Kalman Filter Driven Video Subsampling for Energy Efficient Computer Vision

Joshua Martin, Sameeksha Katoch, *Student Member, IEEE,* Suren Jayasuriya, *Member, IEEE,* and Andreas Spanias, *Fellow, IEEE*

Abstract— **A central challenge in object tracking for embedded computer vision is the large amount of energy consumed during image sensing. This REU study uses a Kalman filter alongside a pre-trained convolutional neural network to track an object's position and size through consecutive video frames. The filter alternates between updating its estimate of the object's location based on new measurements and predicting its location based solely on the filter's mathematical model of the object's trajectory. These predictions of the object's position and size are then used to subsample only the region of the image containing the object. Experiments have shown average energy savings of up to 65% with 24% loss in mean average precision when run on videos from the TB100 and ILSVRC2015 data sets. In summary, our study provides a new method for video subsampling that shows promise for increasing the battery life of embedded devices with minimal loss in task accuracy.**

Index terms: Embedded system, computer vison, Kalman filter, object tracking/detection.

I. Introduction

A major issue in embedded image sensing and object tracking applications is the amount of energy required for image sensing. Applications ranging from autonomous vehicles to augmented reality headsets are made possible due to recent advances in object detection and tracking algorithms and the development of hardware powerful enough to run them. However, many of these embedded devices experience poor, if not abysmal, battery life in part due to the energy requirements imposed by their camera(s). For instance, the Google Glass headset attains a meager 45-minute battery life under continuous use, and image sensing accounts for nearly 50% of its power usage [1].

Previous research has developed methods for energy efficient object tracking, most commonly using image subsampling, where certain rows and columns of pixels are turned off or the resolution of the image is reduced by taking averages or maxima of surrounding pixels [2]. A recent study used an algorithm to generate an objectness map which identified regions of interest which they used to create an image mask [3]. This mask was used to determine which pixels to disable. They compared this approach to disabling random pixels throughout the image in an attempt to maintain image coherence [3]. However, these methods have several limitations. The row/column disabling, binning, and random disabling methods all suffer from reduced accuracy due to loss of image features of the object(s) of interest. Methods using objectness metrics require frequently capturing new reference frames whenever the object moves out of the subsampled area, thus limiting the amount of energy saved [3].

In this study, we address some of these issues specifically the loss of image features of the object(s) of interest by utilizing a Kalman filter. We begin by using a pre-trained convolutional neural network (CNN) to identify the location and bounding box of an object in video frames. This information is then fed into a Kalman Filter which will update an estimate of the object's location [4]. The system alternates between running object detection on fully sampled frames to update the estimate, and outputting predictions of the object's state based solely on the filter's learned model. This prediction will be used to determine which areas of the image to capture and which to disable. The goal is to outperform the previously mentioned methods of subsampling in terms of average number of pixels disabled while maintaining accuracy and image features.

II. Method

In short, our algorithm uses an object detector to measure the position and size of objects in video frames. These measurements are used by a Kalman filter to track the object and predict where it is going to be. These predictions are used to sample only the regions of the frame where the object is anticipated to be.

A. Kalman Filter

The Kalman Filter is an efficient mathematical solution to the problem of discrete signal estimation [5]. Put more simply, it takes in noisy measurements of a dynamic system and recursively updates an estimate of the system's state (in this case, position and size). The filter is comprised of a set of linear equations that can be broken up into two distinct phases:

Predict:

State Prediction: $$x_{pred} = Ax_{n-1} \quad (1)$$

Covariance Prediction: $$P_{pred} = AP_{n-1}A^T + Q \quad (2)$$

Update:

Innovation: $$\mathrm{y} = z_n - Hx_{pred} \quad (3)$$

Innovation Covariance: $$\mathrm{S} = HP_{pred}H^T + R \quad (4)$$

Kalman Gain: $$\mathrm{K} = P_{pred}H^T S^{-1} \quad (5)$$

State Update: $$x_n = x_{pred} + Ky \quad (6)$$

Covariance Update: $$P_n = (I - KH)P_{pred} \quad (7)$$

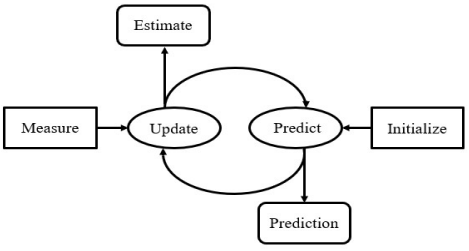

Figure 1. Logical flow of the Kalman filter

where x represents predicted state vector, A represents state transition matrix, Q represents process noise covariance, z represents measurement vector, H represents state transition matrix, R represents measurement covariance matrix and I represent identity matrix.

