Estimating Motion and Object Detection Using Convolutional Neural Networks

Kunal Sharma
Department of Computer Science
Drexel University
kos26drexel.edu

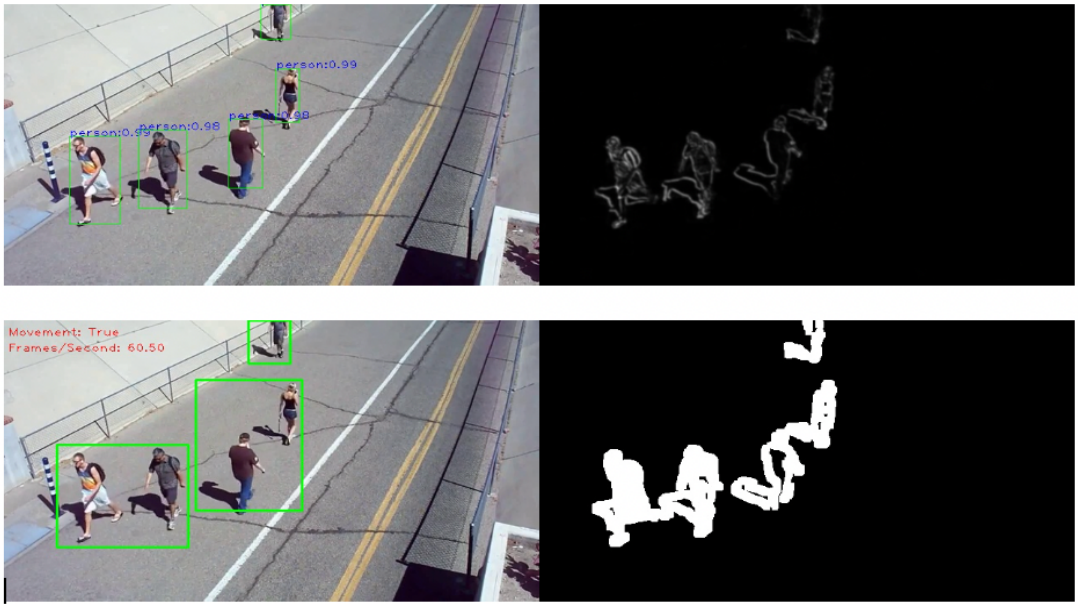


Figure 1: Our system detects objects in the frames and assigns them classes with high confidence and also estimates motion in a sequence of images. The output of motion detection are three different frames i.e. blurred frame, colored frame with green contours and diluted frame.

***Abstract*---Estimating real-time motion and object detection in non-stationary scenes can be a challenging task in image sequences due to dynamic background, changing fore-ground appearance and limited computational resources. These challenges degrade the performance of the existing methods in practical applications. In this paper, an optical flow-based framework is utilized to address motion detection and a Convolutional neural network based model YOLO version three is used to address object detection problems. By applying a novel strategy to utilize optical flow and object detection we enable our method to be free of model constructing, training or updating and can be performed efficiently. Besides, adaptive intervals and adaptive thresholds are designed to heighten the system's adaptation to different situations. The unified architecture for object detection that we have used is fast and works well with videos at up to 50 frames per second. The experimental results show that the method we used adapts well to different situations and outperforms some of the state-of-the-art methods.**

Inder Terms—Estimation Motion, Object detection

1. INTRODUCTION

In this paper, we study the real-time motion and object detection which is useful in many practical applications like increasing the sensory ability in social distance detector, autopilot, capturing significant cues for event analysis or providing intelligence in monitoring. The primary goal of the project is to detect motion and classify objects in the scene as precisely as possible. A classical way to compute the scene or object motion from a sequence of images is by minimizing the sum of squared errors between two images, the template T and the image warped back onto the coordinate frame of the template.

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks like driving with little conscious thought. Fast, accurate algorithms for object detection would allow computers to drive cars without specialized sensors, enable assistive devices to convey real-time scene information to human users, and unlock the potential for general purpose,

responsive robotic systems. Many detection systems repurpose classifiers to perform detection. To detect an object, these systems take a classifier for that object and evaluate it at various locations and scales in a test image. Systems like deformable parts models (DPM) use a sliding window approach where the classifier is run at evenly spaced locations over the entire image and this was an expensive operation, the aspect ratio of the object was usually assumed to be fixed. Redmon and Farhadi [5] states recent approaches like R-CNN use region proposal methods to first generate potential bounding boxes in an image and then run a classifier on these proposed boxes. After classification, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and rescore the boxes based on other objects in the scene [13]. These complex pipelines are slow and hard to optimize because each individual component must be trained separately.

