

# Object detection in images using deep neural networks and synthetic data in scenarios of partial object occlusion

G.A. Algashev<sup>1</sup>, P.A. Kremushchenko<sup>1</sup>, I.A. Lezin<sup>1</sup>

<sup>1</sup> Samara National Research University, 443086, Russia, Samara, Moskovskoye Shosse 34

## Abstract

This research addresses the problem of automatic object detection in images under limited-visibility conditions, where objects are partially occluded, the background is complex, and lighting and viewpoints vary widely. The proposed approach combines pretraining on a programmatically generated synthetic dataset of 18,000 images – produced using the Visualization Toolkit (VTK) library – with fine-tuning on a compact real-image dataset of 2,000 annotated photographs (500 per class). Six deep neural network architectures – Faster R-CNN ResNet-50 FPN, SSD MobileNet V3, YOLOv11n, EfficientDet-D7, DETR-DC5, and CenterNet – were evaluated across three training regimes: synthetic-only, real-only, and combined (90% synthetic / 10% real). Hybrid training yielded the most substantial improvements: YOLOv11n achieved  $mAP@0.5 = 0.91$  and  $mAP@0.75 = 0.86$  (Precision = 0.89, Recall = 0.90, F1 = 0.89, 82 FPS), compared to 0.79 (synthetic-only) and 0.78 (real-only), representing a gain of up to +15 percentage points in  $mAP@0.5$ . EfficientDet-D7 reached  $mAP@0.5 = 0.87$  and  $mAP@0.75 = 0.81$ , while CenterNet achieved  $mAP@0.5 = 0.88$  at 35 FPS. Robustness analysis under simulated occlusion demonstrated that hybrid-trained models maintain reliable detection even under severe conditions: YOLOv11n retained  $mAP@0.5 = 0.78$  at 50% occlusion and  $mAP@0.5 = 0.65$  at 25% object visibility, while the degradation in  $mAP$  under 75% occlusion did not exceed 20% of the baseline level. The results confirm the viability of synthetic data as a standalone pretraining resource and validate the proposed hybrid pipeline for applications in autonomous driving, video surveillance, and industrial inspection.

**Keywords:** computer vision, synthetic data, neural networks, object detection, 3D modeling, dataset generation, deep learning.

**Citation:** Algashev GA, Kremushchenko PA, Lezin IA. Object detection in images using deep neural networks and synthetic data in scenarios of partial object occlusion. *Computer Optics* 2026; 50(2): 1879. DOI: 10.18287/COJ1879.

## Introduction

Over the past decades, computer vision has undergone rapid advancement thanks to the integration of deep learning techniques, which have fundamentally transformed methods for image analysis. One of the most significant and widely adopted tasks in this domain is object detection in images. Unlike classification – which merely identifies the presence of an object of a given class – object detection also requires precise localization, expressed by the coordinates of a bounding box surrounding the object. In this way, detection unifies elements of segmentation, recognition, and spatial scene analysis.

The practical importance of this task lies in its universality: object detection technologies are employed in autonomous driving, video surveillance, robotics, healthcare, industrial automation, agriculture, and other mission-critical areas [1].

Despite rapid progress, object detection continues to confront a range of challenges arising both from the nature of visual data and from limitations of the applied models. One such challenge is processing images in which objects are partially occluded by other scene elements or extend beyond the frame boundaries. Equally problematic is the high variability in how the same object may appear – changes in viewpoint, scale, texture, or color can dramatically affect performance. Furthermore, scenes cluttered with visual noise or complex backgrounds make localization and classification difficult even for clearly distinguishable objects. Under these conditions, a detection system must not only extract features but also generalize them, isolating meaningful patterns among numerous irrelevant details.

Early approaches to object detection relied on hand-crafted image features – gradient histograms, texture descriptors, and shape cues. However, these features were extremely sensitive to variations in capture conditions and did not scale well to complex, multi-class scenarios. The advent of convolutional neural networks (CNNs) marked a qualitative leap: models began to learn features automatically, building representations at multiple levels of abstraction. This gave rise to architectures such as Faster R-CNN [2], YOLO [3], SSD [4], and others, each contributing to a delicate balance between accuracy, processing speed, and generality.

