the recent background is updated continuously using the average of the preceding 100 frames.

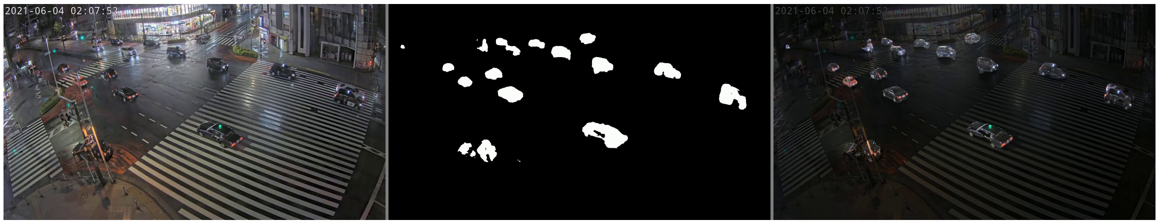
Since, obviously, no ground-truth BGS annotations are available in this case, we only show visual results (Figure 4·4). Clearly, the performance of BSUV-Net

---

[6] `youtu.be/RQA5RcIZlAM`



(a) Day-time frame



(b) Night-time frame

Figure 4·4: Sample results of in-the-wild application of BSUV-Net 2.0. The videos have been captured by a live camera during the day and at night to test different illumination conditions. Shown are the input frame (left column), foreground predictions computed by BSUV-Net 2.0 (middle column) and a combination of the input frame and foreground predictions (right column) – darker pixels represent the background while brighter ones show the foreground.

2.0 is promising. In the day-time example (Figure 4·4a), it is able to successfully detect almost all people and cars. Although this video has several challenges, such as intermittently-static objects (some of the cars at the top are waiting to turn), small objects (people on the left side of the frame who are crossing the street) and dynamic background (electronic billboards on the buildings), BSUV-Net 2.0 handles them well. One exception is the detection of extremely small people. While the detection of people in the right part of the frame is nearly perfect, some people on the left are sometimes missed due to their very small size and/or camouflage. The performance of BSUV-Net 2.0 drops slightly at night (foreground detections for some cars have holes in Figure 4·4b), however overall it is still solid as most of the cars are detected either in full or partially. Note, that we have used exactly the same weights while computing the day-time and night-time predictions and updated the empty and recent background frames as described above. These results show that BSUV-Net 2.0 can automatically adapt to different illumination conditions and can be used in real-world applications with a reasonably good performance.

4.4 Discussion

While background subtraction algorithms achieve remarkable performance today, they still often fail in challenging scenarios such as shaking or panning/tilting/zooming cameras, or when moving objects stop for an extended period of time. In the case of supervised algorithms, this is largely due to the limited availability of labeled videos recorded in such scenarios – it is difficult to train end-to-end deep-learning algorithms for unseen videos. To address this, we introduced several spatio-temporal data augmentation methods to synthetically increase the number of inputs in such scenarios. Specifically, we introduced new augmentations for PTZ, camera jitter and intermittent object motion scenarios, and achieved significant performance improvements in these categories and, consequently, a better overall performance on the CDNet-2014 dataset. We also introduced a real-time version of BSUV-Net 2.0 which still performs better than state-of-the-art methods and we proposed a 4-fold cross-validation data split for CDNet-2014 for easier comparison of future algorithms. Furthermore, we demonstrated a strong generalization capacity of BSUV-Net 2.0 using cross-dataset evaluation on LASIESTA in which the proposed model significantly outperforms the current state-of-the-art methods on a completely unseen dataset. Finally, we discussed how BSUV-Net 2.0 can be used in real-life applications that require 24/7 operation and provided examples of real-life results that show promise.

Chapter 5

Datasets for People Detection from Fisheye Cameras

In the second part of this thesis, we focus on another video-analytics task, namely people detection in images captured by overhead fisheye (OHF) cameras. When we started working on the topic, there were just a few existing OHF datasets for people detection and they were all annotated either by point location of a person's head [del Blanco et al., 2016, del Blanco et al., 2021] or by a bounding box aligned with image boundaries [Demiröz et al., 2012, Ma et al., 2018b]. However, due to the overhead vantage point and unique lens geometry, a standing person captured by an OHF camera appears radially oriented in the field of view of the camera. This is clearly visible in Figure 5·1. Therefore, bounding boxes aligned with image boundaries cannot capture these varying body orientations.

In order to address this limitation, we collected three new datasets of overhead fisheye videos and annotated them with rotated bounding boxes tightly drawn around each person and re-annotated a subset of the Mirror Worlds dataset [Ma et al., 2018b] with rotated bounding boxes:

- **H**uman-**A**ligned **B**ounding **B**oxes from **O**verhead **F**isheye cameras (HABBOF) [Li et al., 2019]
- **M**irror **W**orlds-**R**otated (MW-R) [Ma et al., 2018b, Duan et al., 2020]
- **C**hallenging **E**vents for **P**erson **D**etection from **O**verhead **F**isheye cameras

(CEPDOF) [Duan et al., 2020]

- In-the-**W**ild **P**eople **D**etection from **O**verhead **F**isheye cameras (WEPDOF) [Tezcan et al., 2021a]

Table 5.1 shows the statistics of these datasets. We report an approximate number of distinct people for MW-R since the exact number is not provided [Ma et al., 2018b].

Table 5.1: Statistics of 4 new datasets for people detection from overhead fisheye cameras. "people" refers to the annotated people and "# of people" refers to the number of annotated people in a single frame.

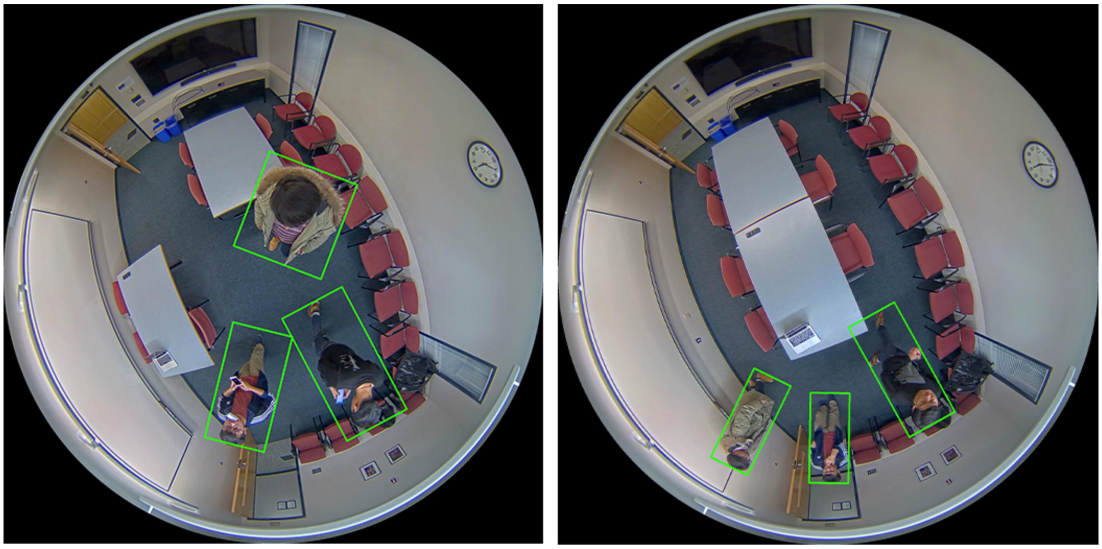
Dataset	**# of video clips**	**# of scenes**	**# of identities**	**# of people per frame**	**# of frames**	**Resolution (MP)**
HABBOF	4	2	9	2-5	5,837	∼4.2
MW-R	19	7	∼15	1-6	8,752	1.1 to 2.2
CEPDOF	8	1	17	1-13	25,504	1.1 to 4.2
WEPDOF	16	14	188	1-35	10,544	0.6 to 5

5.1 HABBOF

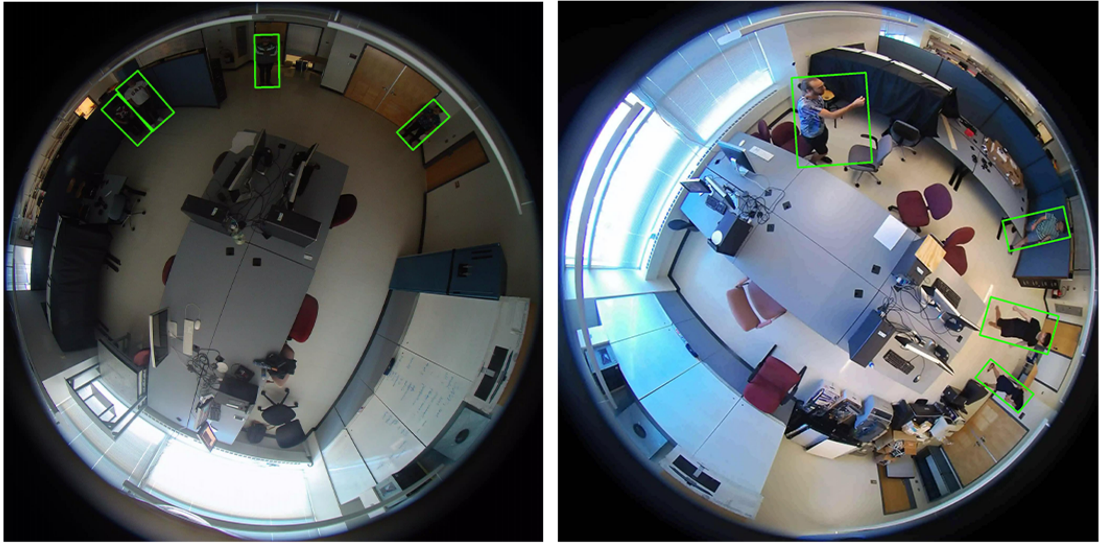
Our first dataset, Human-Aligned Bounding Boxes from Overhead Fisheye cameras (HABBOF) [Li et al., 2019], includes 4 videos captured in 2 different rooms. Table 5.2 provides scenario and quantitative details for each video in HABBOF, while Figure 5·1 shows sample frames from all videos with superimposed bounding-box annotations. "Meeting1" and "Meeting2" videos were recorded by an AXIS M3057-PLVE camera whereas "Lab1" and "Lab2" videos – by a Geovision GV-FER12203 camera. Videos in the dataset capture some challenging scenarios, such as spatial and temporal illumination variations, occlusions, and occupant presence in the center and at the periphery of the fisheye field of view. Capturing occupants at the field-of-view periphery is important in the case of fisheye cameras due the challenge it introduces – a severe geometric distortion for far-away objects. However, since HABBOF is the

Table 5.2: Scenario details and statistics of individual videos in HABBOF.

Video Scenario	**Video Sequence**	**Description/Challenges**	**Max # of people**	**# of frames**	**Resolution** ($W \times H$)	**FPS**
Conference room	Meeting1	Walking activity at image periphery	3	1,200	2048 × 2048	30
Conference room	Meeting2	Walking activity in image center and at periphery	3	1,200	2048 × 2048	30
Computer lab	Lab1	Occlusions Complex poses and walking at image periphery Spatially-nonuniform illumination	4	1,800	2048 × 2048	30
Computer lab	Lab2	Close proximity between people Time-varying global illumination Spatially-nonuniform illumination	4	1,800	2048 × 2048	12



Meeting1 Meeting2



Lab1 Lab2

Figure 5·1: Sample frames with rotated bounding-box annotations from all 4 videos in HABBOF.

first benchmark dataset with rotated bounding boxes it is somewhat limited in terms of the number of people, videos and challenging scenarios.

We made the videos and annotations of HABBOF publicly available[1] for research purposes. To date, it has been downloaded 264 times.

5.2 MW-R

Mirror Worlds (MW) [Ma et al., 2018b] is another dataset developed for people detection from OHF cameras. It includes 30 videos at two resolutions: $1,056 \times 960$ or $1,488 \times 1,360$. However, all person objects in the videos are annotated with bounding boxes aligned with image boundaries.

In order to evaluate our algorithms on the MW dataset and to provide more resources for the research community, we manually annotated a subset of MW videos with rotated bounding-box labels from scratch. We refer to this new dataset as Mirror Worlds – Rotated (MW-R). In MW-R, bounding boxes of the same person carry the same ID in consecutive frames, and thus can be also used for additional vision tasks using overhead, fisheye images, such as video-object tracking and person re-identification.

MW-R consists of training-set videos of the original MW dataset, 19 videos in total. Table 5.3 provides scenario and quantitative details for each video in MW-R, while Figure 5·2 shows sample frames with superimposed bounding-box annotations. More detailed information about the videos and frames can be found on the original Mirror Worlds website[2].

We made the annotations of MW-R publicly available[3] for research purposes. To date, it has been downloaded 78 times.

---

[1] `vip.bu.edu/habbof`

[2] `www2.icat.vt.edu/mirrorworlds/challenge/index.html`

[3] `vip.bu.edu/mw-r`

Table 5.3: Description and statistics of individual videos in MW-R.

Video Name	**Description**	**Duration (sec)**	**Resolution** $(W \times H)$	**FPS**
MW-18Mar-2	1 person walk in observation room	30	1488×1360	15
MW-18Mar-3	1 person walk in observation room	52	1488×1360	15
MW-18Mar-7	2 person in observation room	30	1056×960	15
MW-18Mar-8	2 person in observation room	30	1056×960	15
MW-18Mar-10	4 person walk in observation room	30	1056×960	15
MW-18Mar-12	4 person pose change in observation room	30	1056×960	15
MW-18Mar-13	4 person pose change in observation room	30	1056×960	15
MW-18Mar-14	4 person walk in observation room	30	1056×960	15
MW-18Mar-17	4 person pose change in observation room	30	1056×960	15
MW-18Mar-18	4 person walk in observation room	30	1056×960	15
MW-18Mar-19	3 person in hallway	30	1056×960	15
MW-18Mar-21	3 person in hallway	30	1056×960	15
MW-18Mar-22	3 person in hallway	30	1056×960	15
MW-18Mar-23	3 person in hallway	30	1056×960	15
MW-18Mar-24	2 person in hallway	30	1056×960	15
MW-18Mar-25	5 person in hallway	30	1056×960	15
MW-18Mar-26	2 person in hallway	30	1056×960	15
MW-18Mar-27	5 person in hallway	30	1056×960	15
MW-18Mar-31	3 person in hallway	30	1056×960	15

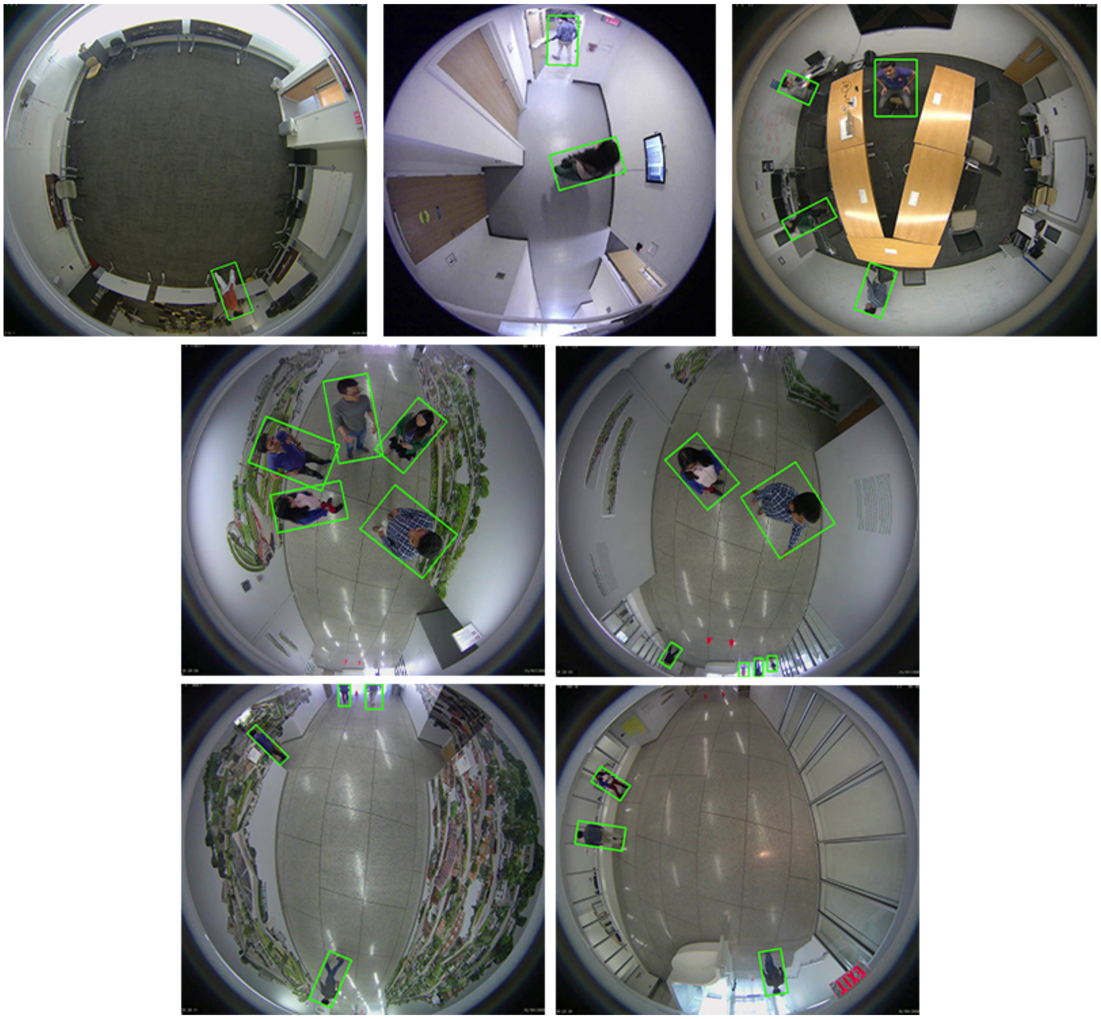


Figure 5·2: Sample frames with rotated bounding-box annotations from MW-R.

5.3 CEPDOF

After publishing HABBOF and MW-R, we collected and labeled another dataset that we named Challenging Events for Person Detection from Overhead Fisheye cameras (CEPDOF) [Duan et al., 2020]. Table 5.4 provides scenario and quantitative details for each video in CEPDOF and Figure 5·3 shows sample frames from all videos with superimposed bounding-box annotations. Clearly, CEPDOF contains many more frames and human objects than HABBOF, and also includes challenging scenarios such as a crowded room, various body poses, and low-light conditions. Similarly to MW-R, CEPDOF is also annotated spatio-temporally, i.e., the same person carries the same ID across frames. All of the videos in CEPDOF were recorded by an AXIS M3057-PLVE camera.

We made the videos and annotations of CEPDOF publicly available[4] for research purposes. To date, it has been downloaded 229 times.

Although HABBOF, CEPDOF and MW-R include some challenging scenarios, the recorded videos have been staged, that is people move according to predefined test scenarios (e.g., everyone starts moving at the same time and performs similar actions). Furthermore, the variety of scenes and person identities are limited in these datasets (see Table 5.1). Thus, they are not sufficient to evaluate the performance of people detection algorithms for unplanned, natural scenarios that can be expected to occur in real life. This exposes the need for a more challenging dataset recorded *in the wild* with a large variety of different scenes, actions and people.

5.4 WEPDOF

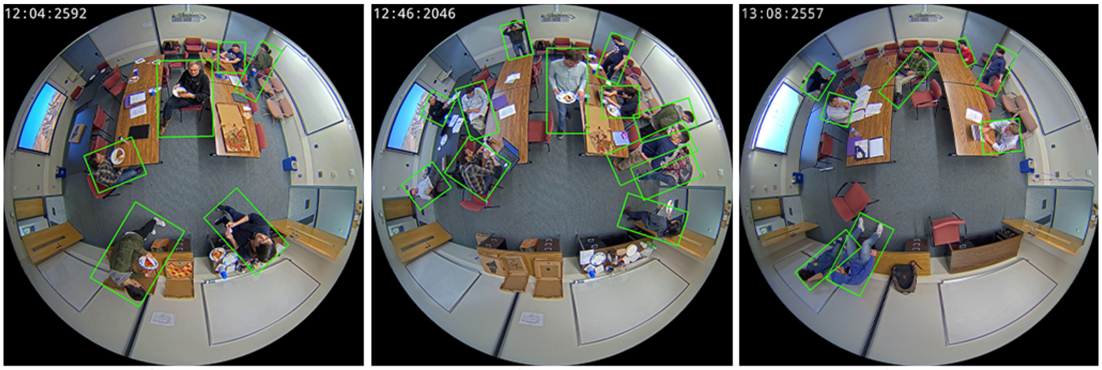
Our most recent, yet unpublished, dataset, In-the-Wild Events for People Detection from Overhead Fisheye cameras (WEPDOF) [Tezcan et al., 2021a], consists of videos

---

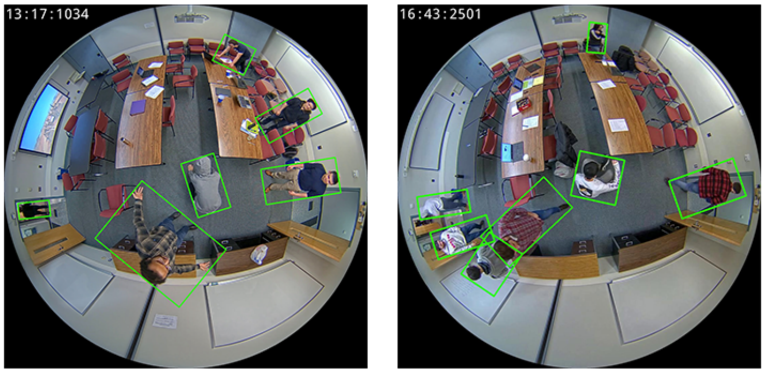
[4] `vip.bu.edu/cepdof`

Table 5.4: Scenario details and statistics of individual videos in CEPDOF.