In our project for detecting motion we compute the difference between two frames in sequence of images, apply a threshold for the pixels that have changed from others and then count all the black pixels. We have found this technique to be robust and it works well with videos at up to 50 frames per second. We have used YOLOv3 which is a one of the latest variant of a popular object detection algorithm, YOLO stands for You Only Look Once and the published model recognizes 80 different objects in images and videos and most importantly it is super-fast and nearly as accurate as Single Shot Multi Box. The algorithm forwards the whole image only once through the network. SSD is another object detection algorithm that forwards the image once though a deep learning network, but [5]YOLOv3 is much faster than SSD while achieving very comparable accuracy. YOLOv3 gives faster than realtime results on a M40, TitanX or 1080 Ti GPUs. First, it divides the image into a 13×13 grid of cells. The size of these 169 cells vary depending on the size of the input. For a 416×416 input size that we used in our experiments, the cell size was 32×32. Each cell is then responsible for predicting a number of boxes in the image. For each bounding box, the network also predicts the confidence that the bounding box actually encloses an object, and the probability of the enclosed object being a particular class. Most of these bounding boxes are eliminated because their confidence is low or because they are enclosing the same object as another bounding box with very high confidence score. This technique is called non-maximum suppression.

2. RELATED WORK

2.1 R-CNN

R-CNN and its variants use region proposals instead of sliding windows to find objects in Images. Selective Search generates potential bounding boxes, a convolutional network extracts features, an SVM scores the boxes, a linear model adjusts the bounding boxes, and non-max suppression eliminates duplicate detections. Each stage of this complex pipeline must be precisely tuned independently, and the resulting system is very slow, taking more than 40 seconds per image at test time.

2.2 Optical Flow Based

Kurnianggoro, Shahbaz and Jo [2] modeled the background using zero optical flow vectors. After using a homography matrix to align the previous frame, dense optical flow was estimated between the alignment result and the current frame. Finally, a simple optical-flow magnitude threshold was used to judge out the foreground points. As the homography matrixes are only used for aligning, the background model and the judge mechanism constructed by this method are too simple to deal with intricate unconstrained scenes.

2.3 CNN based motion detection

Tokmakov P, Alahari K, Schmid [3] and Siam, Mahgoub, Zahran [4] learned motion detection and score highly on public datasets. Their methods require training process and rely seriously on the completeness of the training sample. Besides, large amount of computation keeps them away from being real-time.

Dosovitskiy, Fischer, Ilg, H¨ausser, Hazırbas, Golkov and Brox [1] proposed and compare a generic architecture FlowNetSimple which stack both input images together and feed them through a rather generic network, allowing the network to decide itself how to process the image pair to extract the motion information and FlowNetCorr which creates two separate, yet identical processing streams for the two images and to combine them at a later stage. With such architecture the network is constrained to first produce meaningful representations of the two images separately and then combine them on a higher level.

CONTRIBUTION / METHOD

The framework of our estimating motion and object detection method in image sequences are mainly two processes. In the first process finds the contours of the moving objects and calculate the area of all of them. Then the average of the surface changing is compared with the total surface of the image and the alarm is triggered if it exceeds the given threshold. The assumption made for this process was the the first frame of the video file will contain no motion and just background. Due to tiny variations in the digital camera sensors, no two frames will be 100% the same therefore some pixels will most certainly have different intensity values. A gaissain smoothing was applied to average pixel intensities across the image. This helps smooth out high frequency noise that could throw our motion detection algorithm off. The

output of this algorithm is three different frames i.e. colored frame with a green contour on the objects that are moving, blurred framed which is a difference between the original frame and the current frame and a diluted frame which is gained by thresholding the blurred frame to segment the foreground from the background.