It is important to note, however, that model effectiveness largely depends on the quality and diversity of the training dataset. Models trained on narrow, low-variability samples may achieve high accuracy under laboratory conditions but fail in real-world scenarios. This issue becomes especially acute when data collection and manual annotation are difficult or impossible. In particular, when working with objects encountered in industrial or non-standardized environments, traditional methods demand substantial time and financial resources to assemble a sufficiently representative training set.

Consequently, increasing attention is being paid to the use of synthetic data, which can be programmatically generated with controlled variability in object pose, orientation, background, lighting, noise level, etc. Such data not only accelerate

and simplify the preparation of training datasets but also intentionally bolster model robustness to conditions encountered in real environments. Moreover, synthetic generation enables creation of scenes that would be difficult or unsafe to reproduce physically.

Thus, the research trajectory in object detection is shifting toward hybrid approaches that combine synthetic and real data, seeking a compromise between theoretical generalization capacity and practical applicability. Concurrently, methods for modeling partially visible objects and evaluating algorithm robustness to such distortions are under active investigation. As a result, a new direction is emerging – one that assesses not only accuracy and speed but also the reliability of models operating under unstable visual conditions.

### 1. Introduction problem statement and overview of approaches

At the core of any computer-vision system tasked with object detection lies the following formal definition: one must learn a mapping

$$\phi: I \rightarrow \{b_i\}_{i=1}^K, \quad (1)$$

where  $I$  denotes the input image, and each  $b_i = (x_1^i, y_1^i, x_2^i, y_2^i, c_i, s_i)$  represents a bounding box parameterized by the coordinates of its top-left and bottom-right corners, the object class  $c_i$  and a confidence score  $s_i$ . The scores  $s_i$  encodes the posterior probability that the box indeed contains an instance of class  $c_i$ . This formulation highlights two intertwined subtasks: spatial localization of each object within the frame and its subsequent classification.

Despite the apparent simplicity of this formalism, implementing the function  $\phi$  in practice encounters multiple challenges. First, bounding boxes must tightly enclose objects even when they are partially invisible or subject to distortion, necessitating meticulous tuning of localization algorithms. Second, real-world images seldom exhibit homogeneous backgrounds: cluttered scenes can contain textures and colors that closely resemble target objects, forcing detection systems to filter out irrelevant details. Finally, objects span a broad range of visual appearances and scales – from large, prominent items to small, fine-grained details – which often requires specialized multi-scale analysis techniques.

Historically, the earliest object-detection methods relied on manually engineered features such as gradient histograms, corner detectors, and texture descriptors. Approaches like HOG (Histogram of Oriented Gradients) [5] demonstrated notable success in pedestrian detection tasks but proved fragile under variations in illumination, viewpoint, and background. Likewise, interest-point algorithms such as SIFT [6] and SURF [7] enabled matching distinctive image patches across frames, yet their high computational cost and sensitivity to noise limited their real-time applicability and scalability to large datasets.

The transition to machine learning introduced automatic feature selection: support-vector machines (SVMs) [8] and random forests [9] were trained on precomputed descriptors, offering greater flexibility and noise resistance. Nonetheless, these classifiers still depended on careful feature design and struggled to capture complex, high-level dependencies among pixels.

The real revolution arrived with deep neural networks capable of learning hierarchical representations directly from data. Convolutional architectures gave rise to R-CNN, where a region-proposal stage was followed by classical CNN-based classification [10]. Although R-CNN achieved significant accuracy gains, its need to process each candidate region separately incurred substantial computational overhead.

Faster R-CNN overcame this limitation by integrating the Region Proposal Network into a single end-to-end architecture, approaching real-time performance without sacrificing precision [2]. Concurrently, single-stage detectors such as YOLO reinterpreted detection as a regression problem, directly predicting box coordinates and class probabilities in one pass through the network [3]. This simplification enabled very fast inference, albeit with somewhat reduced accuracy on small or partially occluded objects.

