Object Motion Forecasting with Object Tracking and Trajectory Prediction Techniques

Utku Acar
CS523.V: Computer Vision
Ozyegin University
Vestel Electronics
Manisa, Turkey
utku.acar@ozu.edu.tr

June 1, 2023

Abstract

This report presents a study on predicting object motion using object tracking and trajectory prediction techniques. The goal is to develop a system that accurately forecasts the future motion of objects in computer vision applications. The report discusses the techniques and models used for object detection, feature extraction, object tracking, and motion prediction, along with their implementation details. The results demonstrate the effectiveness of the developed system in accurately predicting object motion, as evidenced by metrics such as average IoU and average FPS with using different pre-trained models. The system shows promise for various computer vision scenarios and can contribute to applications such as autonomous driving, video surveillance, and human-computer interaction. The report also highlights related works in the field of object motion prediction and compares them to the proposed system.

Keywords: Object motion prediction, object tracking, trajectory prediction, YOLOv4, Histogram of Oriented Gradients (HOG), Haar cascade method.

1 Introduction

Object motion prediction is a crucial task in computer vision applications such as autonomous driving, video surveillance, and human-computer interaction. By accurately forecasting the future motion of objects, these systems can make informed decisions and take appropriate actions. This report presents a study on predicting object motion using object tracking and trajectory prediction techniques. The developed system aims to enhance the understanding and prediction of object motion in various computer vision scenarios.

2 Related Work

In the field of object motion prediction, several approaches have been proposed. Here, we discuss three related works:

2.1 On Human Motion Prediction Using Recurrent Neural Networks

Previous studies on human motion modeling using deep RNNs have overlooked the task of short-term motion prediction. Martinez et al. (2017) [1] have demonstrated that a simple zero-velocity prediction baseline outperforms the state-of-the-art methods. To address this gap, they propose a sequence-to-sequence architecture with residual connections that surpasses previous approaches when trained on sample-based loss. Their architecture is straightforward and scalable, making it suitable for training on large-scale human motion datasets, which they found crucial for capturing short-term dynamics. While providing high-level supervision improves performance, their unsupervised baseline remains highly competitive. This departure from previous work typically focused on small action-specific datasets, is promising. Future research can explore leveraging even larger motion capture datasets in an unsupervised manner.

2.2 Trajectory Prediction Based on Social Interaction

Alahi et al. (2016) [2] proposed the "Social" LSTM, an LSTM-based model that predicts human trajectories by jointly reasoning across multiple individuals in a scene. Each trajectory is modeled by a separate LSTM, and information is shared among them through a novel Social pooling layer. The proposed method outperforms existing approaches on publicly available datasets and successfully captures non-linear behaviors resulting from social interactions. Future work aims to extend the model to multi-class settings, where different objects share the same space, and to incorporate human-space interactions by including local static-scene images as additional inputs to the LSTM.

2.3 PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning

PredRNN, introduced by Wang et al. (2021) [3], is a recurrent network designed to capture spatial deformations and dynamics across frames. It uses a Spatiotemporal LSTM unit with hierarchical memory interactions to effectively represent different information levels. The unique memory flow, with separate pathways for horizontal and zigzag states, enables PredRNN to learn distributed representations for spatiotemporal variations. The authors propose to reverse scheduled sampling as a curriculum learning strategy to improve encoding and learn temporal dynamics from longer context frames. Experimental results demonstrate PredRNN's state-of-the-art performance on various spa-

tiotemporal datasets, including action-free and action-conditioned scenarios.

These works use Neural networks intensely and model architectures are much more complex compared to mine. They also aim to predict human motions in more crowded scenarios[2]. I only use position and motion parameters due to the previous frame and time while Wang et al. [3] are using multiple spatial parameters for similar operations and also they use zigzag memory flow and customized LSTM [2], [3] which makes it superior to my project. I'm also using pre-trained models due to the computational requirements of the model and limited time while they are training their models according to the specific goal continuously.

3 Methodology

The system utilizes a combination of object detection, feature extraction, object tracking, and motion prediction techniques. The following techniques and algorithms were employed:

3.1 Object Detection

For object detection, the YOLOv4 model was utilized. YOLOv4 is a state-of-the-art deep learning-based object detection algorithm that offers high accuracy and real-time performance. It employs a convolutional neural network to detect and classify objects within an image The YOLOv4 (You Only Look Once version 4) model is a popular deep learning-based object detection algorithm. It utilizes a convolutional neural network (CNN) architecture to detect and classify objects with high accuracy and real-time performance [4]. The pre-trained model for both standard Yolov4[5] and the lightweight version of it as Yolov4-tiny[6] has been used in this project.

3.2 Feature Extraction

Object tracking is the process of following and maintaining the trajectory of objects across consecutive frames in a video sequence. To extract features from objects, the Histogram of Oriented Gradients (HOG) descriptor was employed. Histogram of Oriented Gradients (HOG) descriptor is a widely used feature extraction method. It calculates and represents the local gradients of an object's appearance. These gradients capture important information about the object's edges and texture, which is useful for object tracking and motion prediction [7].