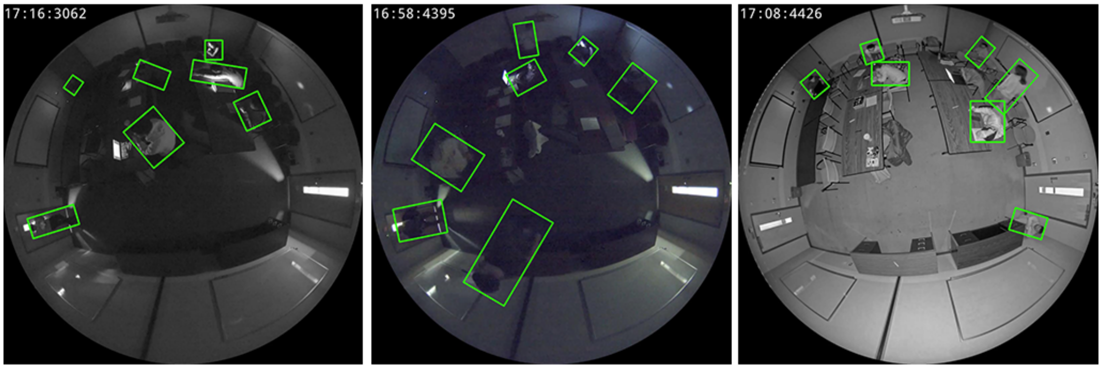
Video Scenario	**Video Sequence**	**Description/Challenges**	**Max # of people**	**# of frames**	**Resolution** ($W \times H$)	**FPS**
Common activities	Lunch meeting 1	People walking and sitting	11	1,201	2048×2048	1
	Lunch meeting 3	People walking and sitting	10	900	2048×2048	1
Crowded scene	Lunch meeting 2	More than 10 people sitting and having lunch	13	3,000	2048×2048	10
Edge Cases	Edge cases	People walking and sitting Extreme body poses Head camouflage Severe body occlusions.	8	4,201	2048×2048	10
Walking activity	High activity	People frequently walking in through one door and leaving through the other door.	9	7,202	1080×1080	10
Low light	All-off	People walking and sitting Overhead lights off Camera IR filter removed No IR illumination	7	3,000	1080×1080	10
	IRfilter	People walking and sitting Overhead lights off With camera IR filter No IR illumination	8	3,000	1080×1080	10
	IRill	People walking and sitting Overhead lights off Camera IR filter removed With IR illumination	8	3,000	1080×1080	10



Lunch Meeting 1 Lunch Meeting 2 Lunch Meeting 3



Edge Cases High Activity



All Off IR Filter IR Illumination

Figure 5·3: Sample frames with rotated bounding-box annotations from all 8 videos in CEPDOF.

recorded *in the wild* with diverse scenes, actions, people, imaging hardware, etc. Such a dataset is crucial for a fair and extensive evaluation of people detection algorithms from OHF images and videos. Table 5.5 provides scenario and quantitative details for each video in WEPDOF, while Figure 5·4 shows sample frames from all videos with superimposed bounding-box annotations. The unique characteristics of WEPDOF are discussed below.

- **In-the-wild videos:** Unlike the previous datasets for people detection from OHF cameras that have been recorded in staged scenarios, all of the videos in WEPDOF have been collected from YouTube (mostly via security cameras) and represent natural human behavior. This is important for assessing an algorithm's performance in real-world scenarios.

- **Variety:** As shown in Table 5.1, WEPDOF includes 14 different videos[5] recorded in completely different scenes (e.g., open office, cubicles, exhibition center, kindergarten and shopping mall). The number of people appearing in a single frame, spatial resolution and length of the videos in WEPDOF all vary significantly. Furthermore, since the videos in WEPDOF come from different sources, they have been captured by different camera hardware (e.g., sensor and lens) installed at different heights working under different illumination conditions.

- **Real-life Challenges:** As shown in Figure 5·4, WEPDOF captures real-world challenges such as camouflage, severe occlusions and geometric distortions. For example, in the frame from "Exhibition Setup" in Figure 5·4 it is very difficult to find some people since the color of their clothing is very similar to the background, an effect known as camouflage that is frequently encountered in

---

[5] Two of WEPDOF videos have been divided into two segments, thus overall there are 16 video clips in the dataset.

Table 5.5: Scenario details and statistics of individual videos in WEPDOF.

Video	**Challenges**	**# of clips**	**# of people**	**# of frames**	**Resolution ($W \times H$)**
Empty Store	Occlusion	1	1	358	1440 × 1080
Exhibition Setup	Camouflage Tiny people Partly visible people Cropped view	1	3-9	2400	1920 × 1080
Convenience Store	Cropped view Static people	1	6-7	1075	960 × 720
Large Office	Static people Occlusion Tiny people	2	7-9	700	1350 × 1080
Warehouse	High camera Cropped view	1	7-10	496	1920 × 1080
Exhibition	Occlusion Static People Crowded space Tiny people	1	17-18	690	1080 × 1080

Video	**Challenges**	**# of clips**	**# of people**	**# of frames**	**Resolution ($W \times H$)**
Call Center	Static people Tiny people Cropped view Crowded space	1	33-35	610	1920 × 1080
Tech Store	Tiny People Camouflage Occlusion High camera Cropped view	1	2-4	633	2592 × 1944
Jewelry Store	Occlusion	2	3-10	900	1620 × 1080
Street Grocery	Distorted aspect ratio Non-circular field of view Cropped view	1	5-7	231	944 × 1080
Printing Store	Camouflage Static people	1	7-8	1055	1440 × 1440
Repair Store	Camouflage Occlusion Partly visible people	1	8-10	947	900 × 720
IT Office	Static people Tiny people	1	14-15	500	1440 × 1080
Kindergarten	Children	1	18-21	549	900 × 720

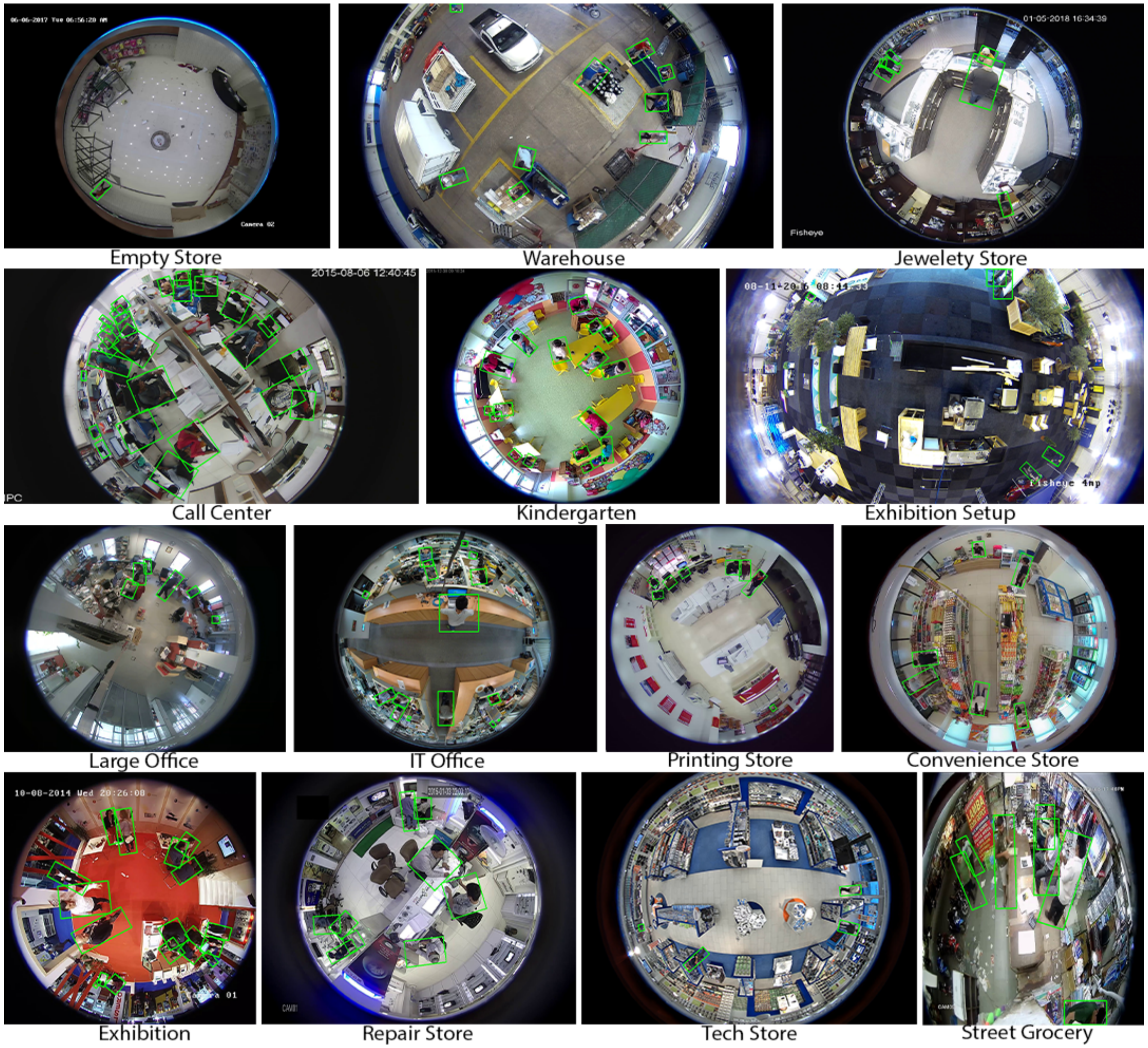


Figure 5·4: Sample frames with rotated bounding-box annotations from all 14 videos in WEPDOF.

practice. On the other hand, severe occlusions are clearly visible in "Call Center". Finally, geometric distortions manifest themselves either as a distorted aspect ratio of the images, such as in "Street Grocery", or as a dramatically reduced size of a person at the field-of-view periphery (the person is far away) as seen in "IT Office". The challenge of geometric distortions was not significantly captured in any of the previous datasets.

- **Region-of-Interest Maps:** In the annotations of WEPDOF, we exclude some of the areas that are close to the field-of-view periphery since people appear very small and close to each other making it nearly impossible to annotate accurately. These excluded regions are identified by means of a binary region of interest (ROI) map for each video. Figure 5·5 shows an example of ROI for "IT Office".
- **Spatio-Temporal Annotations:** Similar to CEPDOF, WEPDOF is annotated spatio-temporally, so it can be used for additional tasks such as person tracking and re-identification.

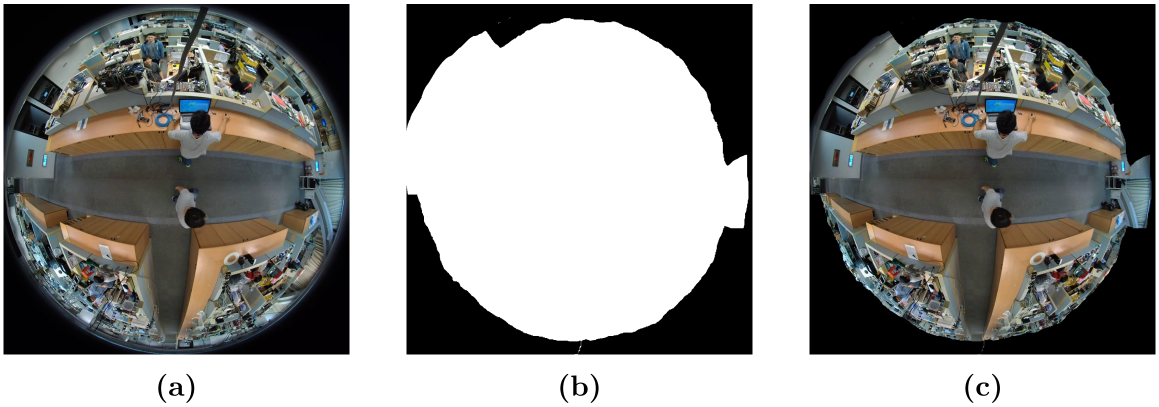

(a) **(b)** **(c)**

Figure 5·5: (a) Sample frame from WEPDOF's "IT Office", (b) its ROI map, (c) its ROI map overlayed on top of the frame.

We made the videos and annotations of WEPDOF publicly available[6] for research purposes and believe that WEPDOF will be beneficial for the development

---

[6] `vip.bu.edu/wepdof`

and evaluation of future algorithms for real-world people detection, tracking and re-identification.

5.5 Evaluation Methodologies

In order to fairly compare various supervised people detection algorithms, it is essential to apply several metrics on multiple datasets. Also, since some of the algorithms that we will introduce in the next two chapters use OHF videos for training, a clear separation of training and test sets is important. In this section, we propose two evaluation methodologies to compare people detection algorithms from OHF camera recordings.

5.5.1 Evaluation metrics for Staged-Scenario Datasets

For the staged-scenario datasets with a limited data diversity (HABBOF, CEPDOF and MW-R), we follow the MS COCO challenge [Lin et al., 2014] and adopt Average *Precision* (AP_{50}), i.e., the area under the *Precision-Recall* curve for the intersection over union (IoU) of 0.5, as one of our evaluation metrics. In addition to AP, we also adopt *F-score* at a fixed confidence threshold $\widehat{b}_{\text{conf}} = 0.3$ as another performance metric. Note that the *F-score* for a given value of $\widehat{b}_{\text{conf}}$ corresponds to a particular point on the *Precision-Recall* curve. For the algorithms that are trained on OHF images and/or videos (algorithms that will be introduced in Chapter 8 use multiple consecutive frames), we report the cross-validation results on MW-R, HABBOF and CEPDOF datasets, i.e., two datasets are used for training and the remaining one for testing, and this is repeated so that each dataset is used once as the test set.

5.5.2 Evaluation Metric for In-the-Wild Dataset

We compare in-the-wild performance of algorithms on WEPDOF and use AP_{50} as the main evaluation metric as well. Since WEPDOF is more extensive than the other datasets in terms of the variety of scenes and bounding box sizes, we introduce several data-specific metrics for WEPDOF in addition to AP_{50}.

Similarly to the MS COCO challenge [Lin et al., 2014], we use AP_{50} for small, medium and large bounding boxes denoted as AP_{50}^{S}, AP_{50}^{M} and AP_{50}^{L} respectively. Figure 5·6 shows the histogram of bounding-box areas in WEPDOF. We divide the bounding boxes into three groups: small (area ≤ 1200), medium ($1200 <$ area ≤ 8000) and large ($8000 <$ area) based on their areas normalized to image size of 1024×1024. Then, AP_{50}^{S} is calculated as AP_{50} between the small bounding-box annotations and small bounding-box detections. AP_{50}^{M} and AP_{50}^{L} are calculated similarly for medium and large bounding boxes. Table 5.6 shows the number of bounding-box annotations from these 3 categories for each video of WEPDOF. For AP_{50}^{S}, AP_{50}^{M} and AP_{50}^{L} scores, we compute the macro-average of the per-video results for the videos with at least 100 annotations in that category (e.g., "Street Grocery" is not used for AP_{50}^{M}).

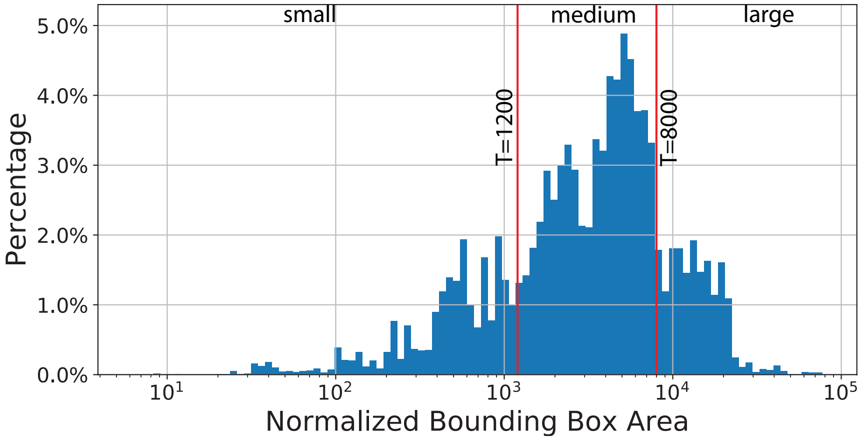


Figure 5·6: Histogram of bounding-box areas of the annotations in WEPDOF. All areas are normalized to image size of 1024×1024.

Although AP_{50} and its variants are very useful for summarizing the performance of an algorithm with a single number, in real-life applications the confidence threshold must be fixed. An optimal confidence threshold can be chosen as the one which

Table 5.6: The number of bounding boxes assigned to small, medium and large class (see text for the definition) for all WEPDOF videos.

Box size	**Empty Store**	**Exhibition Setup**	**Convenience Store**	**Large Office**	**Ware-house**	**Exhibi-tion**	**Call Center**	
Small	4	4472	8	1444	599	746	6622	
Medium	334	6268	5241	3615	3634	7699	9829	
Large	20	262	1314	200	0	3952	4312	
Box size	**Tech Store**	**Jewelry Store**	**Street Grocery**	**Printing Store**	**Repair Store**	**IT Office**	**Kinder-garten**	**Total**
Small	1122	2466	4	735	0	2308	1195	21725
Medium	1270	1900	41	6751	4862	3197	8486	63127
Large	51	1218	1272	65	3696	641	765	17768

results in the best *F-score* on a validation set. We report *Precision*, *Recall* and *F-score* metrics of the algorithms using this optimal confidence threshold.

For in-the-wild evaluation of *supervised* learning algorithms, we use the combination of MS COCO [Lin et al., 2014], MW-R , HABBOF and CEPDOF during training and evaluate the trained algorithms on WEPDOF using 2-fold cross-validation (see Table 5.7). In cross-validation, we use one of the sets as a validation set to find the best set of hyper-parameters and the other set for reporting the performance on unseen videos.

Table 5.7: Proposed sets for 2-fold cross-validation on WEPDOF

Set-1	**Set-2**
Empty Store	Tech Store
Exhibition Setup	Jewelry Store
Convenience Store	Street Grocery
Large Office	Printing Store
Warehouse	Repair Store
Exhibition	IT Office
Call Center	Kindergarten

In Chapters 6-8 we introduce people-detection algorithms for overhead fisheye cameras and use the described datasets for performance evaluation and comparisons.

Chapter 6

People Detection from Overhead Fisheye Cameras Using Rotating Windows

As we discussed in Section 2.2, most of the people-detection algorithms to date have been developed for side-view standard-lens (SVS) cameras; very few algorithms exist for overhead fisheye (OHF) cameras. Among the people-detection algorithms for SVS cameras, the best performance to date has been achieved by deep-learning algorithms. In particular, YOLO v3 [Redmon and Farhadi, 2018] achieves a very competitive performance in real time (using a desktop GPU).

However, YOLO v3 is trained on SVS images, where people usually appear upright, and its application to OHF images is not straightforward. A possible solution is to dewarp OHF frames using azimuthal-to-cylindrical projection so that people become upright-oriented (Figure 6·1). However, this transformation significantly distorts body shape right under the camera as shown in the first example in Figure 6·1 (dewarped image in the second row). Furthermore, depending on the start/end points of the transformation, it may split a body into two parts as shown in the second example (dewarped image in the third row). Moreover, YOLO v3 and other object-detection algorithms for SVS cameras do not support such panoramic images on input, so the dewarped images must be significantly padded from the top and bottom thus making people appear very small and decreasing the chances of detection.

In this chapter[1], we leverage YOLO v3, designed for a variety of objects includ-

---

[1] This chapter describes joint work with Shengye Li. It was published in the 2019 IEEE Interna-

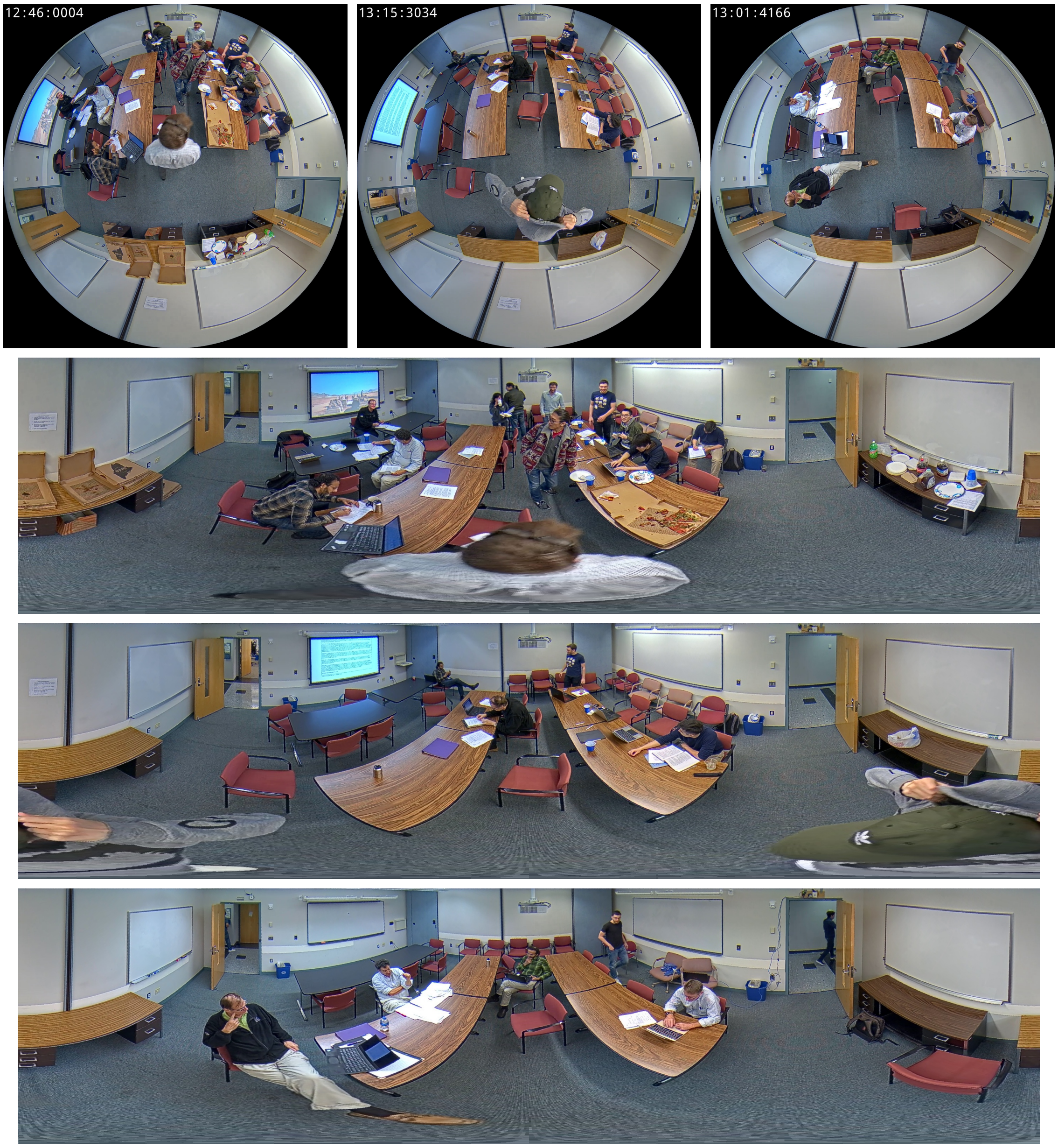


Figure 6·1: Examples of overhead fisheye images (top row) and their dewarped versions (last 3 rows).

ing upright-oriented people, to develop people-detection algorithms for OHF images where people appear at arbitrary orientations. We design two methods[2] leveraging YOLO v3. In one approach, we apply YOLO v3 only to a window extracted from the top-center part of a fisheye image where the orientation of people should be close to upright. To cover the whole image, we create 24 rotations of the image and apply YOLO v3 to the same window after each rotation. Then, we rotate the results back to the original angles and apply post-processing to prune multiple detections of the same person (the results from neighboring rotations may overlap). In an alternative approach, we first identify regions of interest (ROIs), where activity takes place, then we rotate each ROI to the top-center part of the image and apply YOLO v3. To identify areas of activity, we apply our own variant of classical background subtraction.