SSD (Single Shot MultiBox Detector) struck a balance by combining one-pass processing with feature-pyramid analysis, enhancing detection consistency across object scales [4]. Its successor, EfficientDet [11], further optimized this concept via the BiFPN (Bidirectional Feature Pyramid Network), effectively trading off between computational cost and accuracy –thus enabling deployment on resource-constrained hardware.

More recently, transformer-based detectors (e.g., DETR) have emerged, leveraging self-attention to model global relationships across image regions without explicit region proposals [12]. While these methods achieve state-of-the-art performance on complex scenes, they demand substantial computational resources and large volumes of training data.

In summary, the evolution of object-detection techniques has progressed from rigid, hand-crafted features to fully learnable, adaptive systems capable of operating under diverse imaging conditions. In scenarios characterized by partial visibility and cluttered backgrounds, the value of hybrid strategies –combining programmatically generated synthetic data with fine-tuning on real imagery –becomes increasingly clear, as they provide an optimal trade-off among accuracy, speed, and robustness.

### 2. Formation of synthetic and real data

In the present study, a two-stage approach was employed to prepare training datasets. In the first stage, a fully synthetic dataset with controlled variability was created; in the second stage, a real-image dataset was assembled to assess the models' generalization performance. This methodology not only encompassed a wide range of conditions but also ensured robust adaptation of the networks to real-world capture artifacts.

The synthetic dataset was generated using the VTK library [13], which offers photorealistic rendering of 3D models and automatic computation of bounding-box coordinates. For each of four types of LEGO blocks, detailed 3D models were constructed accounting for geometry, surface reflectance, and material properties. During generation, object positions were sampled uniformly within the virtual scene, and orientations were drawn from random Euler angles covering the full spectrum of spatial viewpoints. Color parameters were varied within  $\pm 5\%$  of the reference hue to simulate both matte and subtly glossy surfaces. Backgrounds were composed from HDR images selected to produce complex, high-contrast compositions, while lighting setups – ranging from one to three sources – were arranged to provide a combination of key, fill, and backlighting. Additional visual complexity was introduced by inserting random occluders that partially covered the blocks and by applying motion-blur or Gaussian-blur operators to emulate focus imperfections.

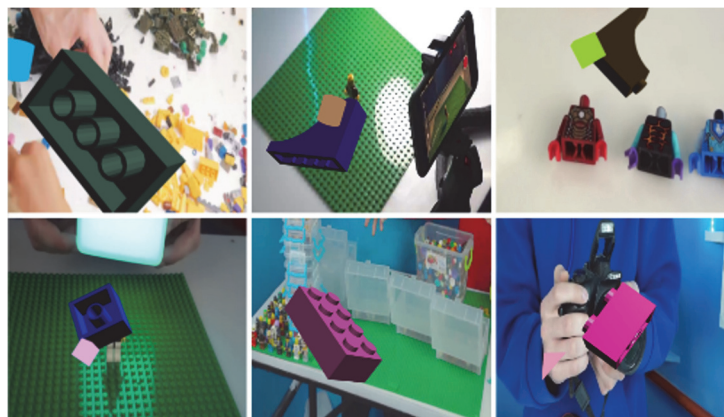


Fig. 1. Example of synthetic training images

The final synthetic collection comprised 18,000 color images at a resolution of  $640 \times 640$  pixels. Each image was accompanied by metadata containing exact bounding-box coordinates and class identifiers. The fully automated generation pipeline yielded a dataset that was homogeneous in quality yet diverse in content, encompassing both ideal and edge-case detection scenarios.

The real-image dataset consisted of 2,000 photographs (500 images per class) captured under varied conditions. All images were resized to  $640 \times 640$  pixels without altering color profiles, thereby preserving natural sensor noise and device-specific characteristics. Bounding boxes were annotated manually.