3.3 Object Tracking

Object tracking is the process of following and maintaining the trajectory of objects across consecutive frames in a video sequence. Object tracking was performed using the CV2 tracking algorithm, specifically the CV2.TrackerCSRT_create

implementation. The algorithm utilizes a discriminative correlation filter to track objects across consecutive frames, maintaining a bounding box around the object of interest [8].

3.4 Motion Prediction

The motion prediction phase involves estimating the direction and speed of an object's motion by analyzing the changes in its position over time. To predict the object's next frame, the code calculates the displacement between the current and previous frames and estimates the object's velocity. This velocity is then used to extrapolate the object's future trajectory by multiplying it with the unit time "dt". The algorithm of the prediction process can be seen in the algorithm 1 below:

Algorithm 1 Update Bounding Box with Velocity

Require: $prev\_bbox, p1, p2, dt$
Ensure: $next\_p1, next\_p2$
while True do
  if $prev\_bbox$ is not None **then**
    $displacement \leftarrow np.array(p1) - np.array(prev\_bbox[:2])$
    $velocity \leftarrow displacement/dt$
    $next\_p1 \leftarrow tuple((np.array(p1) + velocity \times dt).astype(int))$
    $next\_p2 \leftarrow tuple((np.array(p2) + velocity \times dt).astype(int))$
  end if
end while

- $prev\_bbox$: Previous bounding box
- $p1$: Coordinates of the top-left corner
- $p2$: Coordinates of the bottom-right corner
- dt: Unit time (delta time)

Overall, the code combines object tracking with velocity estimation to predict the future trajectory of the objects based on their current positions with green rectangles and velocities. These frame predictions drew as a red rectangle onto the current frame and also drew blue arrows from the current frame corners to the next frame prediction rectangle corners to display the direction of the prediction.

3.5 Haar Cascade Method

In addition to the mentioned techniques, the Haar cascade method was incorporated into the system. The Haar cascade classifier is a machine learning-based approach that employs Haar-like features and cascading classifiers to detect objects in images or video frames. It is particularly useful for detecting faces, but

can also be trained to recognize other object classes. I have used a pre-trained frontal face detection model [9] in this project's favor.

4 Experimental Setup & Data

The experiments were conducted on a machine equipped with an Intel Core i5-1135G7 CPU. The code was implemented in a Python environment with Anaconda and methods of OpenCV and numpy libraries have been used. The system was evaluated using a pre-recorded webcam feed with a resolution of 640x480 with 20 FPS. Any pre-processing operations have been applied to the data but the methods like Haar or HOG are using the gray-scaled version of the frames so when these operations are used the frames transform into BGR to gray-scale.

5 Results and Discussion

The results demonstrate the accuracy and effectiveness of the developed system in predicting object motion. The object tracking algorithm successfully maintained track of objects across frames, providing reliable input for the motion prediction phase. The motion prediction algorithm accurately estimated the future trajectory of objects, showcasing its potential for real-world applications.

The IoU calculation is used as a measure of overlap between the current tracking bounding box and the previously predicted bounding box. It helps in assessing the accuracy of the tracking algorithm by comparing the areas of intersection and union between the two bounding boxes.

Additionally, visual examples and plots were provided to illustrate the system's performance. The configuration of the models can be stated as arguments given from the terminal while running the Python script. We can easily see how the different argument combination works from Table 1 below.

1st Arg	2nd Arg	Model Name	Input Type	Initial ROI
0	0	yolov4-tiny	Camera	Manual
0	1	yolov4-tiny	Camera	Auto
1	0	yolov4-tiny	Video	Manual
1	1	yolov4-tiny	Video	Auto
2	0	yolov4	Camera	Manual
2	1	yolov4	Camera	Auto
3	0	yolov4	Video	Manual
3	1	yolov4	Video	Auto
4	0	haar	Camera	Manual
4	1	haar	Camera	Auto
5	0	haar	Video	Manual
5	1	haar	Video	Auto

Table 1: Options

To get objective results, only the parameter combinations which use recorded video instead of the live camera feed are used. The numerical results which consist of Average IoU, Average FPS, fastest motion, and relative frame numbers can be seen in Table 2.

1st Arg	2nd Arg	Avg FPS	Avg IoU	Fastest Motion	Frame
1	0	24.46	0.92	28.01	42
1	1	12.40	0.96	29.15	39
3	0	19.38	0.94	30.59	40
3	1	10.69	0.96	30.08	39
5	0	10.71	0.94	43.32	29
5	1	25.77	0.00	33.02	28

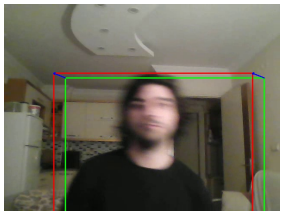
Table 2: Numerical Comparisions

As shown in Table 2, the IoU value for the Haar cascade method with Automated initial ROI is reported as 0, which is highly unlikely. To investigate this discrepancy, I conducted further analysis and discovered that selecting a smaller ROI compared to the automated ROI of the yolov4 algorithms leads to lower IoU values, and even 0, when I choose only my face as the initial ROI. The Haar cascade method with automated ROI sets the initial ROI based on the faces detected in the first frame, resulting in a significantly smaller ROI (refer to Figure 1c) compared to the automated findings of yolov4 (refer to Figure 1b, 1a). This discrepancy leads to incompatible bounding box sizes for calculating IoU in the Haar cascade method.