Note, that our aim in this section is to leverage high-performing object detection algorithms, designed for SVS images, to develop people-detection algorithms from OHF images. In particular, we use YOLO v3 pretrained on MS COCO dataset [Lin et al., 2014] and do not perform any additional training or fine-tuning. In this framework, YOLO v3 can be substituted by other object-detection methods such as R-CNN [Ren et al., 2015], RetinaNet [Lin et al., 2017b] and EfficentDet [Tan et al., 2020].

6.1 Activity-Blind Application of YOLO v3

Our first approach leverages the observation that the appearance of people in the top-center region of OHF images is similar to that in images from SVS cameras. In this approach, we first extract a rectangular window, which we shall call a *focus*

---

tional Conference on Advanced Video and Signal-based Surveillance (AVSS) [Li et al., 2019].

[2] The source code of the algorithms presented in this chapter is publicly available at `vip.bu.edu/habbof`

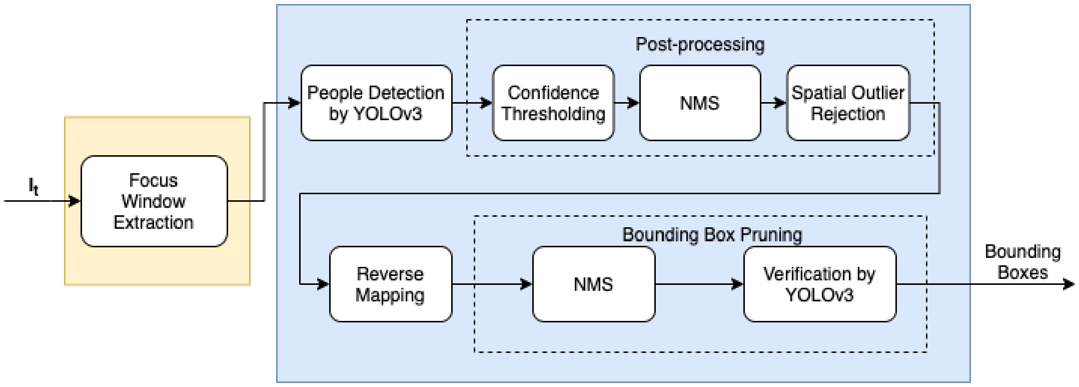


Figure 6·2: Block diagram of the proposed activity-blind (AB) people detection method.

window, at the top-center of an OHF image. In the next step, we rotate the image by a small angle, and extract the same window with new data. The rotation and window extraction steps are repeated until focus windows are extracted from all parts of the image. Then, we apply YOLO v3, trained on MS COCO dataset [Lin et al., 2014], as a person detector to each of the focus windows that we extracted, and perform a series of post-processing steps to generate reliable people detection results within each focus window. Subsequently, all detections are mapped from each focus window onto the complete fisheye image. Finally, multiple detections from neighboring focus windows are merged and verified to produce the final people detections. Since this approach does not utilize any activity information, we call it the *activity-blind* (AB) method. Its block diagram is shown in Fig. 6·2 and algorithm's details are provided next.

6.1.1 Focus Window and Image Rotation

Since YOLO v3 is designed for full-size images, we can use a focus window that is large enough to capture human bodies in an upright or almost upright position in the upper half of the image. We use a focus window whose height and width equal about 65% and 40% of the height and width of the full image, respectively, and the window's top is aligned with the upper boundary of the image. We applied image rotation in 15° increments as a trade-off between the overall complexity and precision. Figure 6·3 illustrates the selected window size and rotation angle.

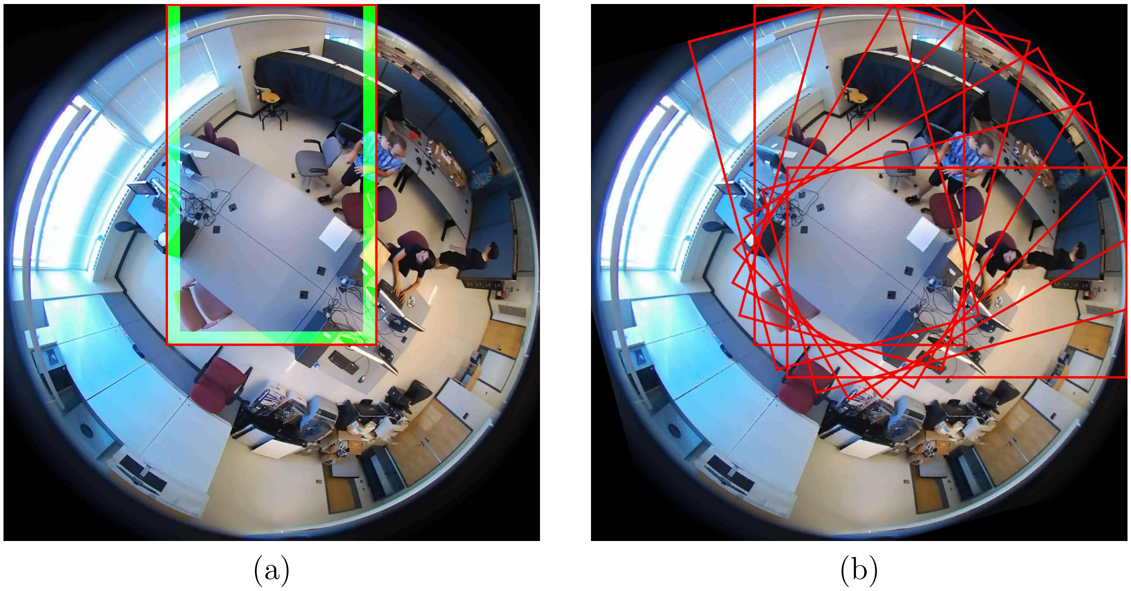

(a) (b)

Figure 6·3: (a) Placement of the 1,300×800-pixel focus window in a 2,048×2,048 image. The faint green area is the margin area defined in Section 6.1.2. (b) Focus window after reverse rotations.

6.1.2 Initial People Detection and Post-Processing

The YOLO v3 detector that we apply to every focus window is a *Fully Convolutional Network* (FCN) trained on the COCO dataset with 80 object classes. Since our focus is on people detection, we only retain those bounding boxes produced by YOLO v3 for which the confidence of the "people class" is high. Specifically, we only retain detections with an "objectness" score above a threshold of 0.3. Then, out of the retained object detections only those are kept whose person-class score is the highest among all object classes. YOLO v3 may detect a person with several bounding boxes that significantly overlap each other within a single focus window. In order to avoid over-counting people, only one representative box among the overlapping detections should be retained. To this end, we apply Non-Maximum Suppression (NMS) to the detections. Finally, people who are situated very close to the left, right, or bottom boundaries of the focus window may not be fully visible and might create bounding boxes that do not cover the whole body of the person. This could deteriorate the performance of people detection when we merge results across different focus windows as described in Section 6.1.4.

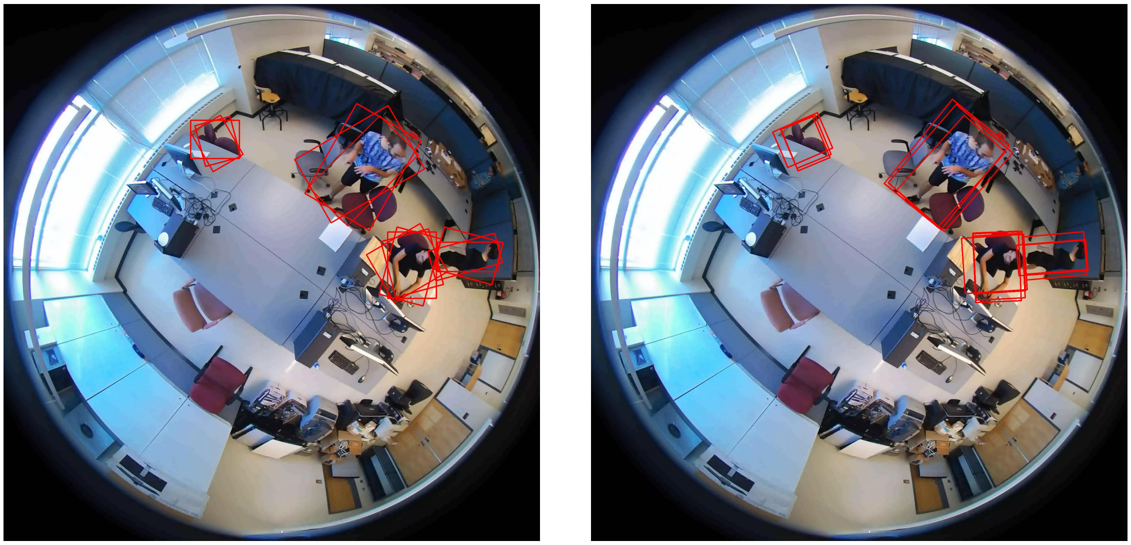
Therefore, we apply Spatial Outlier Rejection (SOR) to remove bounding boxes that intersect a Δ-wide margin inside the focus window along the right, left, and bottom boundaries of each focus window (Figure 6·3(a)). We set Δ to 6.25% of the focus window width.

6.1.3 Reverse Mapping of Detections

People detection results (bounding boxes) need to be mapped from the relative position within each extracted focus window to the absolute position in the full fisheye image. A naïve approach is to rotate the bounding box by reverse angle used in image rotation. However, the bounding boxes generated by YOLO v3 in any focus window are aligned to the focus window axes and need not be radially oriented with respect to the center of the fisheye image. We, therefore, reverse-rotate only the center of a bounding box and then form a new bounding box of the same size as the original one, but oriented radially. This process is illustrated in Figure 6·4.

6.1.4 NMS, Verification, and Final People Detection

Despite the bounding box suppression within each focus window during post-processing, typically there will be multiple overlapping bounding boxes after the reverse mapping. This is because neighboring focus windows have a large overlap with each other and a person can be detected within several focus windows without



(a) (b)

Figure 6·4: Comparison of the two reverse mapping approaches: (a) naïve approach here reverse-rotated bounding boxes are not radially oriented, and (b) improved approach where reverse-rotated bounding boxes are radially oriented and are better aligned with human bodies.

intersecting their boundaries. In order to assure accurate people detections, duplicate person detections need to be eliminated so that one person is associated with only one bounding box. Therefore, we implement NMS on the detections after the reverse mapping.

In order to reduce false positives resulting from erroneous detections by YOLO v3 that have not been eliminated by the preceding steps, we implement a final person detection verification step. Around each remaining bounding box, we extract a rectangular window that encompasses the box with a 30 pixel border. This new window is first rotated to the upright position as described in Section 6.1.3, but in the opposite direction. To account for potential angular misalignment, we apply additional rotations by ±15° to extract three windows that are passed to YOLO v3. The detection results then undergo the confidence thresholding and NMS post processing steps as detailed in Section 6.1.2. If at least 2 results confirm this is a person, then the original bounding box is accepted; otherwise it is rejected.

6.2 Activity-Aware Application of YOLO v3

The activity-blind approach is computationally intense since all the steps described in Section 6.1 need to be applied to each of the 24 focus windows, even if there is no person present. Therefore, we develop an *activity-aware method* (AA) to reduce the computational complexity. The main idea is to identify regions of interest (RoIs) where people are likely to be present, and apply a people detector to only windows containing these regions. We extract ROIs by means of background subtraction (BGS), i.e., by detecting changes in the field of view of the camera.

As we discussed in the earlier chapters, the best BGS algorithms are mostly supervised, learning-based ones (e.g., BSUV-Net, BSUV-Net 2.0). However, *all* of the existing background-subtraction datasets have been recorded with SVS cameras. Based on our experiments, the performance of supervised BGS algorithms trained on these datasets deteriorates significantly when applied to fisheye videos. Thus, in this work we opt for a simple model-based background subtraction algorithm.

The block diagram of our AA algorithm is shown in Figure 6·5 where the steps in the right blue box are exactly the same as those in our AB algorithm.

6.2.1 Background Subtraction

Let $\boldsymbol{I}_t(x,y)=[I_t^R, I_t^G, I_t^B]$ and $\boldsymbol{B}_t(x,y)=[B_t^R, B_t^G, B_t^B]$ denote the RGB color components of the observed image and a reference background image, respectively, at time t and pixel spatial coordinates (x, y). We will describe in Section 6.2.2 how the reference background $\boldsymbol{B}_t$ is obtained.

In the first step, the following thresholding is applied to produce an initial mask

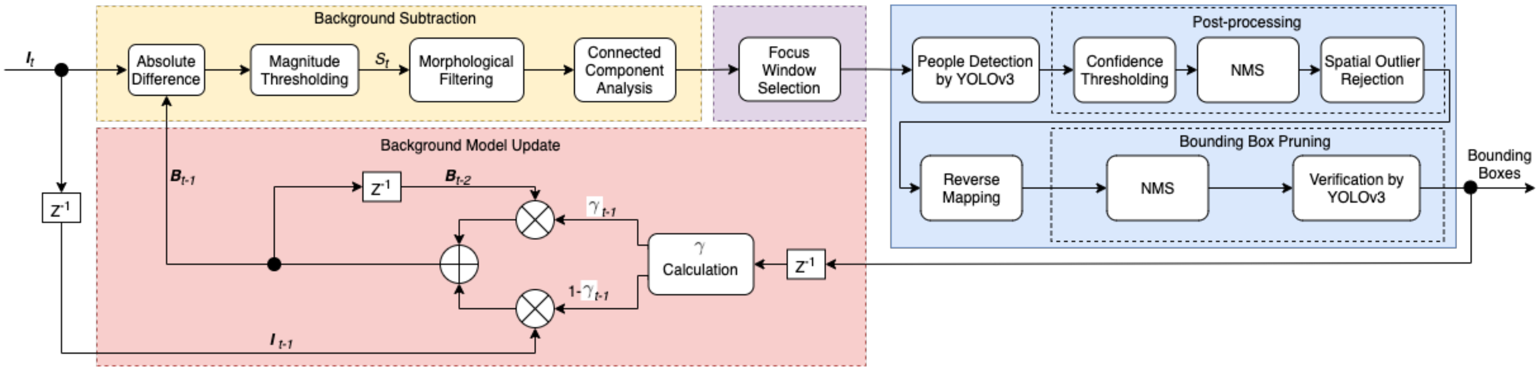


Figure 6·5: Block diagram of the proposed AA algorithm. Blocks within the blue-shaded rectangle are common to both AA and AB methods, while the other parts are unique to the AA algorithm.

of changes $S_t(x, y)$:

$$S_t(x, y) = \begin{cases} 1, & \text{if} \quad \sum_{A \in \{R,G,B\}} |I_t^A(x, y) - B_{t-1}^A(x, y)| > \theta \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

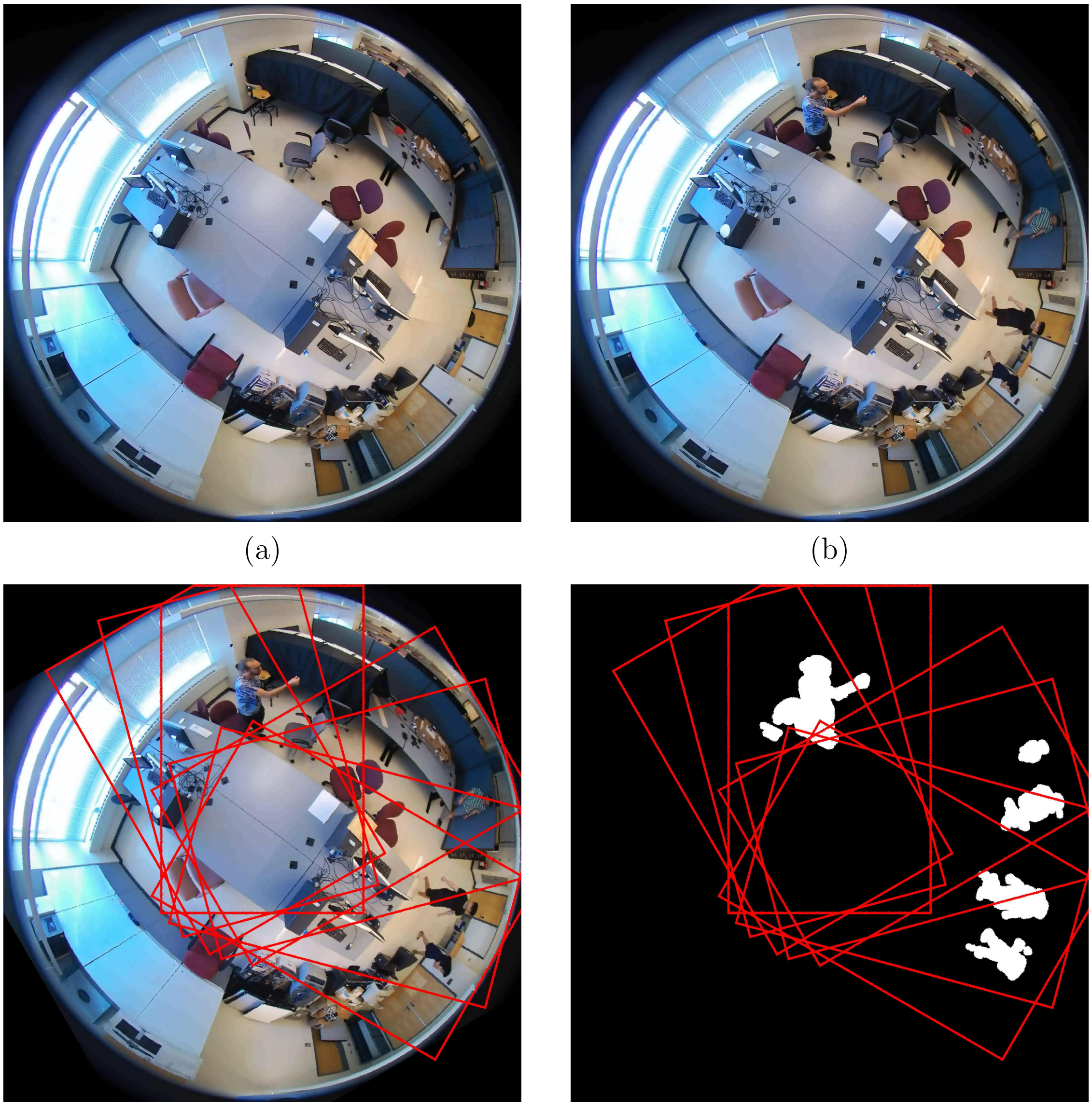
where θ is a threshold.

An example of reference background and input frame is shown in Figure 6·6(a–b). Subsequently, two morphological operations are applied to the mask $S_t(x, y)$. First, opening with a 3×3 rectangular structuring element is applied to remove tiny patches that are likely to arise due to noise. Then, dilation operation with a 25×25 elliptical structuring element is applied to expand the remaining areas of the detected changes. A connected-component analysis is then performed and small-area components (having fewer than 3,600 pixels) are removed. This leads to the final RoI mask shown in Figure 6·6(d).

6.2.2 Background Model

A variety of background models have been studied in the literature. A simple static background model is usually a fixed "empty" frame but it cannot reflect changes in the background, such as those due to illumination variations. Also, it may lead to unnecessarily large RoIs. Dynamic background models utilize recent frames to update model parameters, but they may produce false negatives if moving objects become nearly stationary for longer than the time period at which model parameters get updated. Instead, we use the following simple dynamic background model which leverages people detection results from previous time instants:

$$\boldsymbol{B}_t(x, y) = \gamma_t(x, y) \cdot \boldsymbol{B}_{t-1}(x, y) + (1 - \gamma_t(x, y)) \cdot \boldsymbol{I}_t(x, y),$$



(a) (b)

(c) (d)

Figure 6·6: Examples of: (a) reference background; (b) current frame; (c) focus windows for the current frame overlaid on the RGB image; and (d) the same focus windows overlaid on the connected components of the final RoI.

where $\gamma_t(x, y)$ equals 1 if (x, y) belongs to a bounding box associated with a detected person and is zero otherwise.