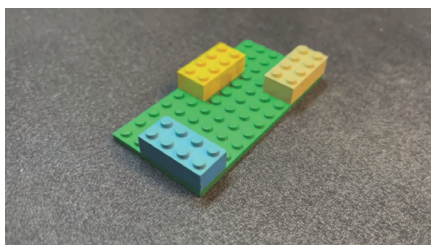


Fig. 2. Example of real training data

To enable seamless integration of synthetic and real images during training, a preprocessing utility was developed to perform the following tasks: color-channel normalization to a common range, application of random augmentations (horizontal flips and minor geometric distortions), and generation of annotations in the COCO format [14]. As a result, synthetic and real data could be interleaved within training batches, allowing the models to simultaneously learn the clean statistics of VTK renders and the chaotic artifacts of real-world captures. This hybrid strategy facilitated a smooth transition from strictly controlled synthetic conditions to the unpredictability of real environments, markedly enhancing detector robustness in final evaluations.

### 3. Training methodology and parameter unification

In this research, a unified training protocol was applied across all examined architectures to obtain comparable results and minimize the influence of external factors. The implementation was based on the PyTorch framework [15], offering flexibility in configuring optimizers, regularization methods, and metric logging. Each model was initialized with pre-trained ImageNet weights [16], except for transformer-based detectors (DETR), which used the original initializations recommended by their authors.

Input images were normalized to a common standard: each color channel was scaled using the mean and standard deviation computed over the combined synthetic and real datasets. This normalization mitigated pixel-statistic discrepancies between the “idealized” VTK renders and natural captures. During training, random augmentations –horizontal flips and minor geometric shifts –were applied to enhance robustness to compositional variability.

A key element of unification was the use of the AdamW optimizer [14] with a fixed learning rate of  $3 \times 10^{-4}$  for all models. This choice balanced the recommendations for transformer and convolutional architectures: AdamW incorporates built-in L2 regularization [17], reducing the risk of overfitting on synthetic data, and demonstrates stable convergence even with a relatively large gradient step. Coupled with a small batch size of 32 images, this configuration kept GPU memory usage within bounds while ensuring acceptable gradient estimation and weight-update speed.

Training was limited to 50 epochs over the entire dataset, allowing models to exhibit main convergence trends without excessive computational expense. For some architectures –such as DETR–this was insufficient to reach a performance plateau, but the uniform training duration provided the fairest basis for comparison. To guard against overfitting, the ReduceLROnPlateau scheduler [10] was employed: if mAP failed to improve for more than two epochs, the learning rate was halved.

Most convolutional detectors used the standard ReLU activation, whereas YOLO and EfficientDet employed the SiLU (Swish) function to facilitate smoother gradient flow. Despite differences in activations, a unified weight-initialization scheme ensured that comparisons focused on architectural features rather than minor parametric variations.

To combat noise and artifacts in synthetic images, two levels of regularization were applied: an L2 weight-decay term ( $\lambda=1 \times 10^{-4}$ ) integrated into the optimizer, and Dropout ( $p=0.2$ ) in the classification heads. This approach suppressed excessive sensitivity to noisy features while preserving the capacity to learn complex representations.

The entire training procedure was logged via TensorBoard, capturing loss curves, accuracy, and auxiliary metrics (Precision, Recall, F1 Score) [18]. Checkpoints were saved whenever mAP@0.5 improved, ensuring selection of the best model for subsequent testing. Thanks to strict hyperparameter unification and a single data-preparation pipeline, the results allow for objective comparison of different architectures under synthetic, real, and combined training regimes.

For model evaluation, an additional set of 500 real-world images –uniformly balanced across all classes and excluded from training –was prepared.

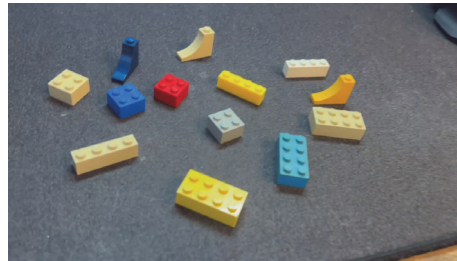


Fig. 3. Example of a test image

#### 4. Experimental study

During the experiments, a comprehensive evaluation was conducted on six deep neural network architectures –Faster R-CNN (ResNet-50 FPN), SSD (MobileNet V3), YOLOv11n, EfficientDet D7, DETR DC5, and CenterNet++ –across three training scenarios: exclusively on synthetic data, exclusively on real images, and on a combination of both. This staged evaluation scheme enabled the identification of each model’s strengths and weaknesses in conditions closely resembling real-world object detection tasks.