In the second process, for object detection in the sequence of images the YOLOv3 detector divides the image into a 13×13 grid of cells. The size of these 169 cells vary depending on the size of the input. For a 416×416 input size that we used in our experiments, the cell size was 32×32. Each cell is then responsible for predicting a number of boxes in the image. For each bounding box, the network also predicts the confidence that the bounding box actually encloses an object, and the probability of the enclosed object being a particular class. Most of these bounding boxes are eliminated because their confidence is low or because they are enclosing the same object as another bounding box with very high confidence score. This technique is called non-maximum suppression.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 2: Architecture of a YOLOv3, Darknet-53

RESULTS

The proposed method is implemented using python on a MacBook with an Intel i5-7400 CPU, 32 GB RAM. We measure the computation time with video mostly at resolution of 640 ∗ 360 to evaluate the efficiency of the proposed methods. The computation time of estimating take up to 50 frames per second and computation time of object detection take upto 20 frames per second which is better than some of the known complex algorithms.

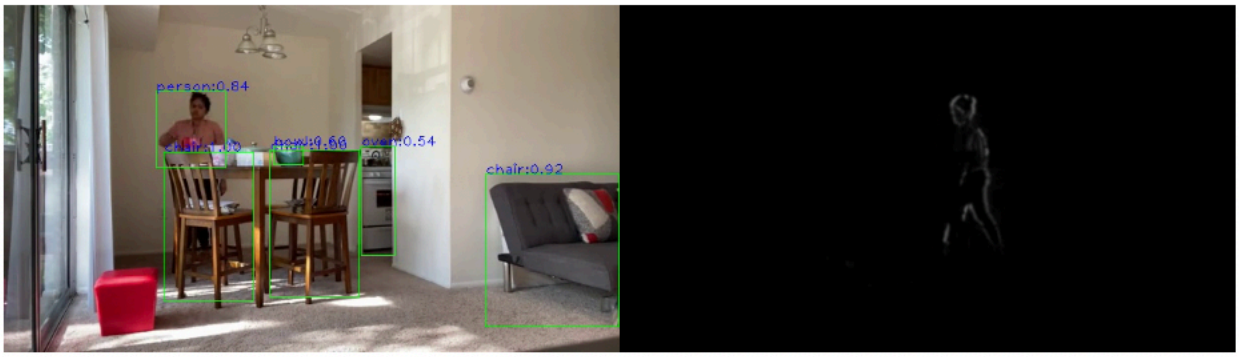
YOLOv3 object detector is a fast, accurate and making it ideal for computer vision applications. The detector is preferable for videos because of its robustness including the time to fetch from the camera and display the detections. The resulting system is interactive and engaging.

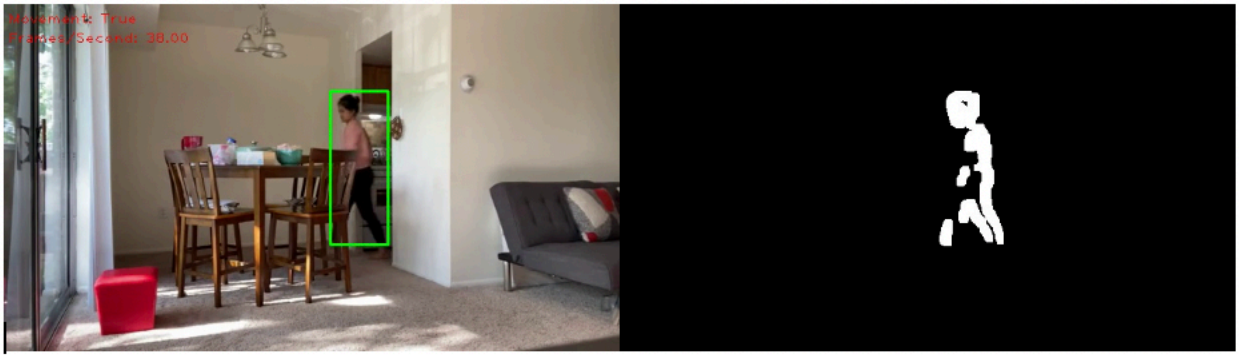
	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Figure 3: Comparisons of different object detectors.