We note that at time t, people detection is performed first (so that γ_t is known) and then the background is updated. Our background update mechanism uses the indicator γ_t to decide at each location (x, y) whether to use the current image value at time t as the new background (pixel belongs to the background) or to use the background value from previous time (pixel belongs to a bounding box associated with a person). This reduces background contamination which affects many dynamic background models. Since background locations outside of a person's bounding box get immediately updated by the current image value, the model is robust to illumination changes that are challenging for static background models. The proposed update mechanism therefore offers benefits of both static and dynamic background models. We set the initial background to zero, i.e., $\boldsymbol{B}_0(x, y) = 0,\ \forall(x, y)$.

6.2.3 Focus Window Selection

We use a subset of the 24 rotated windows proposed in the AB method, and apply the same methodology to each selected window. In order to ensure full coverage of a connected component by rotated windows of width W and height H, the centroid C of a connected component is calculated first. Let O denote the center of camera's FOV and $R_i, i = 1, ...24$ the center of each of the 24 rotated windows. First, window number $k = \arg\min_i \angle(\vec{OC}, \vec{OR_i})$ is selected as the focus window for this connected component. If the connected component exceeds the left boundary of the focus window, the next window in counterclockwise direction is added. Then, the same check is performed for the newly-added window. The process is repeated until the connected component does not exceed the left boundary of the recently-added window. A similar procedure is applied to the right boundary and neighboring windows in the clockwise direction. Upon completion of this procedure, the connected component is fully included in the union of selected focus windows. An example of the final window selection is shown in Figure 6·6(d).

The above steps are repeated for all remaining connected components. Then, people detection by YOLO v3, post-processing, reverse mapping and people detection (as detailed in Section 6.1) are applied to all focus windows selected earlier.

6.3 Experimental Results

Table 6.1: Performance comparison of people detection methods on MW-R, HABBOF and CEPDOF datasets. P, R and F denote *Precision*, *Recall*, and *F-measure*, respectively.

MW-R				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	78.2	0.863	0.759	0.807
AA	88.4	**0.939**	0.819	0.874
AB	**95.6**	0.895	**0.902**	**0.898**

HABBOF				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	87.3	**0.970**	0.827	0.892
AA	87.7	0.922	0.867	0.892
AB	**93.7**	0.881	**0.935**	**0.907**

CEPDOF				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	61.0	0.884	0.526	0.634
AA	73.9	**0.896**	0.638	0.683
AB	**76.9**	0.884	**0.694**	**0.743**

The proposed AA and AB algorithms are compared against the state-of-the-art algorithm at the time [Tamura et al., 2019] on the three staged datasets (MW-R, HABBOF, CEPDOF) in Table 6.1 and on WEPDOF in Table 6.2. In AA and AB, we always use input images of size 1024 × 1024 and we apply YOLO v3 trained on 608 × 608 images from MS COCO [Lin et al., 2014] to the focus windows. Since

Table 6.2: In-the-wild comparison of people-detection algorithms on WEPDOF. Average run-times are computed on a node with a single NVIDIA Tesla V100 GPU.

Algorithm	AP_{50}	AP_{50}^{S}	AP_{50}^{M}	AP_{50}^{L}	P	R	F	Avg. run-time per frame
[Tamura et al., 2019]	59.8	11.6	65.2	61.3	0.777	0.508	0.581	**98 ms**
AA	68.3	11.4	70.1	**63.7**	0.804	**0.647**	**0.705**	1477 ms
AB	**69.8**	**15.8**	**71.3**	63.1	**0.818**	0.643	0.702	1776 ms

there is no publicly available source code for [Tamura et al., 2019], we implemented it based on our best understanding. Following the implementation details listed in [Tamura et al., 2019], it is trained on 608 × 608 images from the rotated version of MS COCO. During inference on MW-R, HABBOF and CEPDOF, we resized the images to 608 × 608 to match the training resolution as suggested in their paper. As for WEPDOF, the people at field-of-view periphery occupy just a few pixels at 608×608 resolution, thus we resized the frames to 1024×1024 during inference. Both AA and AB methods outperform Tamura *et al.* [Tamura et al., 2019] on almost all datasets for most of the metrics. In particular, the AB method outperforms Tamura *et al.* by $7-26\%$ in terms of AP_{50} and by $2-21\%$ in terms of *F-score*. These results clearly demonstrate that the series of pre- and post-processing steps, that we proposed, enable YOLO to be effective in people detection from OHF images despite the fact that YOLO is not trained on OHF images. The performance gap between Tamura *et al.* and our proposed algorithms is very minimal on HABBOF which is the least challenging dataset. This gap significantly increases on more challenging datasets such as CEPDOF and WEPDOF. This shows the promise of AA and AB algorithms in more challenging scenarios.

In terms of the comparison of AA and AB, there is a clear trade-off in terms of *Precision* and *Recall* on MW-R, HABBOF and CEPDOF. The AA method successfully eliminates some false positives (FPs) and thus produces a better *Precision*. At

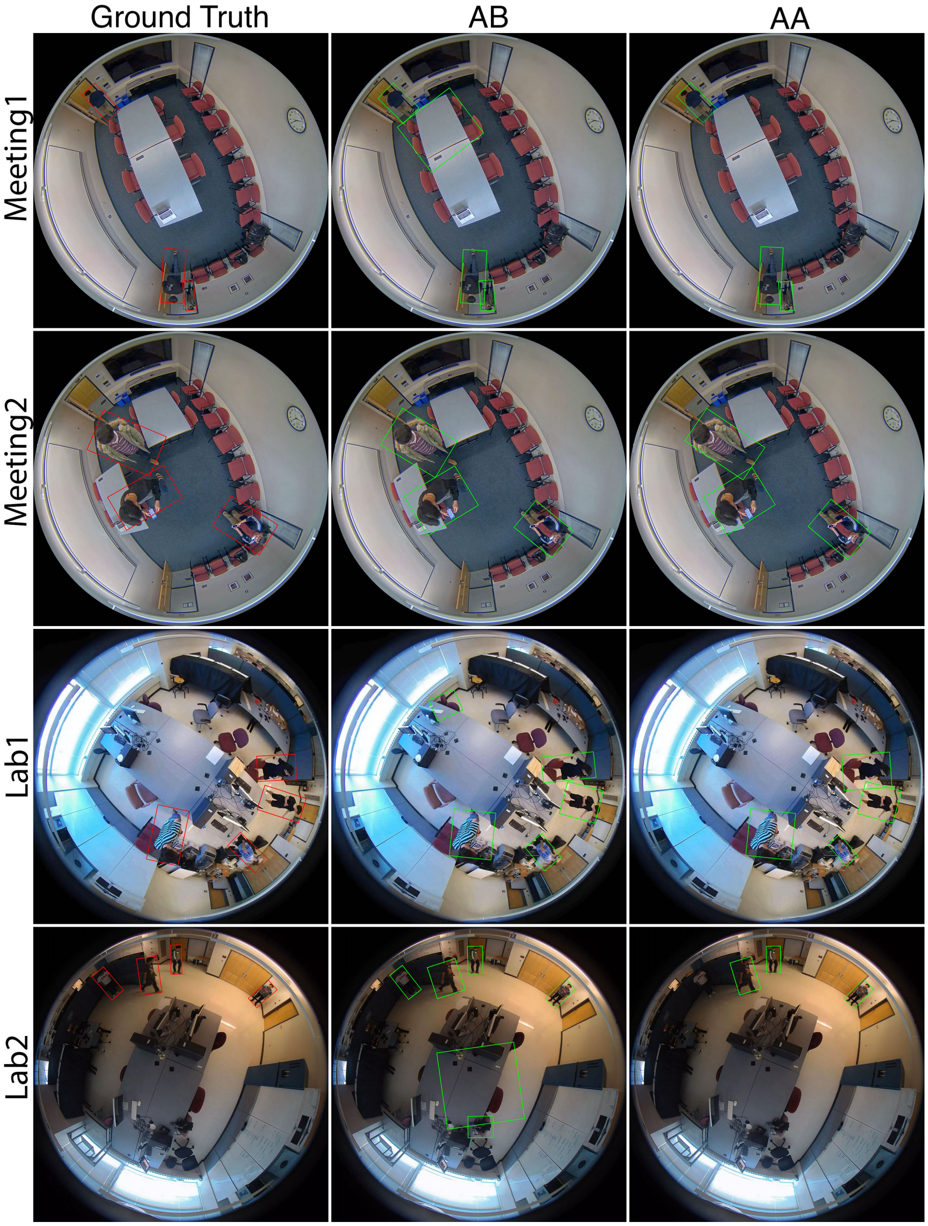


Figure 6·7: Qualitative comparison of people-detection results on sample frames from each video (one per column) in the HABBOF dataset. Columns show the ground truth annotations as well as the AB and AA people-detection results.

the same time, it produces less true positives (TPs) leading to lower *Recall*. This is caused by the fact that most people appear within a single focus window in the AA method (focus window is constructed around RoI). However, even if a person is in the window's center, he/she may not be exactly upright and the detection by YOLOv3 may fail. In contrast, in the AB method a person appears in several neighboring focus windows, in each at a slightly different angle and in some with a fully-visible body, so that there are more chances for detection. Therefore, the activity-blind method outperforms the activity-aware method in terms of true positives. An example of this is shown in the "Lab2" row of Figure 6·7. The person standing in the upper-left part part of the image is not detected by AA, but is successfully detected by AB thus producing a true positive. On the other hand, the AB method produces more false positives than the AA method, since the AA method implements YOLO only on selected windows containing RoIs, and this reduces the chance of making an erroneous detection. In the "Lab1" row of Figure 6·7, AA eliminates the false positive detection produced by AB around the chair in the upper-left part of the image. Similarly, in the "Lab2" row, AA eliminates the large false positive detection produced by AB in the middle of the image and a smaller one around the chair. This trade-off between the *Precision* and *Recall* reverses on WEPDOF as shown in Table 6.2. This suggests that the background subtraction algorithm used in AA is not successful on the challenging videos of WEPDOF recorded in the wild.

Both AA and AB achieve much lower AP_{50} scores on WEPDOF compared to the staged datasets. This suggests that although they can handle relatively simple situations, they are not sufficiently powerful for real-world scenarios. Table 6.2 also shows AP_{50} scores of the tested algorithms for bounding boxes of different sizes, as described in Section 5.5.2. The performance difference between AA and AB algorithms is significantly larger for small objects compared to large ones. This suggests that the background subtraction algorithm introduced in AA misses small foreground objects resulting in false negatives in people detection.

Table 6.2 also shows run-times of the algorithms. Since the proposed algorithms apply YOLO v3 multiple times and also perform several pre- and post-processing steps, they are both much slower than Tamura *et al.* [Tamura et al., 2019]. As expected, AA is faster than AB since it applies YOLO v3 only to the focus windows centered around the regions of interest. However, this difference will diminish as the monitored scene includes more activity.

6.4 Discussion

We proposed two novel people-detection algorithms for overhead fisheye images by leveraging a state-of-the-art object-detection algorithm (YOLO v3) trained on standard side-captured images. In our activity-blind approach, we rotated the image 24 times to cover the entire field of view and produced detection results for the whole image. In the activity-aware approach, we used background subtraction as a pre-processing step to find the windows of interest in the frame and applied YOLO v3 only to these windows. Experimental results show that both AA and AB outperform the state-of-the-art in terms of multiple metrics. The observed trade-offs between AA and AB suggest that they can be individually leveraged for different types of applications. The main bottleneck of both AA and AB methods is the elevated processing time.

In the next chapter, instead of leveraging an object-detection algorithm designed for standard images, we will introduce an end-to-end approach which produces rotated bounding boxes in a single run with a run-time that is comparable to that of Tamura *et al.*

Chapter 7

End-to-End People Detection from Overhead Fisheye Cameras

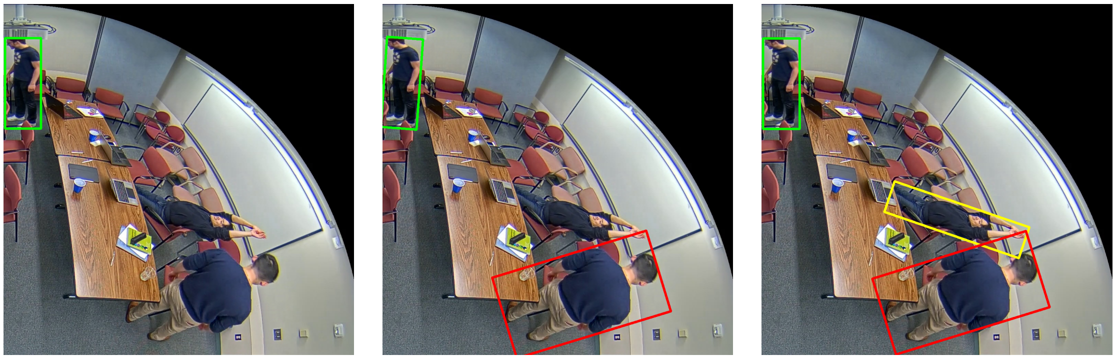
As discussed in Chapter 2, object-detection algorithms that are designed for standard images perform poorly on OHF images, usually missing non-upright bodies (Figure 7·1a). In such images, standing people appear along image radius (Figure 7·1b), due to the overhead placement of the camera, and rotated bounding boxes are needed. To accommodate this rotation, in Chapter 6 we introduced YOLO-based rotating-window approaches for people detection from OHF cameras. Although our approaches outperformed the state-of-the-art, their computational complexity proved to be very high due to the detection step applied to up to 24 focus windows and the pre- and post-processing steps.

In this chapter[1], we introduce Rotation-Aware People Detection (RAPiD)[2], a novel end-to-end people-detection algorithm for overhead, fisheye images. RAPiD is a convolutional neural network that predicts arbitrarily-rotated bounding boxes (Figure 7·1c) of people in a fisheye image. It extends the model proposed in YOLO v3 [Redmon and Farhadi, 2018], one of the most successful object detection algorithms for standard images. In addition to predicting the center and size of a bounding box, RAPiD also predicts its angle. This is accomplished by a periodic loss function based

---

[1] This chapter describes joint work with Zhihao Duan. It was published in the Omnidirectional Computer Vision Workshop within the 2020 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) [Duan et al., 2020].

[2] The source code of RAPiD is publicly available at `vip.bu.edu/rapid`



(a) Axis-aligned **(b)** Radius-aligned **(c)** Human-aligned

Figure 7·1: Illustration of *typical* people-detection results on overhead, fisheye images (one quarter shown) for algorithms using various bounding-box orientation constraints; the human-aligned bounding boxes fit bodies most accurately. These are not outputs from any algorithms. See the text for discussion.

on an extension of a common regression loss. This allows us to predict the exact rotation of each bounding box in an image without any assumptions on its orientation and additional computational complexity. Since RAPiD is an end-to-end algorithm, we can train or fine-tune its weights on annotated fisheye images. Indeed, we show that such fine-tuning of a model trained on standard images significantly increases the performance. An additional aspect of this work, motivated by its focus on people detection, is the replacement of the common regression-based loss function used in multi-class object detection algorithms [Redmon et al., 2016, Liu et al., 2016, Girshick, 2015, Ren et al., 2015] with single-class object detection. The inference speed of RAPiD is nearly identical to that of YOLO since it is applied to each image only once without the need for pre-/post-processing.

We evaluate the performance of RAPiD on the OHF camera datasets that we introduced in Chapter 5 and show that it significantly outperforms the state-of-the-art including the AA and AB methods introduced in Chapter 6, while also running significantly faster than both of them.

7.1 Notation

RAPiD's design has been largely motivated by YOLO v3 [Redmon and Farhadi, 2018]. In this section, we explain this design in detail and we highlight the concepts we borrowed from YOLO as well as novel ideas that we proposed.

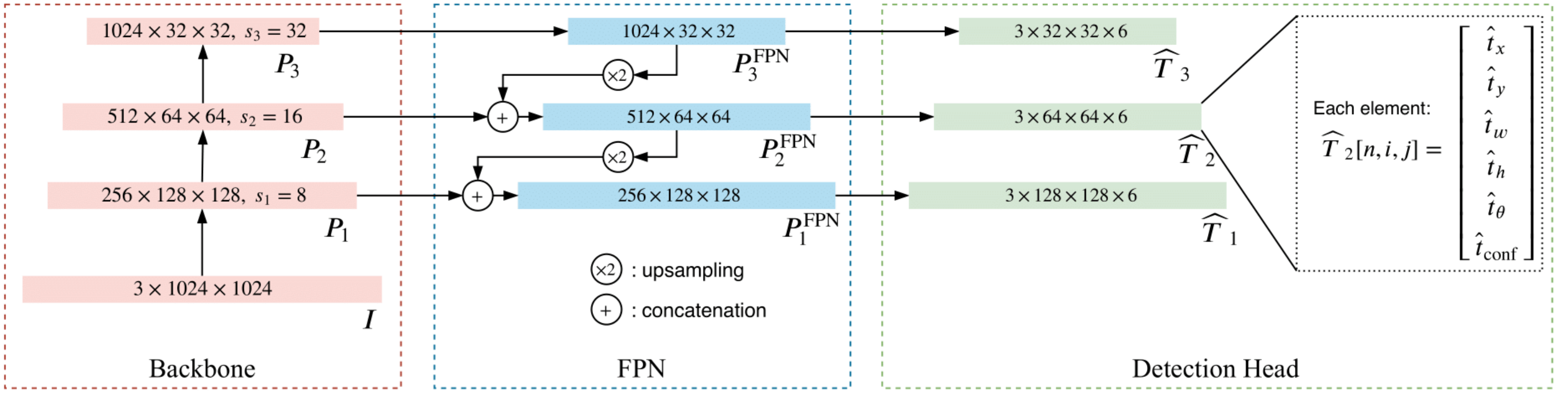
We use $\boldsymbol{b} = (b_x, b_y, b_w, b_h, b_\theta) \in \mathbb{R}^5$ to denote a ground-truth bounding box, where b_x, b_y are the coordinates of the bounding box center; b_w, b_h are the width and height and b_θ is the angle by which the bounding box is rotated clockwise. Similarly $\hat{\boldsymbol{b}} = (\widehat{b}_x, \widehat{b}_y, \widehat{b}_w, \widehat{b}_h, \widehat{b}_\theta, \widehat{b}_{\text{conf}}) \in \mathbb{R}^6$ denotes a predicted bounding box, where the additional element $\widehat{b}_{\text{conf}}$ denotes the confidence score of the prediction. All the angles used in the paper are in radians.

7.2 Network Architecture

Our object-detection network, illustrated in Figure 7·2, can be divided into three stages: backbone network, feature pyramid network (FPN) [Lin et al., 2017a], and bounding-box regression network, also known as the detection head:

$$\begin{aligned} P_1, P_2, P_3 &= \text{Backbone}(I) \\ P_1^{\text{fpn}}, P_2^{\text{fpn}}, P_3^{\text{fpn}} &= \text{FPN}(P_1, P_2, P_3) \\ \widehat{T}_k &= \text{Head}_k(P_k^{\text{fpn}}) \quad \forall k = 1, 2, 3 \end{aligned} \tag{7.1}$$

where $I \in [0,1]^{3\times h\times w}$ is the input image, $\{P_k\}_{k=1}^3$ denotes a multi-dimensional feature matrix and $\{\widehat{T}_k\}_{k=1}^3$ denotes a list of predicted bounding boxes in transformed notation (the relationship between $\widehat{T}$ and $\hat{\boldsymbol{b}}$ will be defined soon – see equation (7.2)) at three levels of resolution. Below, we describe each stage in depth. For additional details, interested readers are referred to [Redmon and Farhadi, 2018].



$1024 \times 32 \times 32,\ s_3 = 32$
P_3
$512 \times 64 \times 64,\ s_2 = 16$
P_2
$256 \times 128 \times 128,\ s_1 = 8$
P_1
$3 \times 1024 \times 1024$
I
Backbone
$1024 \times 32 \times 32$
P_3^{FPN}
$512 \times 64 \times 64$
P_2^{FPN}
$256 \times 128 \times 128$
P_1^{FPN}
×2 : upsampling
+ : concatenation
FPN
$3 \times 32 \times 32 \times 6$
$\widehat{T}_3$
$3 \times 64 \times 64 \times 6$
$\widehat{T}_2$
$3 \times 128 \times 128 \times 6$
$\widehat{T}_1$
Each element:
$\widehat{T}_2[n,i,j] = \begin{bmatrix} \hat{t}_x \\ \hat{t}_y \\ \hat{t}_w \\ \hat{t}_h \\ \hat{t}_\theta \\ \hat{t}_{\text{conf}} \end{bmatrix}$
Detection Head

Figure 7·2: RAPiD architecture. Following the paradigm of one-stage detectors, our model contains a backbone, FPN, and detection head (bounding-box regression network). In the diagram, each arrow represents multiple convolutional layers and the colored rectangles represent multi-dimensional matrices, i.e., feature maps, whose dimensions correspond to input image of size $h \times w = 1024 \times 1024$.

7.2.1 Backbone

The backbone network, also known as the feature extractor, takes an input image I and outputs a list of features (P_1, P_2, P_3) from different parts of the network. The main goal is to extract features at different spatial resolutions (P_1 being the highest and P_3 being the lowest). By using this multi-resolution pyramid, we expect to leverage both the low-level and high-level information extracted from the image.

7.2.2 Feature Pyramid Network (FPN)

The multi-resolution features computed by the backbone are fed into FPN in order to extract features relevant for object detection, denoted $(P_1^{\text{fpn}}, P_2^{\text{fpn}}, P_3^{\text{fpn}})$. We expect P_1^{fpn} to contain information about small objects and P_3^{fpn} – about large objects.