The initial series of experiments, based solely on the synthetic dataset (18,000 images of LEGO blocks) and summarized in Table 1, showed that EfficientDet D7 and YOLOv11n achieved the highest accuracy. At the mAP@0.5 threshold, EfficientDet D7 reached 0.75, while YOLOv11n achieved 0.79. Both models scored 0.74 at mAP@0.75, indicating their capability to effectively handle complex cases involving partial occlusions and varied viewpoints. However, YOLOv11n demonstrated superior inference speed, making it a more favorable choice for real-time applications.

Although Faster R-CNN and DETR DC5 achieved solid mAP@0.5 scores of 0.72 and 0.68, respectively, they lagged significantly in processing speed –14 FPS and 9 FPS –which limits their suitability for time-critical deployments. CenterNet++, by contrast, offered a balanced performance with an mAP@0.5 of 0.74 and an inference speed of 35 FPS, positioning it as a viable compromise between localization accuracy and computational efficiency.

Tab. 1. Comparison of model performance trained on synthetic data

Model	mAP@0.5	mAP@0.75	Precision	Recall	F1-Score	FPS
Faster R-CNN (ResNet-50-FPN)	0.72	0.63	0.68	0.75	0.71	14
SSD (MobileNet V3)	0.65	0.55	0.73	0.62	0.67	45
YOLOv11n	0.79	0.74	0.76	0.76	0.76	82
EfficientDet-D7	0.75	0.74	0.77	0.70	0.73	28
DETR-DC5	0.68	0.60	0.65	0.60	0.62	9
CenterNet++	0.74	0.72	0.76	0.72	0.74	35

When shifting to training exclusively on real images (2,000 samples, with 500 per class under varying lighting conditions and background noise), all models exhibited a noticeable sensitivity to the limited size of the training dataset. The testing results are presented in Table 2. YOLOv11n achieved an mAP@0.5 score of 0.78, while EfficientDet D7 consistently

maintained a score of 0.75. However, under the stricter  $mAP@0.75$  threshold, the performance gap between the detectors narrowed, reflecting an overall decline in localization quality due to the insufficient number of training examples.

In this experiment, the leading models in terms of  $mAP$  remained EfficientDet D7 and YOLOv11n. SSD (MobileNet V3) delivered a respectable  $mAP@0.5$  of 0.65 while achieving 45 FPS, demonstrating high efficiency in resource-constrained environments.

Tab. 2. Comparison of model performance trained on real data

Model	$mAP@0.5$	$mAP@0.75$	Precision	Recall	F1-Score	FPS
Faster R-CNN (ResNet-50-FPN)	0.70	0.60	0.68	0.65	0.66	14
SSD (MobileNet V3)	0.65	0.55	0.67	0.62	0.64	45
YOLO11n	0.78	0.70	0.75	0.74	0.74	82
EfficientDet-D7	0.75	0.67	0.73	0.70	0.71	28
DETR-DC5	0.68	0.60	0.66	0.63	0.64	9
CenterNet++	0.72	0.64	0.70	0.68	0.69	35

A key stage in evaluating generalization capability involved training on a mixed dataset composed of 90 % synthetic images and 10 % real-world frames. Models trained under these conditions demonstrated a significant increase in  $mAP$ : YOLOv11n achieved 0.91 ( $mAP@0.5$ ), while EfficientDet-D7 reached 0.87, confirming the effectiveness of the hybrid training approach.

The improvement was particularly notable at the  $mAP@0.75$  threshold: YOLOv11n rose to 0.86, nearly 8 % higher than the results from training on synthetic data alone. An important observation was the performance of the transformer-based DETR DC5. Initially showing weak convergence over 50 epochs, it reached  $mAP@0.5 = 0.80$  in the hybrid setup, indicating substantial progress due to the inclusion of real examples.