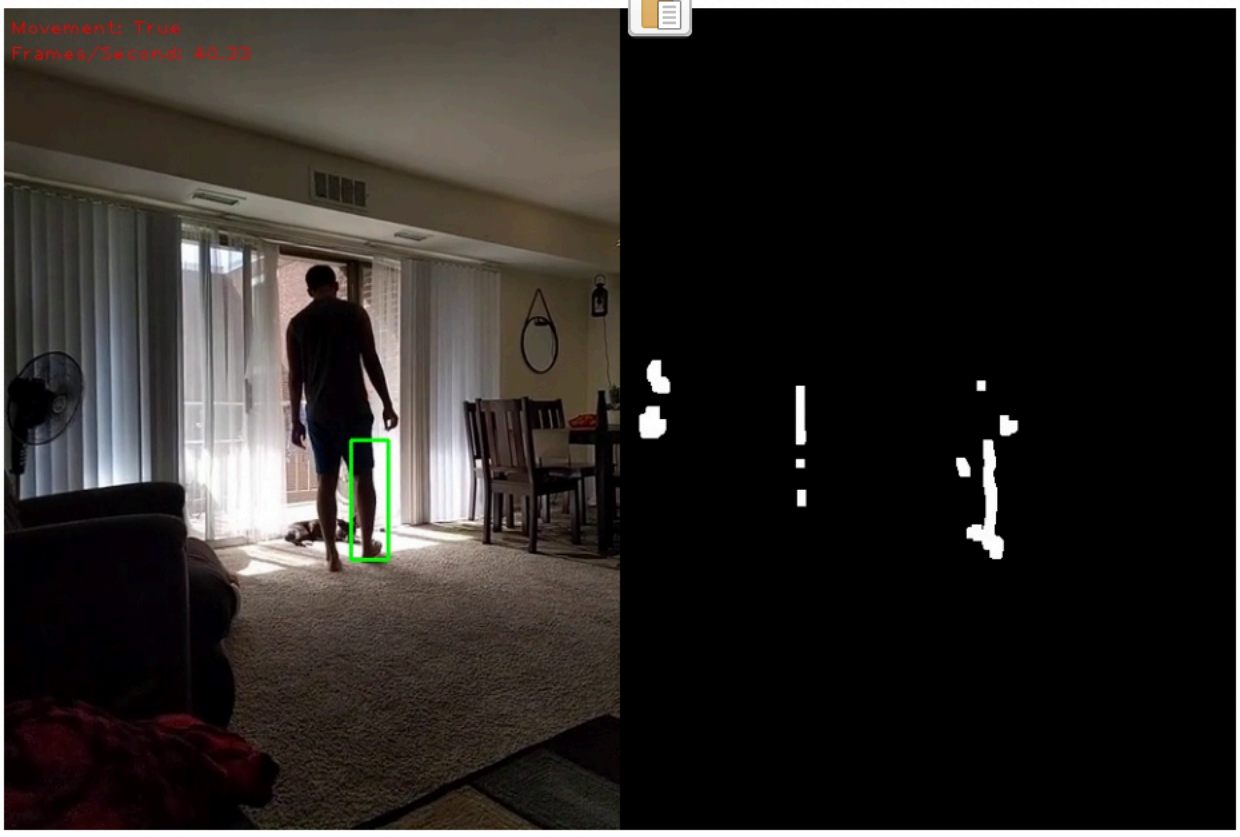
Bibliography:

1. A. Dosovitskiy, P. Fischer, E. Ilg, P. H¨ausser, C. Hazırbas,V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, 2015.
2. Kurnianggoro L, Shahbaz A, Jo K H, Dense optical flow in stabilized scenes for moving object detection from a moving camera, IEEE International Conference on Control, Automation and Systems (ICCAS), 704-708, 2016.
3. Tokmakov P, Alahari K, Schmid C, Learning motion patterns in videos, IEEE Conference on Computer Vision and Pattern Recognition, 531- 539, 2017.
4. Siam M, Mahgoub H, Zahran M, et al, MODNet: Moving Object Detection Network with Motion and Appearance for Autonomous Driving. arXiv preprint arXiv:1709.04821, 2017.
5. Redmon, Joseph & Farhadi, Ali. (2018). YOLOv3: An Incremental Improvement.


person:0.84
chair:0.92


Movement: True
Frames/Second: 38.00


Movement: True
Frames/Second: 40.00


Movement: True
Frames/Second: 40.33