7.2.3 Detection Head

After FPN, a separate CNN is applied to each feature vector $P_k^{\text{FPN}}, k \in \{1, 2, 3\}$ to produce a transformed version of bounding-box predictions, denoted $\widehat{T}_k$ – a 4-dimensional matrix with $\langle 3, h/s_k, w/s_k, 6\rangle$ dimensions. The first dimension indicates that there are three anchor boxes being used in $\widehat{T}_k$, the second and third dimensions denote the prediction grid, where $h \times w$ is the resolution of the input image and s_k is the stride at resolution level k as shown in Figure 7·2, and the last dimension denotes a transformed version of the predicted bounding box for each grid cell. We denote the n^{th} transformed bounding-box prediction of $Head_k$ in grid cell (i, j) as $\widehat{T}_k[n, i, j] = (\widehat{t}_x, \widehat{t}_y, \widehat{t}_w, \widehat{t}_h, \widehat{t}_\theta, \widehat{t}_{\text{conf}})$ from which a bounding-box prediction can be computed as follows:

$$\begin{aligned}
\widehat{b}_x &= s_k\left(j + \text{Sig}(\widehat{t}_x)\right), & \widehat{b}_w &= w_{k,n}^{\text{anchor}} e^{\widehat{t}_w} \\
\widehat{b}_y &= s_k\left(i + \text{Sig}(\widehat{t}_y)\right), & \widehat{b}_h &= h_{k,n}^{\text{anchor}} e^{\widehat{t}_h} \\
\widehat{b}_\theta &= \alpha\ \text{Sig}(\widehat{t}_\theta) - \beta, & \widehat{b}_{\text{conf}} &= \text{Sig}(\widehat{t}_{\text{conf}})
\end{aligned} \tag{7.2}$$

where $\mathrm{Sig}(\cdot)$ is the logistic (sigmoid) activation function and $w_{k,n}^{\text{anchor}}$ and $h_{k,n}^{\text{anchor}}$ are the width and height of the n^{th} anchor box for $Head_k$. Note, that angle prediction $\widehat{b}_\theta$ is limited to range $[-\beta, \alpha-\beta]$ (7.2). In Section 7.3.2 below, we discuss the selection of α and β values.

7.3 Angle-Aware Loss Function

Our loss function is inspired by that used in YOLOv3 [Redmon and Farhadi, 2018], with an additional bounding-box rotation-angle loss:

$$\begin{aligned}\mathfrak{L} = &\sum_{\hat{\boldsymbol{t}}\in\widehat{T}^{\text{pos}}} \mathrm{BCE}(\mathrm{Sig}(\widehat{t}_x), t_x) + \mathrm{BCE}(\mathrm{Sig}(\widehat{t}_y), t_y)\\ &+ \sum_{\hat{\boldsymbol{t}}\in\widehat{T}^{\text{pos}}} (\mathrm{Sig}(\widehat{t}_w) - t_w)^2 + (\mathrm{Sig}(\widehat{t}_h) - t_h)^2\\ &+ \sum_{\hat{\boldsymbol{t}}\in\widehat{T}^{\text{pos}}} \ell_{\text{angle}}(\widehat{b}_\theta, b_\theta)\\ &+ \sum_{\hat{\boldsymbol{t}}\in\widehat{T}^{\text{pos}}} \mathrm{BCE}(\mathrm{Sig}(\widehat{t}_{\text{conf}}), 1) + \sum_{\hat{\boldsymbol{t}}\in\widehat{T}^{\text{neg}}} \mathrm{BCE}(\mathrm{Sig}(\widehat{t}_{\text{conf}}), 0)\end{aligned} \tag{7.3}$$

where BCE denotes binary cross-entropy, ℓ_{angle} is a new angle loss function that we propose in the next section, $\widehat{T}^{\text{pos}}$ and $\widehat{T}^{\text{neg}}$ are positive and negative samples from the predictions, respectively, as described in YOLOv3, $\widehat{b}_\theta$ is calculated in equation (7.2) and t_x, t_y, t_w, t_h are calculated from the ground truth as follows:

$$\begin{aligned} t_x &= \frac{b_x}{s_k} - \left\lfloor \frac{b_x}{s_k} \right\rfloor, \quad t_w = \ln\left(\frac{b_w}{w_{k,n}^{\text{anchor}}}\right)\\ t_y &= \frac{b_y}{s_k} - \left\lfloor \frac{b_y}{s_k} \right\rfloor, \quad t_h = \ln\left(\frac{b_h}{h_{k,n}^{\text{anchor}}}\right)\end{aligned} \tag{7.4}$$

Note, that we do not use the category-classification loss since we use only one class (person) in our problem. For the confidence score ($\widehat{t}_{\text{conf}}$), we use BCE loss instead of the focal loss [Lin et al., 2017b] commonly used in object-detection algorithms, since the technical report of YOLO v3 states that focal loss decreases the performance of their network [Redmon and Farhadi, 2018].

Traditionally, regression functions based on $L1$ or $L2$ distance are used for angle prediction [Ma et al., 2018a, Ding et al., 2019, Yang et al., 2019b]. However, these metrics do not consider the periodicity of the angle and might result in misleading cost values due to symmetry in the parameterization of rotated bounding boxes. We solve these issues by using a periodic loss function and changing the parameterization, respectively.

7.3.1 Periodic Loss for Angle Prediction

Since a bounding box remains identical after rotation by π, the angle loss function must satisfy $\ell_{\text{angle}}(\widehat{\theta}, \theta) = \ell_{\text{angle}}(\widehat{\theta} + \pi, \theta)$, i.e., must be a π-periodic function with respect to $\widehat{\theta}$.

We propose a new, periodic angle loss function:

$$\ell_{\text{angle}}(\hat{\theta}, \theta) = f(\text{mod}(\hat{\theta} - \theta - \frac{\pi}{2}, \pi) - \frac{\pi}{2}) \tag{7.5}$$

where mod$(\cdot)$ denotes the modulo operation and f is any symmetric regression function such as $L1$ or $L2$ norm. Since $\frac{\partial}{\partial x}\text{mod}(x, \cdot) = 1$, the derivative of this loss function with respect to $\hat{\theta}$ can be calculated as follows,

$$\ell'_{\text{angle}}(\hat{\theta}, \theta) = f'(\text{mod}(\hat{\theta} - \theta - \frac{\pi}{2}, \pi) - \frac{\pi}{2}) \tag{7.6}$$

except for angles such that $\hat{\theta} - \theta = (k\pi + \pi/2)$ for integer k, where ℓ_{angle} is non-differentiable. However, we can ignore these angles during backpropagation as is commonly done for other non-smooth functions, such as $L1$ distance. Figure 7·3 shows an example plot of $\ell_{\text{angle}}(\hat{\theta}, \theta)$ with $L2$ distance as well as its derivative with respect to $\Delta\theta = \hat{\theta} - \theta$.

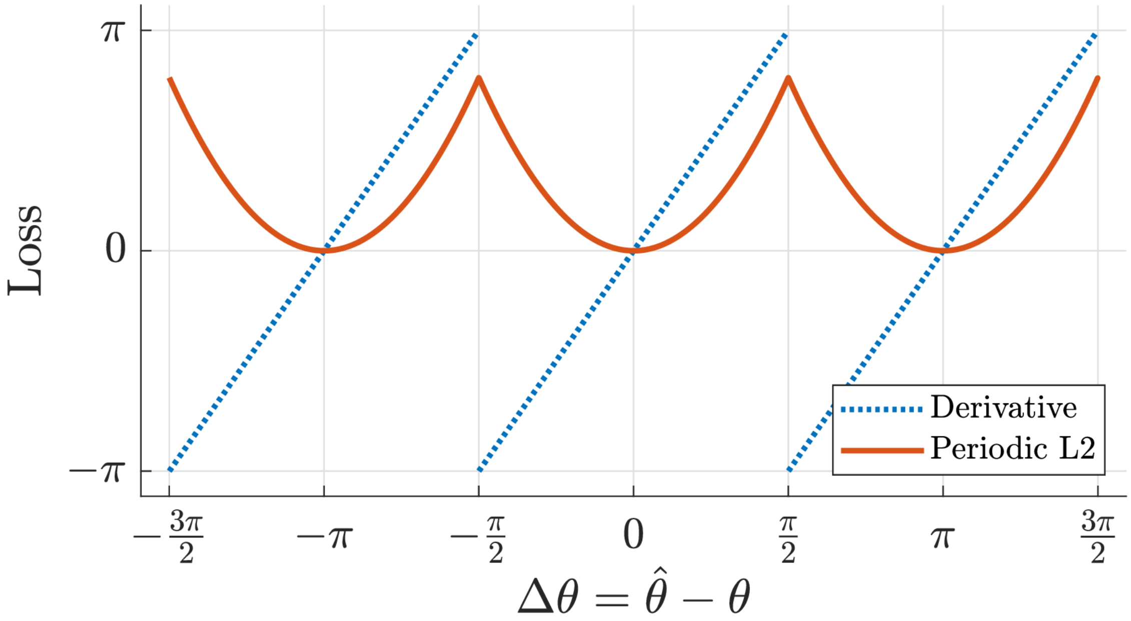


Figure 7·3: Periodic loss function with $L2$ norm as regressor and its derivative

7.3.2 Parameterization of Rotated Bounding Boxes

In most of the previous work on rotated bounding-box (RBB) detection, $[-\frac{\pi}{2}, 0]$ range is used for angle representation. This ensures that all RBBs can be uniquely expressed as $(b_x, b_y, b_w, b_h, b_\theta)$ where $b_\theta \in [-\frac{\pi}{2}, 0]$. However, as discussed in Chapter 2 and also in [Qian et al., 2019], this approach might lead to a large cost even when the prediction is close to the ground truth due to the symmetry of the representation, i.e., $(b_x, b_y, b_w, b_h, b_\theta) = (b_x, b_y, b_h, b_w, b_\theta - \pi/2)$. We address this by enforcing the following rule in our ground-truth annotations: $b_w < b_h$ and extending the ground-truth angle range to $[-\frac{\pi}{2}, \frac{\pi}{2})$ to be able represent all possible RBBs. For bounding boxes that are exact squares, a rare situation, we simply decrease a random side by 1 pixel. Under this rule, each bounding box will correspond to a unique 5-D vector representation.

Given the fact that the ground-truth angle θ is defined in $[-\frac{\pi}{2}, \frac{\pi}{2})$ range, it seems logical to force the predicted angle $\hat{\theta}$ to be in the same range by assigning $(\alpha, \beta) =$

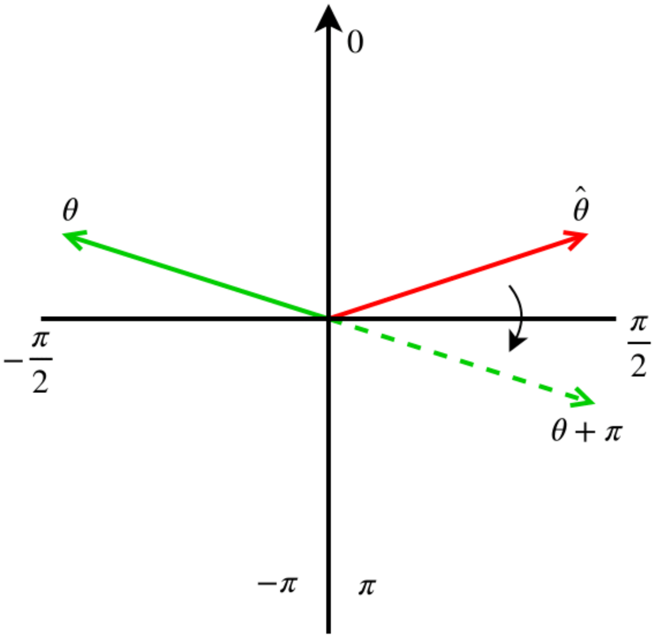


Figure 7·4: Illustration of the necessity to expand the predicted-angle value range. Gradient descent applied to the predicted angle $\hat{\theta}$ (red arrow) may rotate it clockwise and away from the ground truth angle θ (green arrow). Since a bounding box at angle $\theta + \pi$ is the same as the one at θ, we need to extend the angle range to include $\theta + \pi$ (dashed green arrow) otherwise $\widehat{\theta}$, pushed by the gradient, will stop at $\pi/2$.

$(\pi, \pi/2)$ in equation (7.2). However, this creates a problem for gradient descent when $\pi/2 < \hat{\theta} - \theta < \pi$ since the derivative of angle loss (7.6) will be negative (Figure 7·3). In this case, gradient descent will tend to increase $\hat{\theta}$ which will move it further away from the actual angle θ. Clearly, the network should learn to estimate the angle as $\theta + \pi$ instead of θ (Figure 7·4). To allow this kind of behavior, we extend the range of allowed angle predictions to $[-\pi, \pi)$ by assigning $(\alpha, \beta) = (2\pi, \pi)$.

Note that our new RBB parameterization will not have the symmetry problem explained above if the network eventually learns to predict the parametrization rule, $\widehat{b}_w \leq \widehat{b}_h$, which is very likely considering the fact that all ground-truth RBBs satisfy $b_w \leq b_h$. Indeed, based on our experiments in Section 7.5.1, we show that nearly all

RBBs predicted by RAPiD satisfy $\widehat{b}_w \leq \widehat{b}_h$.

In summary, by 1) defining $[-\frac{\pi}{2}, \frac{\pi}{2})$ as the ground truth angle range and forcing ground truth $b_w < b_h$, 2) using our proposed periodic angle loss function, and 3) setting predicted angle range to be $(-\pi, \pi)$, our network can learn to predict arbitrarily-oriented RBBs without problems experienced by previous RBB methods. Based on the experimental results in Section 7.5.1, we choose periodic L1 to be our angle loss function ℓ_{angle}.

7.4 Inference

During inference, an image $I \in \mathbb{R}^{3\times h\times w}$ is fed into the network, and three groups of bounding boxes (from three feature resolutions) are obtained. A confidence threshold is applied to select the best bounding-box predictions. After that, non-maximum suppression (NMS) is applied to remove redundant detections of the same person.

7.5 Experimental Results

A performance comparison of RAPiD against state-of-the-art algorithms is shown in Table 7.1 for the three staged datasets and in Table 7.2 for WEPDOF. Similarly to Section 6.3, RAPiD was trained using 608×608 images and tested on 608×608 images from MW-R, HABBOF and CEPDOF. However, in the case of WEPDOF it was trained using 608×608 images and tested on $1,024 \times 1,024$ images[3]. Clearly, RAPiD outperforms the other algorithms in nearly all of the evaluations while running just slightly slower than the fastest algorithm by Tamura *et al.* We note that RAPiD's performance is slightly better, in terms of AP_{50}, than that of the AB algorithm on MW-R dataset in which most human objects appear in an upright pose (walking).

---

[3] Training the spatio-temporal models from Chapter 8 using $1,024 \times 1,024$ images could not be supported by the memory of GPUs at our disposal and, consequently, we could not compare RAPiD with its extensions at this training-image resolution.

Table 7.1: Performance comparison of people-detection algorithms on three *staged* test datasets (MW-R, HABBOF, CEPDOF). P, R, and F denote *Precision*, *Recall*, and *F-measure*, respectively.

MW-R				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	78.2	0.863	0.759	0.807
AA	88.4	0.939	0.819	0.874
AB	95.6	0.895	0.902	0.898
RAPiD	**96.6**	**0.951**	**0.931**	**0.941**

HABBOF				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	87.3	0.970	0.827	0.892
AA	87.7	0.922	0.867	0.892
AB	93.7	0.881	**0.935**	0.907
RAPiD	**97.3**	**0.984**	**0.935**	**0.958**

CEPDOF				
Algorithm	AP_{50}	P	R	F
[Tamura et al., 2019]	61.0	0.884	0.526	0.634
AA	73.9	0.896	0.638	0.683
AB	76.9	0.884	0.694	0.743
RAPiD	**82.4**	**0.921**	**0.719**	**0.793**

Table 7.2: *In-the-wild* comparison of people-detection algorithms on WEPDOF. The average runtimes are computed on a node with a single NVIDIA Tesla V100 GPU.

Algorithm	AP_{50}	AP_{50}^{S}	AP_{50}^{M}	AP_{50}^{L}	P	R	F	avg. runtime per frame
[Tamura et al., 2019]	59.8	11.6	65.2	61.3	0.777	0.508	0.581	**98 ms**
AA	68.3	11.4	70.1	63.7	0.804	0.647	**0.705**	1477 ms
AB	69.8	15.8	71.3	63.1	**0.818**	0.643	0.702	1776 ms
RAPiD	**72.0**	**18.4**	**72.8**	**67.9**	0.731	**0.676**	0.668	118 ms

This is encouraging since people walking or standing appear radially oriented in OHF images, a scenario for which AA, AB and Tamura *et al.*'s algorithm have been designed. However, RAPiD outperforms the other algorithms by a large margin on HABBOF, which is relatively easy, and CEPDOF, which includes challenging scenarios, such as various body poses and occlusions. We conclude that RAPiD works well in both simple and challenging cases while maintaining high computational efficiency. Note, that among the algorithms reported in Table 7.1 RAPiD is the only one which allows training with overhead fisheye images which significantly improves performance (see Section 7.5.1 for a detailed discussion).

On WEPDOF, RAPiD outperforms the other algorithms in terms of AP_{50} but is outperformed by AA and AB in terms of *Precision* and *F-score*. In order to better understand this trade-off, we compare AA, AB and RAPiD using *Precision-Recall* plots in Figure 7·5a and *F-score* versus confidence threshold plots in Figure 7·5b. Although RAPiD produces a higher area under the PR curve, AA and AB perform better than RAPiD for high confidence-score thresholds suggesting that RAPiD produces bounding boxes with lower confidence. This might be due to the fact that AA and AB compute bounding-box predictions from overlapped crops of the same image and combine these results in a post-processing step. Thus, they can analyze the same person from different angles which boosts the confidence score of the bounding box for that person. Note that RAPiD is more than 10 times faster than AA and AB.

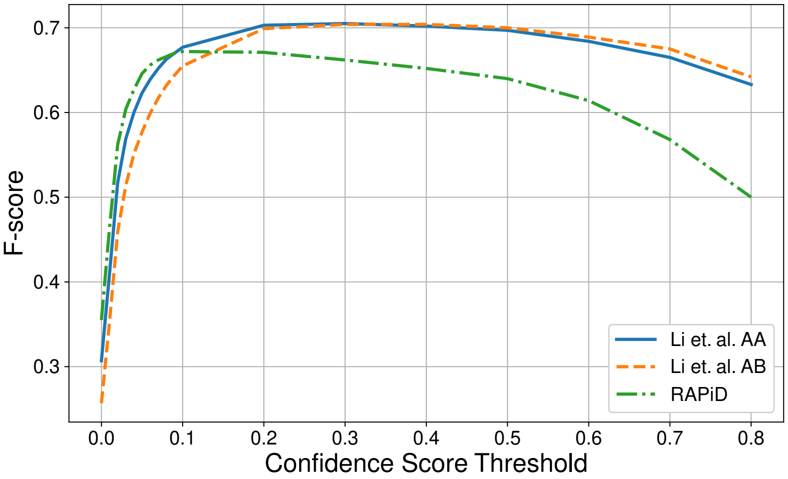
Figure 7·6 shows sample results produced by RAPiD on several videos; the detections are accurate in a range of scenarios, such as various body poses, orientations, and diverse background scenes. However, some scenarios, such as people's images on a projection screen (Figure 7·6g), low-light conditions, hard shadows, severe camouflage and very small people remain challenging.

7.5.1 Design Evaluation

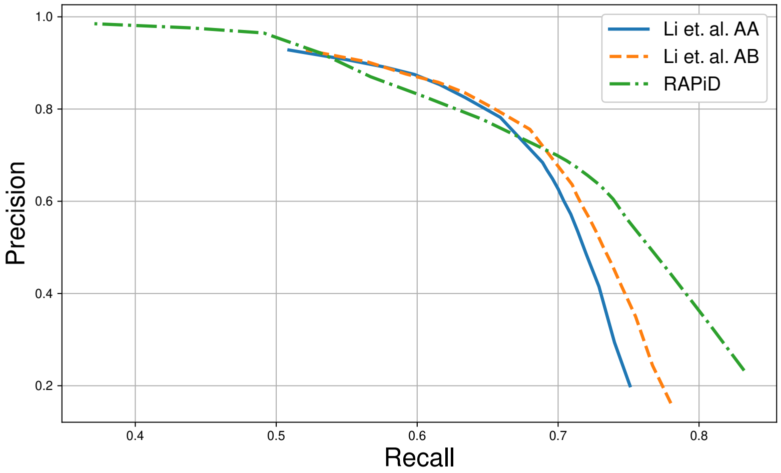
We conducted a number of experiments to better understand the impact of the novel elements we introduced in RAPiD on performance. Specifically, we conducted an ablation study and compared different angle loss functions. Due to limited GPU resources at our disposal, we did not run all of the evaluations for these experiments. Instead, we trained these algorithms on COCO and then fine-tuned them on MW-R using the same optimization parameters as reported in Section 7.5, unless stated otherwise. Then, we tested each algorithm on every video in the HABBOF and CEPDOF datasets at $1,024 \times 1,024$ resolution. The resulting AP values were averaged across all videos.

Ablation Experiments

We present various ablation experiments to analyze how each element of RAPiD individually contributes to the overall performance. As the baseline, we use the Tamura *et al.* algorithm [Tamura et al., 2019] with NMS and analyze the differences between this baseline and RAPiD one element at a time. Tamura *et al.* use standard YOLO [Redmon and Farhadi, 2018] trained on 80-classes of COCO with rotation-invariant training [Tamura et al., 2019] in which the object's angle is uniquely determined by its location. The first row of Table 7.3 shows the result of this baseline algorithm. Note that, the baseline algorithm is not trained or fine-tuned on overhead, fisheye frames.