Tab. 3. Comparison of model performance trained on synthetic and real data

Model	$mAP@0.5$	$mAP@0.75$	Precision	Recall	F1-Score	FPS
Faster R-CNN (ResNet-50-FPN)	0.85	0.78	0.80	0.84	0.82	14
SSD (MobileNet V3)	0.79	0.72	0.82	0.76	0.79	45
YOLO11n	0.91	0.86	0.89	0.90	0.89	82
EfficientDet-D7	0.87	0.81	0.85	0.86	0.85	28
DETR-DC5	0.80	0.73	0.78	0.79	0.78	9
CenterNet++	0.88	0.82	0.86	0.87	0.86	35

To thoroughly assess the robustness of the networks to real-world artifacts, an occlusion analysis was conducted using a test set of 500 real images, augmented to simulate object occlusion levels of 25 %, 50 %, and 75 % of the object's area. The results showed that models trained using the hybrid approach retained a high detection capability even under severe occlusion.

At 50 % visibility, YOLOv11n maintained an  $mAP@0.5$  of 0.78, whereas Faster R-CNN dropped to 0.68 and DETR DC5 to 0.64. In the scenario of minimal visibility (25%), YOLOv11n still preserved an  $mAP@0.5$  of 0.65. These findings highlight that synthetic data enables models to "see" objects under extreme conditions, thereby extending their generalization capacity to real-world imagery.

Tab. 4. Results of trained deep learning models on synthetic and real data in case of data occlusion

Model	$mAP@0.5$ (75%)	$mAP@0.75$ (75%)	$mAP@0.5$ (50%)	$mAP@0.75$ (50%)	$mAP@0.5$ (25%)	$mAP@0.75$ (25%)	FPS
Faster R-CNN	0.74	0.65	0.68	0.60	0.55	0.50	14
SSD	0.70	0.62	0.63	0.55	0.50	0.45	45
YOLO11n	0.85	0.77	0.78	0.70	0.65	0.58	82
EfficientDet-D7	0.80	0.72	0.72	0.65	0.60	0.53	28
DETR-DC5	0.71	0.64	0.64	0.57	0.52	0.47	9
CenterNet++	0.79	0.71	0.71	0.63	0.59	0.52	35

In addition to purely quantitative metrics, a qualitative analysis of detection errors was carried out in this study. For each architecture, the most frequent types of false positives and missed detections were identified. For instance, SSD (MobileNet V3) exhibited the lowest number of false positives on synthetic data but often failed to detect small details in real-world images. CenterNet++ demonstrated stable performance under partial occlusion but underperformed when dealing with uncommon viewpoints. DETR DC5, being the most sensitive to training parameters, required additional epochs to fully leverage the potential of transformer-based attention mechanisms.

A comprehensive analysis of the experimental data leads to the conclusion that, for detection tasks under limited visibility and mixed lighting conditions, the most optimal solution is the single-shot detector YOLOv11n trained on a combination of synthetic and real data. EfficientDet D7 remains a strong contender when a balance between accuracy and resource constraints is required, while CenterNet++ offers solid resilience to partial occlusion. Transformer-based DETR DC5 demands longer training durations, but has the potential to become the foundation for advanced systems capable of capturing global relationships within images.

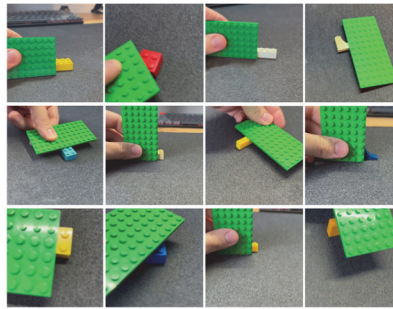


Fig. 4. Example of real test images with partially visible objects

### Conclusion

The conducted study demonstrated the high effectiveness of a hybrid training approach for modern neural object detectors, combining synthetic and real-world data. It was shown that pretraining on a scalable, controllable synthetic dataset –generated using the VTK library –enables models to learn fundamental geometric and textural features of objects. Subsequent adaptation on a smaller, but artifact-rich real dataset helps correct error distributions and improves localization under conditions close to those found in industrial and field environments. This synergy was especially evident in the case of the single-shot YOLOv11n detector, which exhibited an increase in mAP@0.5 from 0.79 (pure synthetic) and 0.78 (pure real) to 0.91 under combined training, confirming the importance of including even a small amount of real examples.