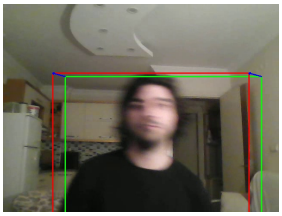
In Figure 1, we can observe all the results for video input. When option "1" is selected, which corresponds to the lightweight version of yolov4 known

as "yolov4-tiny," it achieves higher FPS but lower accuracy compared to the standard yolov4 model. However, considering the relatively small difference in accuracy, this lightweight model can be a suitable choice for storage and computational constraints. The highest FPS is achieved when the arguments are set as "5,1," which can be attributed to the smaller ROI requiring less computational power. It is worth noting that the selection of manual ROI may not be exactly the same for each case, leading to slight variations in the results for the selection of the frame with the fastest motion. When I choose a wider ROI that encompasses my head and torso, the IoU values reach more reasonable levels, as demonstrated in Table 2.

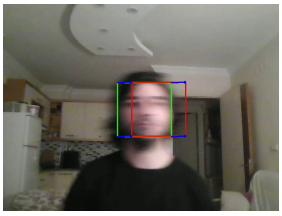
My algorithm of the yolov4 with automatic ROI selection should be optimized further to get a higher average FPS and detect more narrow/precise objects in multiple detection manner.



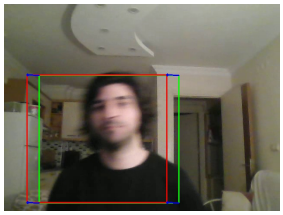
(a) YOLOv4-tiny, auto ROI



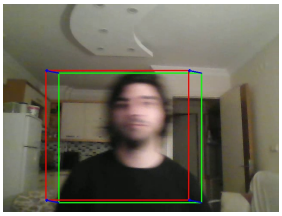
(b) YOLOv4, auto ROI



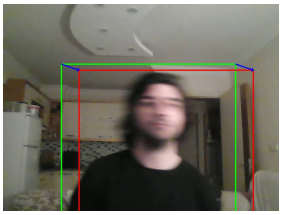
(c) Haar cascade, auto ROI



(d) YOLOv4-tiny, manual ROI



(e) YOLOv4, manual ROI



(f) Haar cascade, manual ROI

Figure 1: Fastest motion frames with different ROI and Method selections

Also, there is GPU support rather than CPU in my code but I used CPU since I do not possess a powerful GPU.

6 Conclusion

In summary, this report has investigated the prediction of object motion through the utilization of object tracking and trajectory prediction techniques. The developed system has exhibited impressive capabilities in accurately forecasting the future motion of objects. By leveraging a combination of object detec-

tion, feature extraction, object tracking, and motion prediction techniques, the system has demonstrated its potential for a wide range of computer vision applications.

Nevertheless, to fully leverage its capabilities and ensure its effectiveness in complex and dynamic scenarios, further research and development are required. Enhancements should focus on improving the system's robustness and adaptability, allowing it to handle challenging conditions and unpredictable object behaviors more effectively.

Overall, this study lays a solid foundation for future advancements in object motion prediction. With continued exploration and refinement, this field holds great promise for revolutionizing numerous domains that rely on accurate understanding and anticipation of object movements.

References

[1] Julieta Martinez, Michael J. Black, and Javier Romero. "On human motion prediction using recurrent neural networks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S. (2016). Social LSTM: Human Trajectory Prediction in Crowded Spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 961-971).

[3] Y. Wang, H. Wu, J. Zhang, Z. Gao, J. Wang, P. S. Yu, and M. Long, "PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. XX, no. X, pp. XXXX-XXXX, March 2021.

[4] Redmon, J., Farhadi, A. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[5] Darknet, "Yolov4 pre-trained model," GitHub, `https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights`.

[6] Darknet, "Yolov4-tiny pre-trained model," GitHub, `https://github.com/AlexeyAB/darknet/releases/download/yolov4/yolov4-tiny.weights`.

[7] Dalal, N., Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Vol. 1, pp. 886-893).

[8] Lukezic, A., Vojir, T., Zajc, L.C., Matas, J., Kristan, M. (2017). Discriminative Correlation Filter with Channel and Spatial Reliability. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6309-6318).

[9] Opencv, "Open CV's Default Haar-Cascade Model for Frontal Face Detection," GitHub, `https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml`.