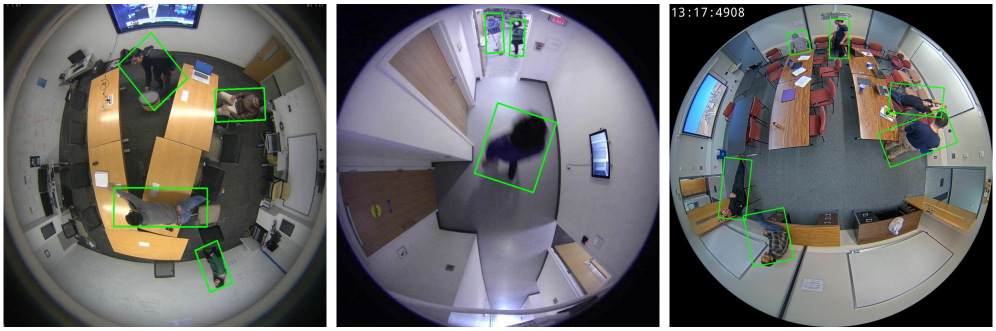


(a)

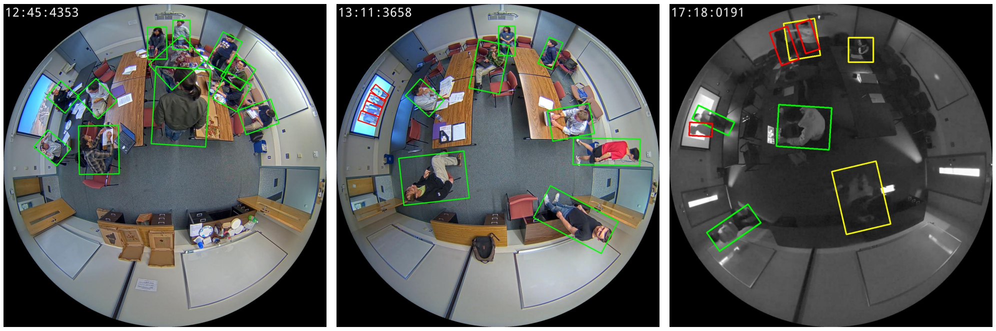


(b)

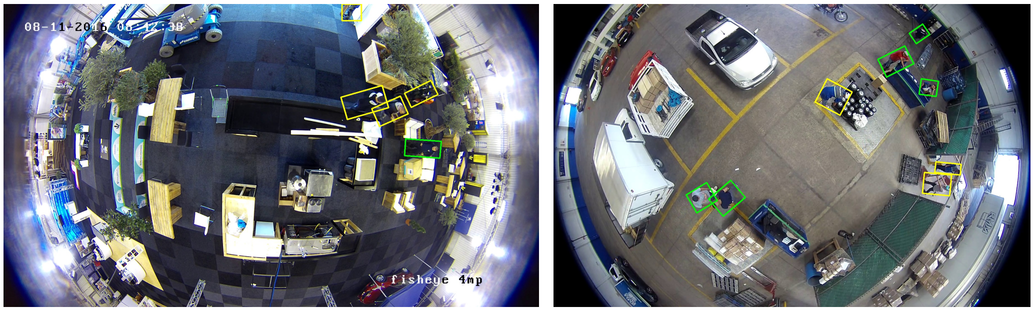
Figure 7·5: Comparison of AA, AB and RAPiD on WEPDOF in terms of: (a) *F-score* versus confidence-score threshold; and (b) *Precision-Recall* curves.



(a) Different poses. **(b)** Under camera. **(c)** Various angles.



(d) Occlusions. **(e)** People on the screen. **(f)** Low-light scenario.



(g) Severe camouflage. **(h)** Tiny-moving people.

Figure 7·6: Qualitative results of RAPiD on videos from MW-R (a, b), CEPDOF (c–f) and WEPDOF (g, h). Green boxes are true positives, red boxes are false positives, and yellow boxes are false negatives.

Table 7.3: Ablation study of RAPiD. The first row corresponds to the baseline algorithm. Fine-tuning is applied using the MW-R dataset.

No. of classes	Angle prediction	Fine- tuning	AP_{50}
80	Rotation-invariant		81.4
1	Rotation-invariant		81.2
1	Rotation-invariant	✓	85.9
1	Rotation-aware	✓	**88.9**

Multi-Class versus Single-Class: In RAPiD, we remove the category classification part of YOLO since we are dealing with a single object category, namely, a person (see Section 7.3). As can be seen from the second row of Table 7.3, this results in a slight performance drop, which was to be expected since training on 80 classes of objects can benefit from multi-task learning. However, removing the category-classification branch reduces the number of parameters by 0.5M.

Fine-Tuning with Overhead, Fisheye Images: To analyze this effect, we fine-tuned the single-class algorithm trained on COCO with images from MW-R. As shown in the third row of Table 7.3, this results in a significant performance increase. Recall that the test set used in Table 7.3 does not include any frames from the MW-R dataset.

Rotation-Aware People Detection: As discussed in Section 7.3, we introduced a novel loss function to make RAPiD *rotation-aware*. Instead of setting the object's angle to be along the FOV radius, we added a parameter, $\widehat{b_\theta}$, to each predicted bounding box and trained the network using periodic L1 loss. As shown in the last row of Table 7.3, the angle prediction further improves the performance of RAPiD.

Comparison of Different Angle Loss Functions

To analyze the impact of the loss function on angle predictions, we ablate the angle value range and angle loss function in RAPiD while keeping the other elements unchanged. We compare our proposed periodic loss with two baselines: standard unbounded regression loss and bounded regression loss. We perform the same experiment for both L1 and L2 loss. As can be seen in Table 7.4, the periodic L1 loss achieves the best performance, and both the periodic L1 and periodic L2 losses outperform their non-periodic counterparts.

Table 7.4: Comparison of RAPiD's performance for different angle ranges and loss functions.

Prediction range	Angle loss	AP_{50}
$(-\infty, \infty)$	L1	86.0
$(-\pi, \pi)$	L1	87.0
$(-\pi, \pi)$	Periodic L1	**88.9**
$(-\infty, \infty)$	L2	86.1
$(-\pi, \pi)$	L2	86.1
$(-\pi, \pi)$	Periodic L2	88.1

Analysis of Prediction Aspect Ratio

As discussed in Section 7.3.2, we relax the angle range to be inside $[-\pi/2, \pi/2)$ and force $b_w < b_h$ in ground-truth annotations so that every bounding box corresponds to a unique representation. In the same section, in order to handle the bounding-box symmetry problem we assumed that the network can learn to predict bounding boxes such that $\widehat{b}_w < \widehat{b}_h$. To demonstrate that this is indeed the case, we analyze the output of our network on both HABBOF and CEPDOF datasets. Figure 7·7 shows the histogram of $\widehat{b}_h/\widehat{b}_w$. We observe that nearly all predicted bounding boxes satisfy $\widehat{b}_w < \widehat{b}_h$ (i.e., $\widehat{b}_h/\widehat{b}_w > 1$), which validates our assumption.

7.5.2 Performance of RAPiD on Real-World Challenges

It is clear from Figures 7·6g and 7·6h, that RAPiD misses people under challenging real-world scenarios such as severe camouflage and very small projected body size. In order to better understand the performance of RAPiD on real-world challenges, we provide the results from Table 7.2 individually for each video in WEPDOF.

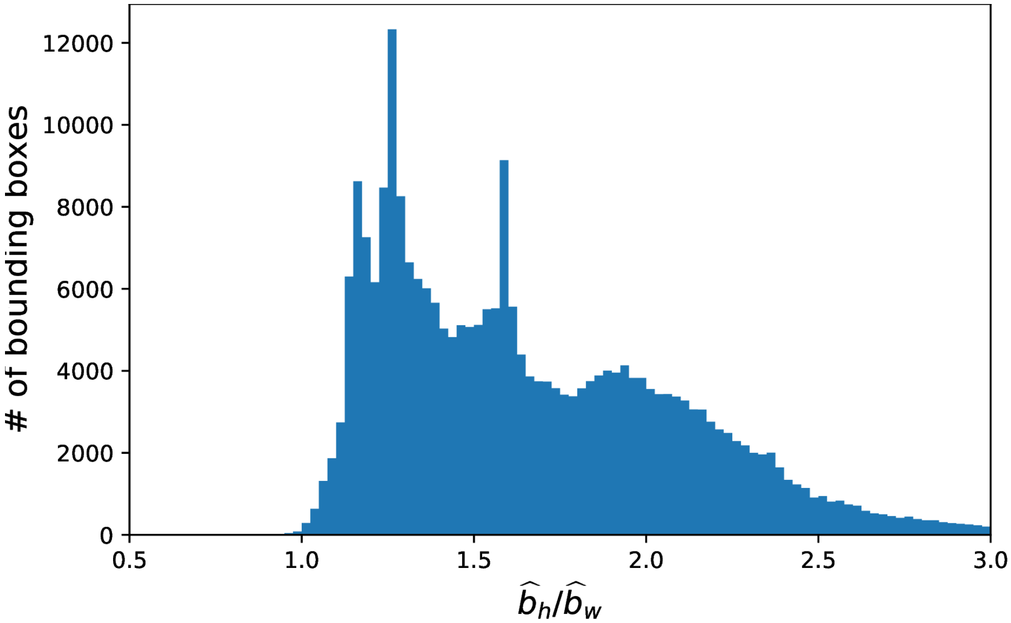


Figure 7·7: Histogram of the height-to-width ratio of the predicted bounding boxes.

Table 7.5 shows a per-video performance comparison of RAPiD with state-of-the-art algorithms on WEPDOF. RAPiD outperforms the other algorithms on 7 out of 14 videos. The performance improvement of RAPiD over state-of-the-art algorithms is most significant for “Street Grocery” and “Large Office”. “Street Grocery” has a non-circular FOV and, therefore, people appear not aligned with the FOV radius. However, the Tamura *et al.* [Tamura et al., 2019], AA and AB algorithms all assume radially-aligned bounding boxes and cannot handle such misalignments. In “Large Office”, people appear directly under the camera during a significant portion of the video. Since, the Tamura *et al.* [Tamura et al., 2019], AA and AB algorithms are trained on SVS images, their training sets do not include such examples and they fail during inference if a person appears directly under the camera.

All the algorithms attain the lowest performance on videos with tiny projected bodies at field-of-view periphery (e.g., “Exhibition Setup”), distorted image aspect ratio (e.g., “Street Grocery”), or strong camouflage (e.g., “Printing Store”). Note,

Table 7.5: Per-video performance comparison of fisheye people-detection algorithms on WEPDOF.

Algorithm	**Empty Store**	**Exhibition Setup**	**Convenience Store**	**Large Office**	**Warehouse**	**Exhibition**	**Call Center**
[Tamura et al., 2019]	85.3	33.1	91.3	35.1	70.7	90.3	59.3
AA	**96.8**	30.5	92.8	40.3	75.0	89.0	64.9
AB	94.6	**33.2**	92.4	41.0	85.8	89.0	63.4
RAPiD	94.5	32.4	**97.2**	**63.4**	**86.5**	**92.8**	**67.8**

Algorithm	**Tech Store**	**Jewelry Store**	**Street Grocery**	**Printing Store**	**Repair Store**	**IT Office**	**Kindergarten**
[Tamura et al., 2019]	69.2	89.9	12.5	20.4	71.2	60.2	49.1
AA	75.5	86.0	26.4	56.0	**86.0**	70.3	66.3
AB	**76.1**	**92.8**	26.2	**59.4**	85.2	**70.7**	67.3
RAPiD	70.7	78.8	**54.2**	51.5	76.2	65.3	**77.0**

that none of these algorithms leverages temporal information. Using a single frame, the detection of people under severe camouflage or when they are tiny is extremely challenging even for humans (see Figures 7·6g and 7·6h). However, when objects move human vision is capable of recognizing objects in motion even in challenging scenarios. We believe that using temporal information for people detection from OHF cameras should improve RAPiD's performance under challenging circumstances.

7.6 Discussion

In this chapter, we proposed RAPiD, a novel people-detection algorithm for overhead, fisheye images. Our algorithm extends object-detection algorithms which use axis-aligned bounding boxes, such as YOLO, to the case of person detection using human-aligned bounding boxes. We show that our proposed periodic loss function outperforms traditional regression loss functions in bounding-box angle prediction. RAPiD outperforms previous state-of-the-art methods by a large margin without introducing additional computational complexity. As shown in Table 7.1 and Figure 7·6, the performance of RAPiD in staged scenarios with normal-light conditions is nearly perfect. Unsurprisingly, RAPiD's performance drops significantly for videos captured in extremely low-light scenarios, where people are barely distinguishable from the background and also in some of the real-world challenges such as “cropped view”, “tiny people”, “camouflage” and “distorted aspect ratio”. The performance of RAPiD on some of these challenges can be improved by leveraging the temporal dimension in addition to the spatial information. In the next chapter, we propose three improved versions of RAPiD that combine temporal information with spatial information to enhance the people-detection performance from OHF videos.

Chapter 8

Leveraging Temporal Information for People Detection From Overhead Fisheye Cameras

In Chapters 6 and 7, we introduced three novel people-detection algorithms for overhead fisheye cameras that outperform state of the art. As discussed in Section 7.6, the performance of our best-performing algorithm, RAPiD, in staged scenarios recorded under normal-light conditions is extremely good. However, its performance significantly decreases on WEPDOF videos recorded *in the wild* even under normal-light conditions. By evaluating its performance in detail (Section 7.5.2), we realized that one of the deficiencies of RAPiD is the independent inference applied to individual video frames rather than a group of consecutive frames.

Recent research on video-object detection demonstrates that that an algorithm's performance can be significantly improved by leveraging temporal information [Zhu et al., 2017, Zhang et al., 2018, Lin et al., 2019, Liu et al., 2019, Wu et al., 2019, Sabater et al., 2020, Han et al., 2020, Chen et al., 2020]. We adopt this approach and introduce 3 extensions to RAPiD that combine spatial and temporal information to boost algorithm's performance. In one approach, we apply RAPiD to the individual frames and use the temporal information in a post-processing step called Robust and Efficient Post-Processing (REPP) [Sabater et al., 2020]. In an alternative approach, we slightly change the network architecture of RAPiD to combine the spatial and temporal information in an end-to-end method. We apply feature aggregation with adaptive weights, introduced in [Zhu et al., 2017], which combines feature maps derived from past, current and future frames to detect people in the current frame. In yet another approach, we combine RAPiD with Flow-Guided Feature Aggregation (FGFA) [Zhu et al., 2017], which extends our second approach, by warping the feature maps derived from past and future frames using optical flow.

8.1 RAPiD+REPP

REPP [Sabater et al., 2020] is a post-processing methodology designed for object-detection algorithms that produces regular bounding boxes (aligned with image axes). It uses a learning-based similarity function to link bounding boxes in consecutive frames and produce the so-called *object tubelets* (known earlier as *object tunnels* [Ristivojevic and Konrad, 2006]). This is followed by a refinement step which smooths the confidence score, location and size of the bounding boxes within tubelets. This, effectively, increases the confidence scores of weaker detections and decreases those of stronger ones. In this section, we introduce RAPiD+REPP which applies post-processing similar to that of REPP to bounding boxes detected by RAPiD. The post-processing consists of two steps explained next.

8.1.1 Construction of Bounding-Box Tubelets

Thus far, the bounding boxes and their confidence scores were computed independently for each video frame. In order to group together the most similar bounding boxes from consecutive frames, we define a similarity function between bounding boxes and a greedy algorithm to perform this grouping. Let's assume that a video consists of N frames and the i^{th} frame has K_i bounding boxes. We denote the k^{th} bounding box in the i^{th} frame as $bb_i^k = (x_i^k, y_i^k, w_i^k, h_i^k, \alpha_i^k)$ where x_i^k, y_i^k represents the spatial location of the center of the bounding box; w_i^k, h_i^k – width and height of the bounding box and α_i^k – counterclockwise rotation angle of the bounding box.

We propose a learning-based similarity function, inspired by [Sabater et al., 2020], which uses the following features of bounding boxes bb_i^k and bb_j^l to compute a similarity score in $[0, 1]$ range:

- Euclidean distance between their centers: $\sqrt{(x_i^k - x_j^l)^2 + (y_i^k - y_j^l)^2}$,
- ratios of their widths and heights: w_i^k / w_l^j and h_i^k / h_l^j,
- absolute difference between their angles: $|\alpha_i^k - \alpha_l^j|$,
- Intersection over Union (IoU) ratio between them.

We use this similarity function to match bounding boxes between the consecutive frames of a video be means of a greedy graph-matching algorithm. Figure 8·1 shows an example of greedy matching. We start by creating a similarity matrix between bounding box pairs in frames i and $i + 1$. Then, we mark the bounding box pair with the highest similarity score as matched and eliminate the corresponding row and column from the similarity matrix. This process is repeated until there are no more bounding boxes left in one of the frames. To account for occlusions, false positive detections, misses etc., we remove the matched bounding-box pairs with low similarity scores from the set of matches. As a similarity threshold we use 0.1 in our experiments.

Using the bounding-box matching algorithm described above, bounding-box tubelets of a video are formed as follows:

1. intitialize a set of tubelets as an empty set and frame number i as 1,
2. apply the greedy matching algorithm (example shown in Figure 8·1) to find matching bounding-box pairs between frames i and $i + 1$,

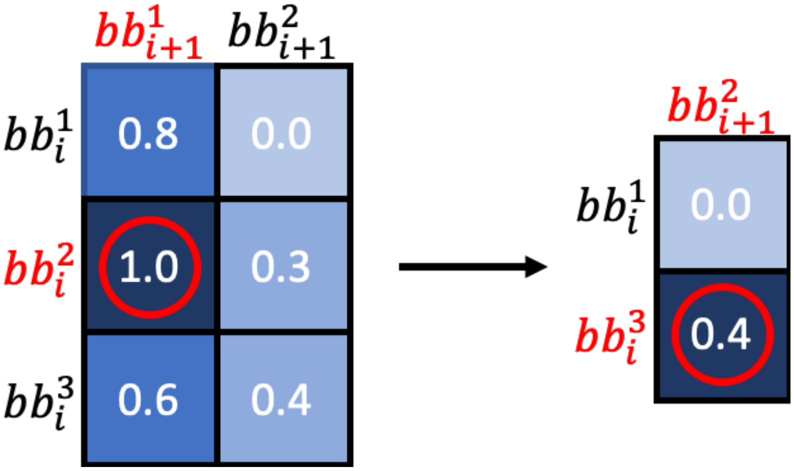


Figure 8·1: An example of greedy bounding-box matching. Frame i has 3 bounding boxes and frame $i+1$ has 2. Similarity scores between the rotated bounding boxes of frame i and frame $i+1$ are given in the matrix on the left. The greedy algorithm will match bb^2_i with bb^1_{i+1} since they have the largest similarity score. Then, the first column and the second row will be eliminated and the similarity matrix will be reduced to a 2×1 matrix as shown on the right. Finally, bb^3_i will be matched with bb^2_{i+1} and the matching algorithm will terminate since there are no more boxes left in frame $i+1$.

3. for each matched bounding-box pair, $(bb^k_i,\ bb^l_{i+1})$, if bb^k_i exists in one of the tubelets, add bb^l_{i+1} to that tubelet as well; otherwise, start a new tubelet with bb^k_i and bb^l_{i+1},

4. increase i and return to step 2 until there are no more frames in the video.

Training of the Similarity Function

In order to train the similarity function described above, we need a set of matched and unmatched bounding box pairs. We use bounding boxes from CEPDOF to form our training set. We randomly select $10,000$ positive and $10,000$ negative bounding box pairs. A positive bounding-box pair is defined as two bounding boxes of the same person taken from the same video that are at most 5 frames apart. All of the other bounding-box pairs are considered as negative examples. We train the similarity function using logistic regression.

8.1.2 Refinement Network

After forming tubelets, we post-process the bounding boxes within each tubelet to improve robustness. We replace the confidence score of all bounding boxes within each tubelet by the average of their confidence scores. We also smooth out the center coordinates and sizes of all bounding boxes within each tubelet using a 1D Gaussian filter with a standard deviation of 0.6 as suggested in [Sabater et al., 2020].

8.2 RAPiD+FGFA

FGFA is an end-to-end video object-detection algorithm which aggregates feature maps computed from past, current and future frames for inference in the current frame. It consists of three parts. The first part is a "feature extraction network" which computes a feature map for each video frame. The second part is a "flow-guided feature aggregation" block. It uses optical flow to warp feature maps of several past and future frames into a single aggregate feature map. Finally, a "detection network" predicts bounding boxes for the current frame based on the output of the aggregation step.

FGFA uses backward motion compensation to warp the feature map of the j^{th} frame (f_j) to the i^{th} frame as follows:

$$f_{j\to i} = \mathcal{W}(f_j, \mathbf{M}_{i\to j}) \tag{8.1}$$

where $\mathcal{W}(.)$ is a warping function with bilinear interpolation, $\mathbf{M}_{i\to j}$ is the predicted optical flow field from frame i to j and $f_{j\to i}$ represents f_j warped to frame i. For computing $\mathbf{M}_{i\to j}$, a neural network called FlowNet [Dosovitskiy et al., 2015] is used.

Then, an aggregate of the warped features is computed as follows:

$$\bar{f}_i = \sum\nolimits_{j=i-K}^{i+K} w_{j\to i} f_{j\to i} \tag{8.2}$$

where K represents the number of past and future frames to be aggregated and $w_{j\to i}$ are adaptive weights. In order to compute weights $w_{j\to i}$, feature maps f_i and $f_{j\to i}$ are fed into a shallow neural network to produce outputs f_i^{ϵ} and $f_{j\to i}^{\epsilon}$, respectively. Then, at each spatial location p, $w_{j\to i}(p)$ is computed as the cosine similarity between $f_i^{\epsilon}(p)$ and $f_{j\to i}^{\epsilon}(p)$ followed by *SoftMax* function to normalize the weights as follows:

$$\bar{w}_{j\to i}(p) = exp\Big(\frac{f_i^{\epsilon}(p) \cdot f_{j\to i}^{\epsilon}(p)}{\|f_i^{\epsilon}(p)\| \|f_{j\to i}^{\epsilon}(p)\|}\Big), \quad w_{j\to i}(p) = \frac{\bar{w}_{j\to i}(p)}{\sum_{j'=i-K}^{i+K}(\bar{w}_{j'\to i}(p))} \tag{8.3}$$

In the last step, $\bar{f}_i$ (8.2) is fed into the "detection network" to predict bounding boxes in the i^{th} frame.