A detailed analysis of the results revealed that pyramid-based architectures (EfficientDet D7, SSD with MobileNet V3) are particularly robust to variations in object scale, maintaining consistent detection quality across changes in view-point and size. Meanwhile, the transformer-based detector DETR DC5, despite its relatively modest performance under a limited number of training epochs, showed noticeable improvement with hybrid training. This highlights the potential of global attention mechanisms for handling complex and cluttered scenes. It suggests that further increasing the number of training iterations and expanding the real dataset could make transformer architectures more competitive for object detection tasks.

Special attention was devoted to testing model robustness under partial occlusion, simulated by varying degrees of object coverage. The results demonstrated that hybrid-trained models significantly outperformed those trained on a single data type: they were able to detect objects even when more than half of their visible area was obscured –an essential capability for applications in robotics, surveillance systems, and autonomous driving. The decrease in mAP under 75% occlusion did not exceed 20% of the baseline level, which is significantly better than the performance of models trained solely on synthetic or purely real data.

From a practical standpoint, the unified training methodology –featuring consistent normalization, optimization, and regularization protocols –enabled objective comparison of architectures under equal conditions. It revealed that the optimal balance between accuracy and speed is achieved in single-shot and BiFPN-based detectors. Furthermore, the automated preprocessing and augmentation system that seamlessly combines synthetic and real frames proved to be effective for industrial-scale training pipelines and model transfer across different hardware platforms.

From a scientific perspective, the study confirms the viability of using synthetic data as a standalone resource for pretraining, especially when real annotation is costly or difficult. At the same time, incorporating “live” examples is necessary to eliminate domain shifts and ensure reliable performance in real-world conditions. The observed patterns regarding how models learn features and adapt to imaging artifacts can serve as a foundation for further research into automatic scenario synthesis tailored to the specifics of applied tasks.

Future directions include expanding the variety of synthetic scenes by generating new types of background distortions, introducing fuzzy labeling techniques for more realistic partial occlusion augmentation, and adapting detection thresholds according to application-specific requirements. Particularly promising are investigations into dynamic hyperparameter tuning based on mixed data distributions and the application of semi-supervised learning for more efficient use of unlabeled real images. In the long term, these methods may enable even deeper integration of synthetic idealization with real-world realism, ensuring reliable and high-speed detection performance across diverse application environments.

### Acknowledgments

The research was carried out within the state assignment theme FSSS-2023-0006.

### References

- [1] Kazanskiy NL, Popov SB. The distributed vision system of the registration of the railway train. *Computer Optics* 2012; 36(3): 419-428.
- [2] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell* 2017; 39(6): 1137-1149. DOI: 10.1109/TPAMI.2016.2577031.
- [3] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. 2016 IEEE Conf on Computer Vision and Pattern Recognition (CVPR) 2016: 779-788. DOI: 10.1109/CVPR.2016.91.