The prediction equations (1 and 2) use the mathematical model of the system, in the form of the state transition matrix, to project the current system state one time-step into the future. The update equations then take in a measurement at that time and compare it to the predicted (projected) state. The difference, or 'innovation', is then used to find the Kalman Gain (K). Consequently, a new state estimate is calculated using K to determine how much the new measurement should be weighted.

B. Tiny-YOLO (CNN)

You Only Look Once (YOLO) is a recent convolutional neural network architecture which boasts improvements in accuracy over other state of the art architectures while also vastly improving on execution time [7]. As the name would suggest, unlike other modern object detection algorithms which iteratively perform classification on numerous subsections of the image, like RCNN (Regions with CNN features), YOLO needs to execute only one pass through its network in order to predict bounding boxes and class probabilities [8].

Tiny-YOLO (TY) is a scaled down version of the network which is less accurate, but also runs significantly faster (up to 155 frames per second (FPS) on a Nvidia Titan X GPU[7]). TY's single pass approach and scaled down network size make it attractive for use in embedded devices with limited resources.

C. Algorithm/System Design

Our algorithm uses Tiny-YOLO (TL) and a constant velocity Kalman Filter to detect objects of interest in video and to track them through subsequent frames. As shown in Figure 3., we alternate between update and prediction phases in order to accurately sample only the regions of each frame containing the object of interest.

During the update phase, a fully sampled reference frame is captured. The first step in our pipeline is object location detection using TY which results in an output in the form of a bounding box (x, y, height, width). The following step

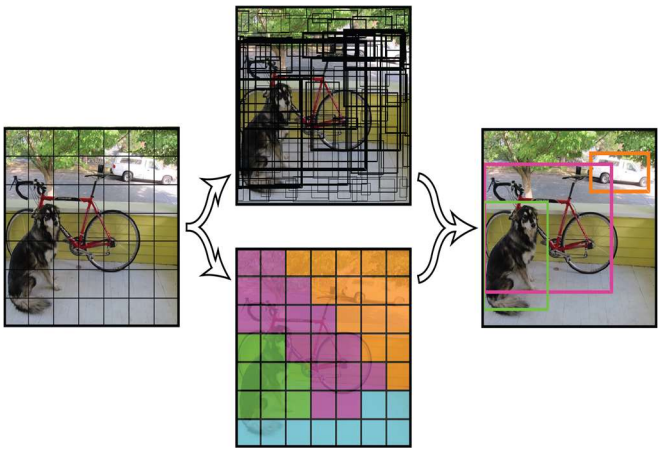

Figure 2. Visualization of the YOLO algorithm [7]

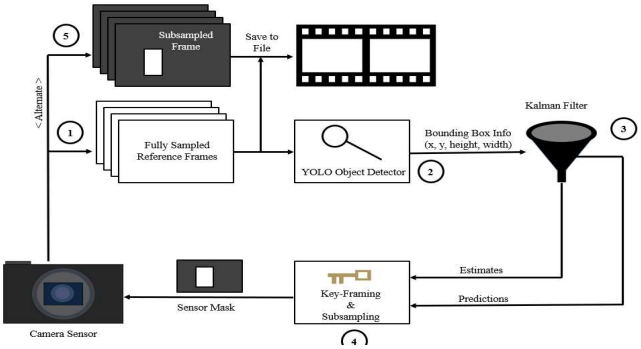

Figure 3. Proposed algorithm consisting of (1) sampling, (2) detection, (3) update/predict, (4) mask creation, and (5) image subsampling the subsequent frames.

involves passing the bounding box parameters into the Kalman filter. The Kalman filter takes the bounding box as a measurement and uses it to update its internal state estimate vector.

In the prediction phase, the two prediction equations (Eq. 1 and Eq. 2) of the Kalman Filter are used to extrapolate the object's current trajectory for both its position and bounding box dimensions. Consequently, a new bounding box is obtained which is used to create a pixel mask and determine specific regions of the image that should be subsampled (i.e. omitted).

Rather than using the predicted bounding box itself as the mask, we opted to subsample based on which cells in a 7x7 grid the bounding box falls within (shown in Figure 4.). This is in consistency with the actual TY algorithm which divides the image into a 7x7 grid.

Expanding the mask to these predefined cell boundaries serves two purposes and has one main drawback. First, increasing the subsampled area serves to preserve more information and thus increases the accuracy of future computer vision tasks performed on the video (see Table. 2 on the following page). It provides a regularly defined buffer zone in case inaccuracies in either TY's detections or the Kalman Filter's predictions lead to portions of the object falling outside the predicted bounding box. Second, the fixed grid-size also provides advantages in terms of implementing the algorithm on imaging sensor hardware since arbitrary

Figure 4. Difference between bounding box mask and cell-based mask

Accuracy loss and energy savings	Number of predictions per update						
	1	5	10	15	20	25	30
mAP/mAP_full	0.930	**0.807**	0.752	0.734	0.814	0.716	0.703
mAP/mAP_ground	1.012	**0.854**	0.784	0.766	0.877	0.734	0.729
Pixel ratio	0.352	**0.590**	0.646	0.659	0.663	0.679	0.682
IOU/IOU_full	1.009	**0.958**	0.923	0.900	0.896	0.887	0.877
IOU	0.680	**0.647**	0.625	0.610	0.608	0.601	0.595

Table 1. Comparison of accuracy and energy savings metrics between various prediction levels.