Following the ideas introduced in FGFA [Zhu et al., 2017] and summarized above, we introduce RAPiD+FGFA which applies temporal aggregation to each of the 3 feature maps generated by the "backbone" network of RAPiD (P_1, P_2 and P_3 in Figure 7·2). We use the Farnebäck optical-flow algorithm [Farnebäck, 2003] since we found that it performs better than FlowNet for OHF videos (FlowNet was trained on standard videos).

We also introduce RAPiD+FA, which applies feature aggregation with adaptive weights, but without feature warping, i.e., $f_{j\to i} = f_j$.

8.3 Experimental Results

Table 8.1 compares the performance of algorithms proposed in this chapter with state-of-the-art algorithms on WEPDOF. The proposed extensions of RAPiD achieve $2-6\%$ better AP_{50} score than the original version. This demonstrates the importance of temporal information for people detection.

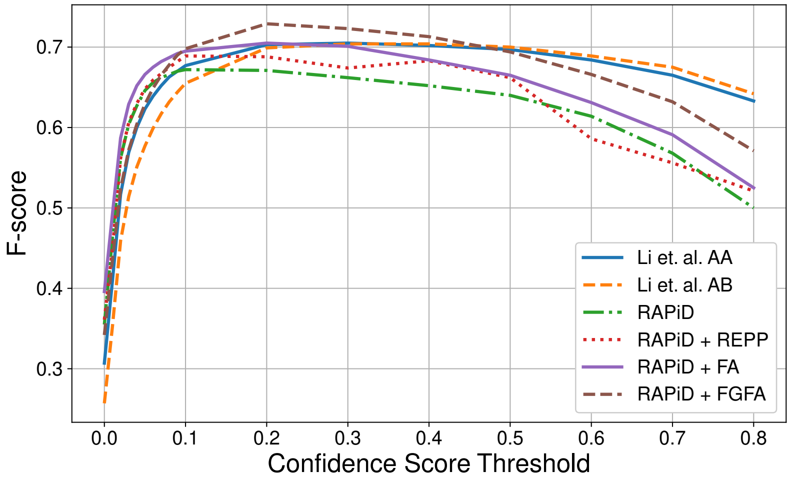
Table 8.1: *In-the-wild* comparison of people-detection algorithms on WEPDOF. The average run-times are computed on a node with a single NVIDIA Tesla V100 GPU.

Algorithm	AP_{50}	AP_{50}^{S}	AP_{50}^{M}	AP_{50}^{L}	P	R	F	Avg. run-time per frame
[Tamura et al., 2019]	59.8	11.6	65.2	61.3	0.777	0.508	0.581	**98 ms**
AA	68.3	11.4	70.1	63.7	0.804	0.647	0.705	1477 ms
AB	69.8	15.8	71.3	63.1	**0.818**	0.643	0.702	1776 ms
RAPiD	72.0	18.4	72.8	67.9	0.731	0.676	0.668	118 ms
RAPiD + REPP	73.7	19.8	74.2	70.2	0.794	0.679	0.703	1667 ms
RAPiD + FA	75.6	19.6	77.5	71.8	0.784	0.672	0.689	269 ms
RAPiD + FGFA	**76.6**	**20.9**	**77.9**	**72.0**	0.803	**0.691**	**0.725**	300 ms

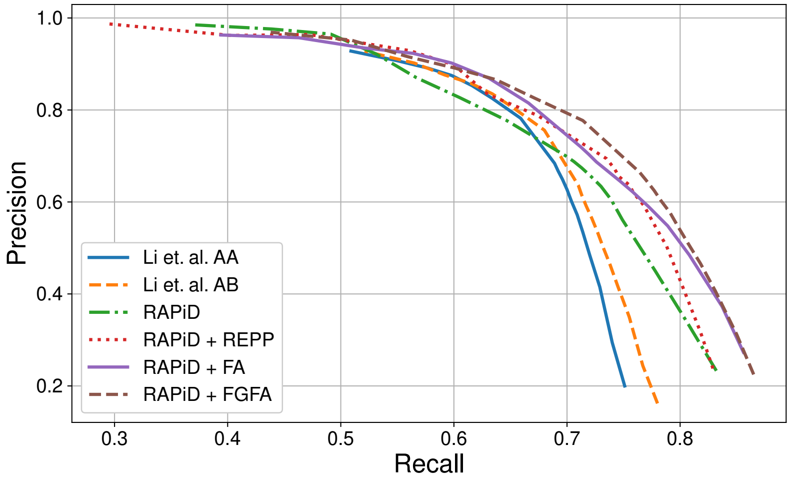
In particular, REPP improves the bounding boxes produced by RAPiD by changing their confidence scores, locations and sizes, but does not introduce new detections that are not produced by RAPiD. Thus, its performance gain is limited.

RAPiD+FA outperforms RAPID+REPP by using an end-to-end integration of the temporal information and RAPiD+FGFA performs even better with the help of optical flow. Figure 8·2 shows that spatio-temporal extensions of RAPiD outperform RAPiD for nearly all confidence score thresholds. In Section 7.5, we discussed the *F-score* versus AP_{50} trade-off between RAPiD and our rotating-window approaches, AA and AB. Although RAPiD achieves a higher AP_{50} score, it is outperfromed by the rotating-window approaches in terms of *F-score*. Figure 8·2a shows that the best *F-score* achieved by RAPiD+FGFA outperforms the one achieved by AA and AB. However, AA and AB still outperform the spatio-temporal extensions of RAPiD for high confidence-score thresholds (see Section 7.5 for the explanation).

Performance gains of the proposed extensions come with a trade-off in terms of efficiency (see Table 8.1). When applied to regular bounding boxes, REPP is proven to be very efficient with just a few of milliseconds of extra computation time per frame [Sabater et al., 2020]. During inference, REPP computes IoU between all pairs of bounding-box predictions in consecutive frames. This computation can be done



(a)



(b)

Figure 8·2: Comparison of people-detection algorithms on WEPDOF in terms of: (a) *F-score* versus confidence-score thresholds; and (b) *Precision* versus *Recall*. Tamura *et al.* [Tamura et al., 2019] is omitted since its performance is significantly lower.

very efficiently for regular bounding boxes but requires computationally expensive geometric libraries for rotated bounding boxes making it inefficient.

For all of the reported algorithms, AP_{50}^{S} is 5-6 times lower than AP_{50}^{M} and AP_{50}^{L}. Both MS COCO and fisheye people-detection datasets used for training are very limited in terms of small bounding boxes and this makes it challenging for learning-based algorithms to predict small bounding boxes. Clearly, this is an open research direction for people-detection algorithms from overhead fisheye cameras.

Figure 8·3 shows sample results produced by RAPiD and its spatio-temporal extensions applied to video frames from WEPDOF. Clearly, leveraging temporal information improves the performance on some of the real-life challenges such as severe camouflage ("Exhibition Setup") and very small body projections ("Tech Store" and "Warehouse"). In the result from "Tech Store", RAPiD produced two false detections in the center of the frame. One of them was corrected by all three proposed extensions and the other by two of them. Usually, this kind of a false detection happens in RAPiD with a low confidence score. In most frames, the score is below a set threshold and no person detection occurs. However, in some frames the confidence score exceeds the set threshold resulting in intermittent false detections. The spatio-temporal versions of RAPiD help smooth out the confidence score temporally thus reducing a chance of a false detection. An analogous observation can be made with respect to missed detections (false negatives) in "Exhibition Setup" and "Warehouse" in Figure 8·3.

In Table 8.2, we report per-video AP_{50} scores of the reported algorithms on WEPDOF. Although spatio-temporal algorithms outperform the spatial-only algorithms for most of the videos, there is no single best algorithm for *all* the videos. Even the improved scores of the reported algorithms are not satisfactory for videos with tiny projected bodies at field-of-view periphery (e.g., "Exhibition Setup"), distorted

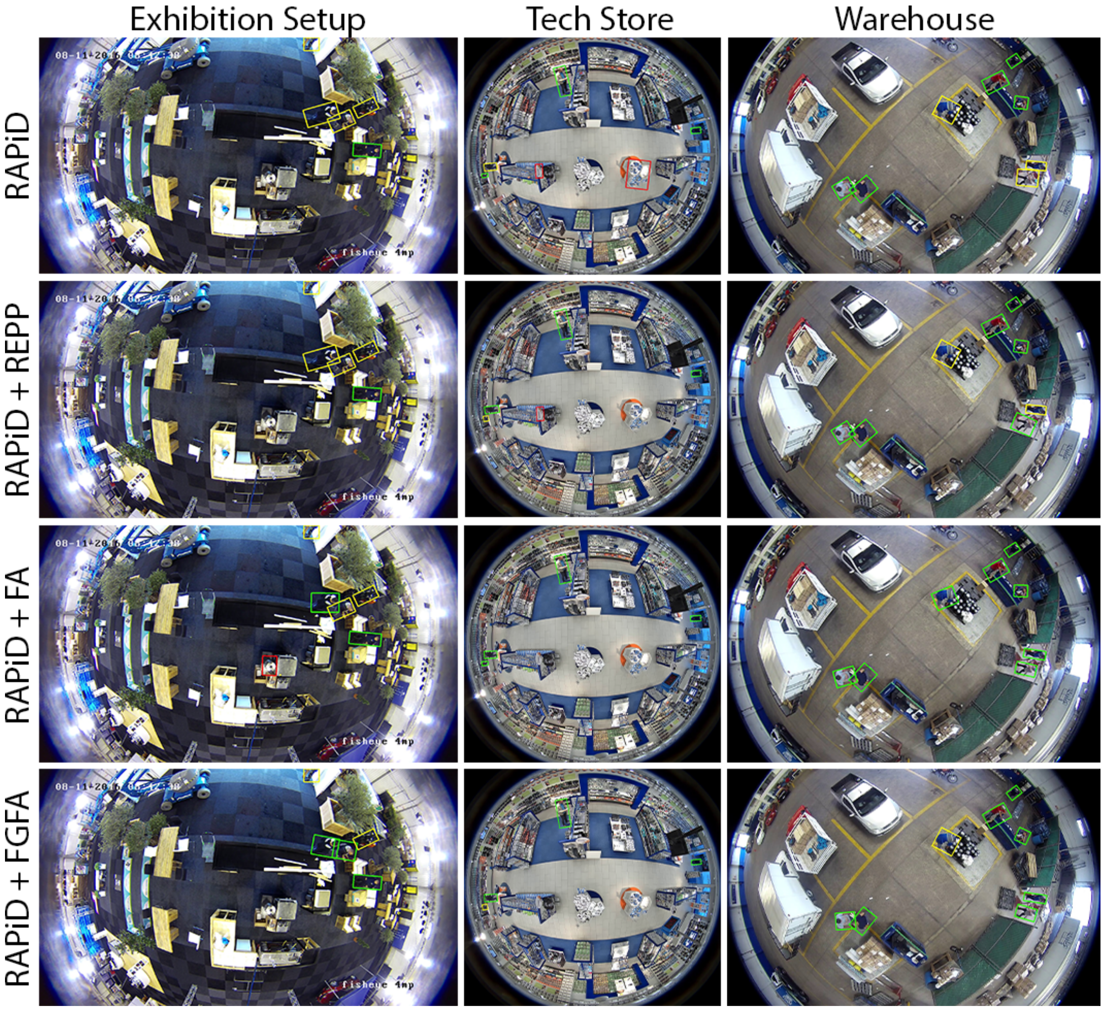


Figure 8·3: Qualitative results of RAPiD and its spatio-temporal extensions on videos from WEPDOF. Green boxes are true positives, red boxes are false positives, and yellow boxes are false negatives.

Table 8.2: Per-video performance comparison of fisheye people-detection algorithms on WEPDOF.

Algorithm	**Empty Store**	**Exhibition Setup**	**Convenience Store**	**Large Office**	**Warehouse**	**Exhibition**	**Call Center**
[Tamura et al., 2019]	85.3	33.1	91.3	35.1	70.7	90.3	59.3
AA	**96.8**	30.5	92.8	40.3	75.0	89.0	64.9
AB	94.6	33.2	92.4	41.0	85.8	89.0	63.4
RAPiD	94.5	32.4	97.2	63.4	86.5	92.8	67.8
RAPiD+REPP	95.3	34.4	**97.7**	66.3	87.6	**94.2**	68.4
RAPiD+FA	91.7	**45.4**	95.9	71.4	**89.0**	91.7	74.5
RAPiD+FGFA	93.5	40.1	96.4	**77.3**	87.5	91.2	**75.4**

Algorithm	**Tech Store**	**Jewelry Store**	**Street Grocery**	**Printing Store**	**Repair Store**	**IT Office**	**Kindergarten**
[Tamura et al., 2019]	69.2	89.9	12.5	20.4	71.2	60.2	49.1
AA	75.5	86.0	26.4	56.0	**86.0**	70.3	66.3
AB	76.1	92.8	26.2	**59.4**	85.2	**70.7**	67.3
RAPiD	70.7	78.8	54.2	51.5	76.2	65.3	77.0
RAPiD+REPP	71.8	83.0	**58.4**	53.8	77.0	65.8	78.2
RAPiD+FA	79.2	**94.1**	57.3	45.1	76.2	65.8	81.4
RAPiD+FGFA	**80.2**	93.2	54.0	55.6	78.9	67.4	**81.8**

image aspect ratio (e.g., “Street Grocery”), and strong camouflage (e.g., “Printing Store”). We believe the performance on these challenges can be further improved by developing algorithms that address them directly (e.g., data augmentation to mimic these challenges in the training set).

8.4 Discussion

In this chapter, we introduced the first ever spatio-temporal people-detection algorithms from overhead, fisheye cameras by combining RAPiD (Chapter 7) with state-of-the-art video-object-detection algorithms. We demonstrated that leveraging temporal information significantly improves the people-detection performance. The proposed extensions of RAPiD outperformed state-of-the-art algorithms on nearly all of the evaluation metrics we have tested.

Even the best-performing algorithm, RAPiD+FGFA, is far from perfect on our *in-the-wild* dataset, WEPDOF, with an AP_{50} score of 76.6%. We conducted a detailed analysis of the results to pinpoint deficiencies of the tested algorithms. In terms of people, very small body size and camouflage turned out to be the main bottlenecks. Also, a cropped or distorted camera field of view proved challenging. In addition to these challenges, computational efficiency is another direction for future research – the current algorithms are far from real-time execution. Further advances in these areas are needed before people detection from OHF cameras becomes a reliable tool in practice.

Chapter 9

Conclusions and Future Directions

In this dissertation, we proposed novel deep-learning algorithms for two different applications of video analysis: (i) background subtraction and (ii) people detection from overhead fisheye cameras.

In terms of background subtraction, we focused on supervised deep-learning algorithms designed for unseen videos and we proposed *video-agnostic* evaluation methodologies that treat each video in a dataset as unseen. The first algorithm that we introduced is called Background Subtraction for Unseen Videos (BSUV-Net). One of the main novelties of BSUV-Net is the use of background models from different time scales, in addition to the current frame, as an input to the network. Another novelty is a temporal data augmentation that we introduced to mimic illumination changes commonly occurring in real-world videos. Experimental results on CDNet-2014 [Goyette et al., 2014] show that BSUV-Net outperforms state-of-the-art *video-agnostic* BGS algorithms and its performance can be further improved by adding SemanticBGS [Braham et al., 2017] as a post-processing layer.

In follow-up work, we introduced BSUV-Net 2.0 which improves BSUV-Net by applying several spatio-temporal data augmentations to synthetically increase the number of inputs mimicking real-world challenges. Specifically, we introduced new augmentations for PTZ, camera jitter and intermittent object motion scenarios, and achieved significant performance improvements in these categories and, consequently, a better overall performance on CDNet-2014 dataset. We also showed that BSUV-Net

2.0 can be simplified by skipping the semantic segmentation network which makes it possible to run the algorithm in real-time while performing better than state-of-the-art methods. Finally, we demonstrated a strong generalization capacity of BSUV-Net 2.0 using cross-dataset evaluation on LASIESTA in which it significantly outperforms the current state-of-the-art methods on a completely unseen dataset.

A *video-agnostic* evaluation of supervised BGS algorithms requires a distinct set of training and testing videos. However, the evaluation server of CDNet-2014 ranks the algorithms based on their average performance on *all* of its videos. In order to compare our algorithms with state of the art reported on this evaluation server in a *video-agnostic* manner, we introduced a 4-fold cross-validation data split for CDNet-2014. We hope that the introduced cross-validation strategy will provide an easy and fair comparison mechanism for supervised BGS algorithms developed in the future.

In the second part of this dissertation (Chapters 5-8), we introduced multiple algorithms and datasets for people detection from overhead fisheye cameras. Due to the lack of prior datasets with rotated bounding boxes, we collected three new datasets with overhead fisheye videos and annotated them with rotated bounding boxes tightly drawn around each person. We also re-annotated a subset of the Mirror Worlds dataset [Ma et al., 2018b] with rotated bounding boxes. While three of the four datasets were recorded in staged scenarios, the fourth one consists of YouTube videos with many real-world challenges. By reporting 7 performance metrics and 2 performance trade-off plots, we offered a detailed analysis of the strengths and weaknesses of our people-detection algorithms.

The first two algorithms, that we introduced for people detection from OHF cameras, apply blindly (Activity-Blind or AB method) or selectively (Activity-Aware or AA method) a state-of-the-art object-detection method to overlapping rotated windows. These algorithms were motivated by the observation that orientations of people in the top-center part of overhead fisheye images are similar to those in standard side-view images. In AB, we applied a state-of-the-art object-detection algorithm, YOLO v3 trained on standard images, to the top-center part of an overhead fisheye image and rotated the image 24 times to cover the entire field of view. In AA, we used background subtraction as a pre-processing step to find the windows of interest in the frame and applied YOLO v3 only to these windows. Through numerous experiments, we showed that both approaches outperformed state of the art at the time, at the cost of very high computational complexity due to the multiple applications of YOLO v3.

In follow-up work, we introduced an end-to-end algorithm by extending YOLO v3 to predict arbitrarily-oriented bounding boxes. We proposed a novel 5-D parameterization of rotated bounding boxes and a novel periodic loss function. Our algorithm, Rotation-Aware People Detection in overhead fisheye images (RAPiD), significantly outperforms state of the art, including the AA and AB algorithms. We further improved RAPiD by leveraging temporal information. We introduced three spatio-temporal extensions of RAPiD and demonstrated their improved performance compared to RAPiD and the state of the art.

9.1 Future Directions

In this section, based on the work introduced in this dissertation, we discuss several research directions that are worth pursuing.

9.1.1 Background Subtraction

Video-Agnostic Benchmarking of Supervised BGS Algorithms

For video analysis tasks, such as background subtraction, a *video-agnostic* evaluation of data-driven algorithms requires that datasets be split into separate training and test sets composed of complete videos (videos should not be divided to be used in both sets). Ideally, the annotations of test videos should be kept private to ensure a fair comparison. However, as discussed in Section 3.7, existing BGS datasets provide annotations from each video and compare the algorithms based on their performance on the full dataset. Several researchers, including ourselves, reported *video-agnostic* comparison of their algorithms using different training/testing or cross-validation splits. Due to inconsistencies between these evaluation schemes, a detailed comparison of state-of-the-art BGS algorithms has not been completed to date. This issue can be addressed by implementing and benchmarking supervised BGS algorithms using a fixed and unbiased evaluation scheme. This will also encourage future researchers to use the same evaluation scheme to compare their algorithms against state of the art. Another direction is to introduce a new background subtraction dataset, aimed at testing supervised algorithms, with private annotations. This would enable performance evaluation without allowing any training on the annotations of the test data.

Network Architecture

In this dissertation, we kept the network design of our algorithms relatively simple and focused on leveraging temporal information for BGS and on data augmentation techniques to improve robustness to various challenges. The recent advances in neural network design can be explored to improve the performance or efficiency of our approaches [Howard et al., 2017, Sandler et al., 2018, Tan and Le, 2019].

Unsolved BGS Challenges

The performance of BGS algorithms in challenging scenarios, such as night videos and moving cameras, is still insufficient for real-world applications. Further research is needed to accommodate such scenarios. One of the key limitations is the amount of labeled data with various challenges. For example, CDNet2014 has at most 6 videos for each challenge type which makes it difficult for data-driven approaches to successfully learn these challenges. An extensive collection and annotation of training data under various challenges would help with the development of more robust algorithms.

Domain Adaptation

Although the publicly-available BGS datasets include hundreds of thousands of labeled video frames, the number of different scenes is quite limited. For example, CDNet-2014 is comprised of videos depicting 53 different scenes. This limited variety of scene scenarios is insufficient to train a generalizable supervised algorithm. In this dissertation, we addressed this problem by extending the dataset using spatio-temporal data augmentations. Another approach is to apply *domain adaption* (DA) [Wang and Deng, 2018, Zhuang et al., 2021]. Given the target domain of a problem of interest (e.g., background subtraction), the basic idea in DA is to find a source domain of another problem with a large labeled dataset that in some way is similar to the problem of interest and to leverage this similarity. In the case of background subtraction, there exist similar but more extensive and diverse datasets. For example, YouTube - Video Instance Segmentation (VIS) [Yang et al., 2019a] dataset consists of labeled frames from 2,883 different videos. In VIS, the goal is to detect, segment and track every object instance in a given video, so it is very similar to BGS which aims to detect foreground pixels. In this case, one could consider YouTube-VIS as a labeled source domain and CDNet-2014 as a labeled target domain, and leverage the similarity between the two datasets/problems to improve background subtraction.

One could also consider unsupervised domain adaption. The basic idea is that every time a security camera is mounted at a new location, the scene will be different than those used during training. In this case, one could perform a scene-specific calibration as follows. After the camera is mounted, unlabeled frames can be collected during the first few hours or days, and these data might be used to tackle the domain shift between the new scene and the scenes in the training dataset via unsupervised domain adaptation. For example, a common methodology used in both supervised and unsupervised domain adaption is to reduce the difference between the training and test domain distributions by using a domain discrepancy loss [Long et al., 2015] or a discriminator network [Tzeng et al., 2017].