- [4] Howard A, Sandler M, Chu G, et al. Searching for MobileNetV3. 2019 IEEE/CVF Int Conf on Computer Vision (ICCV) 2019: 1314-1324. DOI: 10.1109/ICCV.2019.00140.
- [5] Tan M, Pang R, Le QV. EfficientDet: Scalable and efficient object detection. 2020 IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR) 2020: 10778-10787. DOI: 10.1109/CVPR42600.2020.01079.
- [6] Kwon G, Prabhushankar M, Temel D, AlRegib G. Backpropagated gradient representations for anomaly detection. In: Vedaldi A, Bischof H, Brox T, Frahm JM, eds. Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI. Cham: Springer; 2020: 13-29. DOI: 10.1007/978-3-030-58589-1\_13.
- [7] Duan K, Bai S, Xie L, Qi H, Huang Q, Tian Q. CenterNet++ for object detection. arXiv Preprint. 2022. Source: <<https://arxiv.org/abs/2204.08394>>. DOI: 10.48550/arXiv.2204.08394.
- [8] Schroeder W, Martin K, Lorensen B. The visualization toolkit: An object-oriented approach to 3D graphics. 4th ed. Clifton Park, NY: Kitware Inc; 2006.
- [9] Bochkovskiy A, Wang C-Y, Liao H-YM. YOLOv4: Optimal speed and accuracy of object detection. arXiv Preprint. 2020. Source: <<https://arxiv.org/abs/2004.10934>>. DOI: 10.48550/arXiv.2004.10934.
- [10] Kim N, Park J, Choi Y, Oh S. Viewpoint estimation for visual target navigation by leveraging keypoint detection. 2020 20th Int Conf on Control, Automation and Systems (ICCAS) 2020: 1162-1165. DOI: 10.23919/ICCAS50221.2020.9268215.
- [11] Wu X, Yang S, Cai Z, Song R, Fan S. Measurement of fish motion parameters based on DeepLabCut. 2024 IEEE 19th Conf on Industrial Electronics and Applications (ICIEA) 2024: 1-5. DOI: 10.1109/ICIEA61579.2024.10664666.
- [12] Maji D, Nagori S, Mathew M, Poddar D. YOLO-Pose: Enhancing YOLO for multi-person pose estimation using object keypoint similarity loss. 2022 IEEE/CVF Conf on Computer Vision and Pattern Recognition Workshops (CVPRW) 2022: 2636-2645. DOI: 10.1109/CVPRW56347.2022.00297.
- [13] Yanyan Z, Xiangjin R. A study on 3D visualization system for GoCAD objects based on VTK and QT. 2016 Int Conf on Robots and Intelligent System (ICRIS) 2016: 47-50. DOI: 10.1109/ICRIS.2016.10.
- [14] Obeidavi S, Gandomkar M, Hirtz G. In-pose estimation of covered and uncovered human body from thermal camera images using multi-scale stacked hourglass (MSSHg) network. 2022 16th Int Conf on Signal-Image Technology and Internet-Based Systems (SITIS) 2022: 84-90. DOI: 10.1109/SITIS57111.2022.00021.
- [15] Ichikawa Y, Shioda A, Kawamura K, Chu TV, Motomura M. An accurate FPGA-based ORB feature extractor for SLAM with row-wise keypoint selection. 2024 IEEE Int Conf on Consumer Electronics (ICCE) 2024: 1-2. DOI: 10.1109/ICCE59016.2024.10444305.
- [16] Păvăloi I, Ignat A, Lazăr L-C, Niță C-D. Palmprint recognition with fixed number of SURF keypoints. 2021 Int Conf on e-Health and Bioengineering (EHB) 2021: 1-4. DOI: 10.1109/EHB52898.2021.9657595.
- [17] Ludwig K, Harzig P, Lienhart R. Detecting arbitrary intermediate keypoints for human pose estimation with vision transformers. 2022 IEEE/CVF Winter Conf on Applications of Computer Vision Workshops (WACVW) 2022: 663-671. DOI: 10.1109/WACVW54805.2022.00073.
- [18] Lv X, Zhang K, Li J, et al. Human gait analysis method based on sample entropy fusion AlphaPose algorithm. 2021 33rd Chinese Control and Decision Conference (CCDC) 2021: 1543-1547. DOI: 10.1109/CCDC52312.2021.9602427.

---

#### *Authors' information*

**Algashev Gennady Andreevich**, (b. 1996), graduate of the master's degree program of Informatics faculty at Samara National Research University, assistant of the Department of Information Systems and Technologies of Samara National Research University. Research interests: neural networks, digital image processing, augmented and virtual reality, computer vision, data mining. E-mail: [algashev.ga@ssau.ru](mailto:algashev.ga@ssau.ru)

**Kremushchenko Polina Alexandrovna** (b. 2004), student of the Institute of Informatics and Cybernetics of the Samara National Research University. Research interests: neural networks, digital image processing, computer vision. E-mail: [polinakremuschenko@gmail.com](mailto:polinakremuschenko@gmail.com)

**Lezin Ilya Alexandrovich**, Candidate of Technical Sciences, Associate Professor in the Department of Information Systems and Technologies of Samara National Research University. His research interests include data mining, pattern recognition, and artificial neural networks. E-mail: [lezin.ia@ssau.ru](mailto:lezin.ia@ssau.ru)

---

*Received February 15, 2025. The final version – March 25, 2025.*

---