Effect of cell borders on mAP	Number of predictions per update						
	1	5	10	15	20	25	30
Cell border	0.930	0.807	0.752	0.734	0.814	0.716	0.703
Box border	0.743	0.594	0.516	0.453	0.508	0.501	0.503

Table 2. Comparison of accuracy (mAP) between two methods for creating the subsampling mask.

shaped ROIs are difficult to implement in embedded systems. The camera that is currently being designed using our base algorithm works optimally with rectangular and consistent masks. However, the drawback of using this sizing scheme is the seemingly inherent tradeoff between information preserved and energy expended; by increasing the size of the mask we reduce the energy conserved. Although, subsampling via this grid method is partially arbitrary, in the sense that any other grid size could be used (e.g. 6x6, 8x8, etc.). We leave this exploration of sizing scheme as a part of future work.

III. Results

A. Data Set Description

A majority of the testing and development of the project has been done on the TB100 data set, which is a set of 100 videos of various objects with different types of aberrations (e.g. sporadic motion, blurriness, occlusion, etc.) [9]. Since the scope of the project is currently limited to single object tracking, and the fact that the pretrained Tiny-YOLO weights are trained to detect 20 classes of objects, we selected only the videos that met those criteria [7]. In order to test our algorithm on a larger and more robust data set, we also took videos from the ImageNet Large Scale Visual Recognition Challenge 2015 (ILSVRC2015) [10].

B. Qualitative Results

The predicted and subsampled output video frames from the system look promising, however certain types of aberrations cause either the detector or the filter to lose accuracy, which is expected. For example, we have found that partial object occlusion causes TY to lose confidence in its predictions which can lead to oversized bounding boxes (Figure. 5).

Additionally, sporadic and/or rapid motion of either the object or the camera can cause the Kalman filter to lose the object temporarily if it is performing many predictions in a row, or to lag behind the object if the change in velocity is sharp enough. There are methods to compensate for the latter, in the form of an adaptive Kalman Filter, which we hope to look into in future work [11].

C. Quantitative Results

Metrics: In our project, we focus on three main metrics- Intersection over Union (IOU) with respect to the ground truth, which is the ratio between the overlap of two bounding boxes and their total combined area, Mean Average Precision (mAP), and Energy saved, measured by the ratio of deactivated pixels to total pixels.

More specifically, we look at the performance (mAP) of TY when run on the subsampled video frames output by the system relative to two references- the performance of TY when run on the fully sampled frames (mAP_full) and when run on frames subsampled according to the ground truth bounding boxes (mAP_ground). The point is to gauge how much accuracy is lost from subsampling and to see the difference between the system's subsampling and the best-case scenario (subsampled according to the ground truth).

For the purpose of our study, mAP is calculated as the number of frames in a video in which TY's output correctly

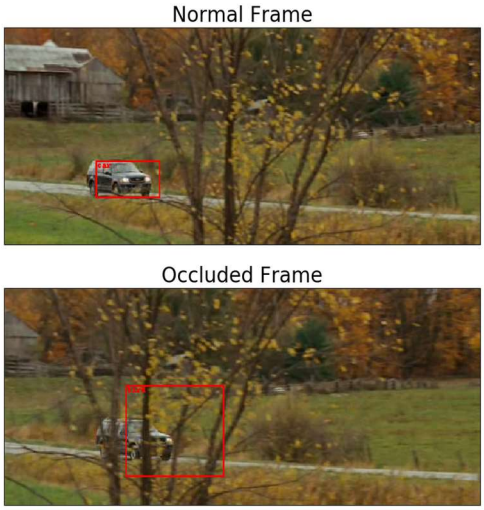

Figure 5. Loss of bounding box accuracy due to occlusion

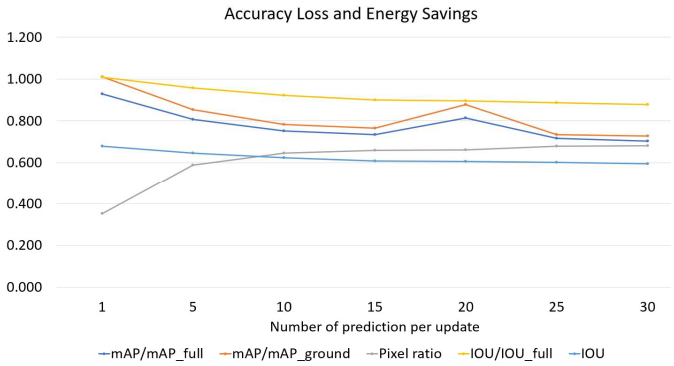

Figure 6. Graph of accuracy and energy savings for varying levels of prediction. The optimal trade off point between accuracy and power efficiency is at the 5 predictions level, with a accuracy loss of 19.3% and an energy savings of 59%.

identifies the object's class and has an IOU greater than 0.5 with the ground truth divided by the total number of frames.