9.1.2 People Detection from Overhead Fisheye Cameras

Fisheye-Lens Distortions

In people detection algorithms, we focused on the radial geometry of OHF images by producing rotated bounding boxes and introducing an angle-aware loss function (see Section 7.3). Another challenging property of OHF cameras are distortions due to the wide-angle fisheye lens; objects appear larger in the center of the image and are radially-compressed at field-of-view periphery. This property can be leveraged for people detection algorithms. For example, during training false detections with large bounding boxes produced at the field-of-view periphery and with small bounding boxes produced in image center can be penalized more than other false detections. Alternatively, these distortions can be modeled and applied during data augmentation to improve networks' robustness. Recently, Tamura *et al.* used a fisheye camera model to transform standard-lens images to fisheye-lens-like images and used these transformed images in training [Tamura and Yoshinaga, 2021]. This idea can be extended further to augment the overhead images captured by a fisheye lens to look like captured by a different fisheye lens (different distortion parameters) and/or installed at different height. A supervised algorithm trained using such an augmentation would likely be more robust to different hardware and installation heights.

Unsolved Challenges

As discussed in Section 8.3, people who appear very small in fisheye images, severe camouflage and cropped/distorted field of view are still challenging for even the best-performing people-detection algorithms. Further advances in these areas are needed before people detection from overhead fisheye cameras becomes a reliable tool in practice. The cropped and distorted fields of view can be easily incorporated into data augmentation to improve performance. The camouflage effects can be mimicked to some degree by decreasing image contrast. The detection of small projections of people is not straightforward in terms of data augmentation. An alternative approach might be to consider small ground-truth bounding boxes as hard examples and weight their loss values higher than those of the easier examples.

Domain Adaptation

The domain adaptation techniques that we discussed for BGS can be easily extended to people detection from overhead fisheye cameras. There exist several video-object detection datasets with significantly more videos than our OHF datasets. For example, ImageNet VID [Russakovsky et al., 2015] consists of about 2,000 videos labeled with image-axis aligned bounding boxes. Also, the video-instance segmentation datasets such as YouTube-VIS [Yang et al., 2019a] can be used to construct bounding box annotations. These extensive datasets can be leveraged in a supervised domain adaptation setting to improve the performance of people detection algorithms from OHF cameras.

People Tracking and Re-Identification from Overhead Fisheye Cameras

Three of the people detection datasets that we introduced for OHF cameras (MW-R, CEPDOF and WEPDOF) are annotated spatio-temporally and thus can be used for people tracking and re-identification. Although these tasks have been explored in depth for standard cameras [Zheng et al., 2015, Milan et al., 2016, Muller et al., 2018, Luo et al., 2020, Ye et al., 2021], to the best of our knowledge there is no recent work on person tracking or re-identification from OHF cameras. A combination of the people-detection algorithms introduced in this dissertation with some state-of-the-art person tracking or re-identification networks could be a good starting point.

References

Babaee, M., Dinh, D. T., and Rigoll, G. (2018). A deep convolutional neural network for video sequence background subtraction. *Pattern Recognition*, 76:635–649.

Bakkay, M., Rashwan, H., Salmane, H., Khoudour, L., Puigtt, D., and Ruichek, Y. (2018). BSCGAN: Deep background subtraction with conditional generative adversarial networks. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 4018–4022.

Barnich, O. and Van Droogenbroeck, M. (2011). ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6):1709–1724.

Belhassen, H., Zhang, H., Fresse, V., and Bourennane, E.-B. (2019). Improving video object detection by Seq-Bbox matching. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP*, pages 226–233.

Berjón, D., Cuevas, C., Morán, F., and Garcia, N. (2018). Real-time nonparametric background subtraction with tracking-based foreground update. *Pattern Recognition*, 74:156–170.

Bianco, S., Ciocca, G., and Schettini, R. (2017). How far can you get by combining change detection algorithms? In *Image Analysis and Processing - ICIAP 2017*, pages 96–107. Springer.

Bilodeau, G.-A., Jodoin, J.-P., and Saunier, N. (2013). Change detection in feature space using local binary similarity patterns. In *2013 International Conference on Computer and Robot Vision*, pages 106–112.

Bouwmans, T., Javed, S., Sultana, M., and Jung, S. K. (2019). Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Networks*, 117:8–66.

Braham, M., Piérard, S., and Van Droogenbroeck, M. (2017). Semantic background subtraction. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 4552–4556.

Braham, M. and Van Droogenbroeck, M. (2016). Deep background subtraction with scene-specific convolutional neural networks. In *2016 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 1–4.

Brunetti, A., Buongiorno, D., Trotta, G. F., and Bevilacqua, V. (2018). Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing*, 300:17–33.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

Chen, Y., Cao, Y., Hu, H., and Wang, L. (2020). Memory enhanced global-local aggregation for video object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10334–10343.

Chiang, A.-T. and Wang, Y. (2014). Human detection in fish-eye images using hog-based detectors over rotated windows. In *2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6.

Cuevas, C., Yáñez, E. M., and García, N. (2016). Labeled dataset for integral evaluation of moving object detection algorithms: LASIESTA. *Computer Vision and Image Understanding*, 152:103–117.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893.

del Blanco, C. R., Carballeira, P., Jaureguizar, F., and García, N. (2016). PIROPO database. `sites.google.com/site/piropodatabase/`.

del Blanco, C. R., Carballeira, P., Jaureguizar, F., and García, N. (2021). Robust people indoor localization with omnidirectional cameras using a grid of spatial-aware classifiers. *Signal Processing: Image Communication*, 93:116135.

Demiröz, B. E., Ari, I., Eroğlu, O., Salah, A. A., and Akarun, L. (2012). Feature-based tracking on a multi-omnidirectional camera dataset. In *2012 5th International Symposium on Communications, Control and Signal Processing*, pages 1–5.

Ding, J., Xue, N., Long, Y., Xia, G.-S., and Lu, Q. (2019). Learning ROI transformer for oriented object detection in aerial images. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2844–2853.

Dollár, P., Appel, R., Belongie, S., and Perona, P. (2014). Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. (2015). FlowNet: Learning optical flow with convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2758–2766.

Duan, Z., Tezcan, M. O., Nakamura, H., Ishwar, P., and Konrad, J. (2020). RAPiD: Rotation-aware people detection in overhead fisheye images. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2700–2709.

Elgammal, A., Duraiswami, R., Harwood, D., and Davis, L. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163.

Enzweiler, M. and Gavrila, D. M. (2009). Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195.

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pages 363–370. Springer.

Fu, C.-Y., Liu, W., Ranga, A., Tyagi, A., and Berg, A. C. (2017). DSSD: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*.

Girshick, R. (2015). Fast R-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448.

Goyette, N., Jodoin, P.-M., Porikli, F., Konrad, J., and Ishwar, P. (2014). A novel video dataset for change detection benchmarking. *IEEE Transactions on Image Processing*, 23(11):4663–4679.

Haines, T. S. and Xiang, T. (2014). Background subtraction with dirichletprocess mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):670–683.

Han, M., Wang, Y., Chang, X., and Qiao, Y. (2020). Mining inter-video proposal relations for video object detection. In *Computer Vision – European Conference on Computer Vision (ECCV) 2020*, pages 431–446. Springer.

Han, W., Khorrami, P., Paine, T. L., Ramachandran, P., Babaeizadeh, M., Shi, H., Li, J., Yan, S., and Huang, T. S. (2016). Seq-NMS for video object detection. *arXiv preprint arXiv:1602.08465*.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861.*

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167.*

Işık, Ş., Özkan, K., Günal, S., and Gerek, Ö. N. (2018). SWCD: A sliding window and self-regulated learning-based background updating method for change detection in videos. *Journal of Electronic Imaging*, 27(2):023002.

Kim, J.-Y. and Ha, J.-E. (2020). Foreground objects detection using a fully convolutional network with a background model image and multiple original images. *IEEE Access*, 8:159864–159878.

Krams, O. and Kiryati, N. (2017). People detection in top-view fisheye imaging. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6.

Lee, S.-H., Kwon, S.-C., Shim, J.-W., Lim, J.-E., and Yoo, J. (2018). WisenetMD: Motion detection using dynamic background region analysis. *arXiv preprint arXiv: 1805.09277.*

Li, S., Tezcan, M. O., Ishwar, P., and Konrad, J. (2019). Supervised people counting using an overhead fisheye camera. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8.

Lim, L. A. and Keles, H. Y. (2018a). Foreground segmentation using convolutional neural networks for multiscale feature encoding. *Pattern Recognition Letters*, 112:256–262.

Lim, L. A. and Keles, H. Y. (2018b). Learning multi-scale features for foreground segmentation. *arXiv preprint arXiv:1808.01477.*

Lin, J., Gan, C., and Han, S. (2019). TSM: Temporal shift module for efficient video understanding. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7082–7092.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017a). Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017b). Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *Computer Vision – European Conference on Computer Vision (ECCV) 2014*, pages 740–755. Springer.

Liu, M., Zhu, M., White, M., Li, Y., and Kalenichenko, D. (2019). Looking fast and slow: Memory-guided mobile video object detection. *arXiv preprint arXiv:1903.10172*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single shot multibox detector. In *Computer Vision – European Conference on Computer Vision (ECCV) 2016*, pages 21–37. Springer.

Long, M., Cao, Y., Wang, J., and Jordan, M. (2015). Learning transferable features with deep adaptation networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 97–105. JMLR.

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., and Kim, T.-K. (2020). Multiple object tracking: A literature review. *Artificial Intelligence*, page 103448.

Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., and Xue, X. (2018a). Arbitrary-oriented scene text detection via rotation proposals. *IEEE Transactions on Multimedia*, 20(11):3111–3122.

Ma, N., Knapp, R. B., Polys, N. F., Huang, J.-B., Ibrahim, A., Hurt, C., and Xiao, Y. (2018b). Mirror worlds challenge. `www2.icat.vt.edu/mirrorworlds/challenge/index.html`.

Maddalena, L. and Petrosino, A. (2008). A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 17(7):1168–1177.

Maddalena, L. and Petrosino, A. (2012). The SOBS algorithm: What are the limits? In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 21–26.

Maddalena, L. and Petrosino, A. (2015). Towards benchmarking scene background initialization. In *New Trends in Image Analysis and Processing – ICIAP 2015 Workshops*, pages 469–476. Springer.

Mandal, M., Dhar, V., Mishra, A., and Vipparthi, S. K. (2019). 3DFR: A swift 3D feature reductionist framework for scene independent change detection. *IEEE Signal Processing Letters*, 26(12):1882–1886.

Mandal, M., Dhar, V., Mishra, A., Vipparthi, S. K., and Abdel-Mottaleb, M. (2021). 3DCD: Scene independent end-to-end spatiotemporal feature learning framework for change detection in unseen videos. *IEEE Transactions on Image Processing*, 30:546–558.

Mandal, M. and Vipparthi, S. K. (2020). Scene independency matters: An empirical study of scene dependent and scene independent evaluation for cnn-based change detection. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–14.

Mandal, M. and Vipparthi, S. K. (2021). An empirical review of deep learning frameworks for change detection: Model design, experimental frameworks, challenges and research needs. *arXiv preprint arXiv:2105.01342*.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. (2016). MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*.

Mittal, A. and Paragios, N. (2004). Motion-based background subtraction using adaptive kernel density estimation. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II.

Muller, M., Bibi, A., Giancola, S., Alsubaihi, S., and Ghanem, B. (2018). TrackingNet: A large-scale dataset and benchmark for object tracking in the wild. In *Computer Vision – European Conference on Computer Vision (ECCV) 2018*, pages 300–327.

Nguyen, D. T., Li, W., and Ogunbona, P. O. (2016). Human detection from images and videos: A survey. *Pattern Recognition*, 51:148–175.

Nosaka, R., Ujiie, H., and Kurokawa, T. (2018). Orientation-aware regression for oriented bounding box estimation. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6.

Qian, W., Yang, X., Peng, S., Guo, Y., and Yan, C. (2019). Learning modulated loss for rotated object detection. *arXiv preprint arXiv:1911.08299*.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

Redmon, J. and Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525.

Redmon, J. and Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 91–99. MIT Press, Cambridge, MA, USA.

Ristivojevic, M. and Konrad, J. (2006). Space-time image sequence analysis: Object tunnels and occlusion volumes. *IEEE Transactions on Image Processing*, 15(2):364–376.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241. Springer.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Sabater, A., Montesano, L., and Murillo, A. C. (2020). Robust and efficient post-processing for video object detection. *arXiv preprint arXiv:2009.11050*.

Saito, M., Kitaguchi, K., Kimura, G., and Hashimoto, M. (2011). People detection and tracking from fish-eye image based on probabilistic appearance model. In *Society of Instrument and Control Engineers (SICE) Annual Conference 2011*, pages 435–440.

Sakkos, D., Liu, H., Han, J., and Shao, L. (2018). End-to-end video background subtraction with 3D convolutional neural networks. *Multimedia Tools and Applications*, 77:23023–23041.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). MobileNetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520.

Seidel, R., Apitzsch, A., and Hirtz, G. (2018). Improved person detection on omnidirectional images with non-maxima suppression. *arXiv preprint arXiv:1805.08503*.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

St-Charles, P.-L., Bilodeau, G.-A., and Bergevin, R. (2015a). A self-adjusting approach to change detection based on background word consensus. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 990–997.

St-Charles, P.-L., Bilodeau, G.-A., and Bergevin, R. (2015b). SuBSENSE: A universal change detection method with local adaptive sensitivity. *IEEE Transactions on Image Processing*, 24(1):359–373.

Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 2, pages 246–252 Vol. 2.

Sultana, M., Mahmood, A., Javed, S., and Jung, S. K. (2019). Unsupervised deep context prediction for background estimation and foreground segmentation. *Machine Vision and Applications*, 30:375–395.

Tamura, M., Horiguchi, S., and Murakami, T. (2019). Omnidirectional pedestrian detection by rotation invariant training. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1989–1998.

Tamura, M. and Yoshinaga, T. (2021). Segmentation-based bounding box generation for omnidirectional pedestrian detection. *arXiv preprint arXiv:2104.13764*.

Tan, M. and Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6105–6114. PMLR.

Tan, M., Pang, R., and Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10778–10787.

Taylor, L. and Nitschke, G. (2017). Improving deep learning using generic data augmentation. *arXiv preprint arXiv:1708.06020*.

Tezcan, M. O., Duan, Z., Çokbaş, M., Ishwar, P., and Konrad, J. (2021a). WEPDOF: A dataset and benchmark algorithms for in-the-wild people detection from overhead fisheye cameras. In *2021 IEEE Winter Conference on Applications of Computer Vision (submitted)*.

Tezcan, M. O., Ishwar, P., and Konrad, J. (2019). BSUV-Net: A fully-convolutional neural network for background subtraction of unseen videos. *arXiv preprint arXiv:1907.11371*.

Tezcan, M. O., Ishwar, P., and Konrad, J. (2020). BSUV-Net: A fully-convolutional neural network for background subtraction of unseen videos. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2763–2772.

Tezcan, M. O., Ishwar, P., and Konrad, J. (2021b). BSUV-Net 2.0: Spatio-temporal data augmentations for video-agnostic supervised background subtraction. *IEEE Access*, 9:53849–53860.

Tian, Z., Shen, C., Chen, H., and He, T. (2019). FCOS: Fully convolutional one-stage object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9626–9635.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., and Bregler, C. (2015). Efficient object localization using convolutional networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 648–656.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. (2017). Adversarial discriminative domain adaptation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2962–2971.

Wang, L., Shi, J., Song, G., and Shen, I.-f. (2007). Object detection combining recognition and segmentation. In *Computer Vision – Asian Conference on Computer Vision (ACCV) 2007*, pages 189–199. Springer.

Wang, M. and Deng, W. (2018). Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153.

Wang, T., Chang, C., and Wu, Y. (2017). Template-based people detection using a single downward-viewing fisheye camera. In *2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pages 719–723.

Wang, Y., Luo, Z., and Jodoin, P.-M. (2017). Interactive deep learning method for segmenting moving objects. *Pattern Recognition Letters*, 96:66–75.

Wu, H., Chen, Y., Wang, N., and Zhang, Z. (2019). Sequence level semantics aggregation for video object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9216–9224.

Yang, L., Fan, Y., and Xu, N. (2019a). Video instance segmentation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5187–5196.

Yang, X., Liu, Q., Yan, J., and Li, A. (2019b). R3DET: Refined single-stage detector with feature refinement for rotating object. *arXiv preprint arXiv:1908.05612*.

Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., and Hoi, S. C. (2021). Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zeng, D., Chen, X., Zhu, M., Goesele, M., and Kuijper, A. (2019a). Background subtraction with real-time semantic segmentation. *IEEE Access*, 7:153869–153884.

Zeng, D. and Zhu, M. (2018). Background subtraction using multiscale fully convolutional network. *IEEE Access*, 6:16010–16021.

Zeng, D., Zhu, M., and Kuijper, A. (2019b). Combining background subtraction algorithms with convolutional neural network. *Journal of Electronic Imaging*, 28(1):013011.

Zhang, S., Chi, C., Yao, Y., Lei, Z., and Li, S. Z. (2020). Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9756–9765.

Zhang, Z., Cheng, D., Zhu, X., Lin, S., and Dai, J. (2018). Integrated object detection and tracking with tracklet-conditioned detection. *arXiv preprint arXiv: 1811.11167*.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239.

Zhao, Q., Sheng, T., Wang, Y., Tang, Z., Chen, Y., Cai, L., and Ling, H. (2019). M2Det: A single-shot object detector based on multi-level feature pyramid network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9259–9266.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., and Tian, Q. (2015). Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. (2017a). Scene parsing through ADE20K dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130.

Zhou, Y., Ye, Q., Qiu, Q., and Jiao, J. (2017b). Oriented response networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4961–4970.

Zhu, X., Wang, Y., Dai, J., Yuan, L., and Wei, Y. (2017). Flow-guided feature aggregation for video object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 408–417.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. (2021). A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

Zivkovic, Z. (2004). Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31 Vol.2.

CURRICULUM VITAE

M. Ozan Tezcan
8 St Marys St, Boston, MA 02215
mtezcan@bu.edu

EDUCATION

Boston University, Boston, MA *Sep 2016 – Present*
Ph.D., Electrical and Computer Engineering, GPA – 4.00

Koç University, Istanbul, Turkey *Sep 2011 – May 2016*
B.S., Electrical and Electronics Engineering, GPA – 3.87
Double Major: Mathematics

RESEARCH & TEACHING EXPERIENCE

Graduate Research Assistant *Sep 2016 – Present*
Visual Information Processing Lab,
Department of Electrical and Computer Engineering, Boston University

Graduate Teaching Assistant *Sep 2017 – May 2018*
Department of Electrical and Computer Engineering, Boston University

WORK EXPERIENCE

Software Engineer Intern, Machine Learning *May 2020 – Aug 2020*
Facebook, Menlo Park, CA

Data Science Intern *June 2019 – Sep 2019*
Wayfair, Boston, MA

PUBLICATIONS

- **M. O. Tezcan**, Z. Duan, M. Çokbaş, P. Ishwar and J. Konrad, "WEPDOF: A Dataset and Benchmark Algorithms for In-the-Wild People Detection from Overhead Fisheye Cameras," submitted to *IEEE/CVF Winter Conference on Applications of Computer Vision* (WACV), 2021.

- Q. Zeng, **M. O. Tezcan** and J. Konrad, "Dynamic Equilibrium Module for Action Recognition" submitted to *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- **M. O. Tezcan**, P. Ishwar and J. Konrad, "BSUV-Net 2.0: Spatio-Temporal Data Augmentations for Video-Agnostic Supervised Background Subtraction," in *IEEE Access*, vol. 9, pp. 53849-53860, 2021.
- Z. Duan, **M. O. Tezcan**, H. Nakamura, P. Ishwar and J. Konrad, "RAPiD: Rotation-Aware People Detection in Overhead Fisheye Images," *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW), pp. 2700-2709, 2020.
- **M. O. Tezcan**, P. Ishwar and J. Konrad, "BSUV-Net: A Fully-Convolutional Neural Network for Background Subtraction of Unseen Videos," *IEEE/CVF Winter Conference on Applications of Computer Vision* (WACV), pp. 2774-2783, 2020.
- İ. Yıldız, E. Ataer-Cansızoğlu, H. Liu, P. Golbus, **M. O. Tezcan**, and J.-W. Choi, "Deep Ranking for Style-Aware Room Recommendations (Student Abstract)", *AAAI Conference on Artificial Intelligence Student Tracks*, vol. 34, pp. 13975-13976, 2020.
- S. Li, **M. O. Tezcan**, P. Ishwar and J. Konrad, "Supervised People Counting Using An Overhead Fisheye Camera," *IEEE International Conference on Advanced Video and Signal Based Surveillance* (AVSS), pp. 1-8, 2019.
- **M. O. Tezcan**, J. Konrad and J. Muroff, "Automatic Assessment of Hoarding Clutter from Images Using Convolutional Neural Networks," *IEEE Southwest Symposium on Image Analysis and Interpretation* (SSIAI), pp. 1-4, 2018.
- I. Bostan, O. T. Buruk, M. Canat, **M. O. Tezcan**, C. Yurdakul, T. Göksun, and O. Özcan, "Hands as a Controller: User Preferences for Hand Specific On-Skin Gestures," *ACM Conference on Designing Interactive Systems* (DIS), pp. 1123–1134, 2017.
- M. Canat, **M. O. Tezcan**, C. Yurdakul, O. T. Buruk, and O. Ozcan, "Experiencing Human-to-Human Touch in Digital Games," *ACM CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA), pp. 3655–3658, 2016.
- M. Canat, **M. O. Tezcan**, C. Yurdakul, E. Tiza, B. C. Sefercik, I. Bostan, O. T. Buruk, T. Göksun, and O. Özcan, "Sensation: Measuring the Effects of a Human-to-Human Social Touch Based Controller on the Player Experience," *CHI Conference on Human Factors in Computing Systems* (CHI), pp. 3944–3955, 2016.