These three criteria are each evaluated at seven different prediction levels ranging from 1 to 30, as shown in Figure. 6. These prediction levels are the number of predictions made per update. i.e. the greater the prediction level, the greater the energy savings, but the greater the expected loss in accuracy. Performing this ablation study, gives us a clear idea that prediction level 5 works well in terms of Energy savings while maintaining detection accuracy.

As evident from the results in Table 1. And Figure 6., as we increase the number of predictions from one, to ten the energy savings (shown here as the "Pixel ratio") quickly jumps up to nearly 65% while only losing 24.8% mAP relative to the fully sampled reference run. Further increases to the number of predictions yield diminishing returns in terms of energy savings, while the mAP continues to fall. Therefore, depending on the video, the optimal trade of between accuracy and power efficiency is somewhere in the range of 5-10 predictions.

We plot the IOU/IOU_full to highlight the fact that at one prediction per update IOU is *higher* than the reference, and overall it falls far slower than the mAP. We believe this indicates that the drop in mAP has more to do with TY misclassifying the subsampled images and not because of inaccurate tracking. We plan to investigate this further in future research by using different object detector networks to perform evaluation.

Lastly, as mentioned earlier, Table 2. shows the positive effect of the cell-boundary based masks on mAP. The use of these larger masks increases the mAP by 30.3% on average.

IV. Discussion and Future Work

Our proposed algorithm for video subsampling shows potential for embedded devices performing computer vision tasks. It has the advantage of using a relatively computationally cheap CNN for its detections, and it uses predictive tracking so as to avoid frequently capturing reference frames. A downside is that unless an application is only interested in tracking objects that belong to the 20 classes that Tiny YOLO (TY) was trained on, TY will need to be trained on new data. Fortunately, TY is just the network we happened to use. Any detector that returns bounding boxes for objects of the class of interest will work, so long as the embedded device has the resources to run the detector.

In the future, we have several avenues for further study we would like to explore. As previously mentioned, we want to look into implementing adaptive Kalman filtering, which involves adaptively adjusting the filter's covariance matrices to reduce tracking lag and jitteriness. Additionally, in its current state the system can only handle having one object of interest in the scene at a time. We wish to make it so that the system maintains separate Kalman filters for every object in the scene and can differentiate multiple objects. Finally, during testing it became apparent that different prediction phase lengths performed better or worse depending on the video, so we plan on looking into adaptively changing the number of predictions made per update.

References

[1] R. LiKamWa, Z. Wang, A. Carroll, F. X. Lin, and L. Zhong, "Draining our glass: An energy and heat characterization of google glass," in *Proc. of 5th Asia-Pacific Workshop on Systems.* ACM, 2014, p. 10.

[2] R. LiKamWa, B. Priyantha, M. Philipose, L. Zhong, and P. Bahl, "Energy characterization and optimization of image sensing toward continuous mobile vision," in Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services. ACM, 2013, pp. 69–82.

[3] D. Mohan, S. Katoch, S. Jayasuriya, P. Turaga, and A. Spanias, "Adaptive video subsampling for energy-efficient object detection."

[4] N.Kovvali, M. Banavar, A. Spanias, An Introduction to Kalman Filtering with MATLAB Examples, Morgan & Claypool Publi., Ed. J. Mura, vol. 6, pp. 1-81, ISBN 13: 9781627051392, September 2013.

[5] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," University of North Carolina, 24-Jul-2006. [Online]. Available: https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf. [Accessed: 08-Jul-2019].

[6] G. Czerniak, "Greg Czerniak's Website," Greg Czerniak's Website - Kalman Filters for Undergrads 1. [Online]. Available: http://greg.czerniak.info/guides/kalman1/. [Accessed: 08-Jul-2019].

[7] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779-788. 2016.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Computer Vision and Pattern Recognition(CVPR), 2014 IEEE Conference on, pages 580–587. IEEE, 2014.

[9] "Visual Tracker Benchmark," Visual Tracker Benchmark. [Online]. Available: http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html. [Accessed: 09-Jul-2019].

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211-252, 2015.

[11] X. Li, T. Zhang, X. Shen and J. Sun, "Object Tracking using an Adaptive Kalman Filter *Combined with Mean Shift," Optical Engineering*, vol. 49, no. 2, p. 020503, 2010.