DESIGN AND FPGA IMPLEMENTATION OF AN ADAPTIVE VIDEO SUBSAMPLING ALGORITHM FOR ENERGY-EFFICIENT SINGLE OBJECT TRACKING

Odrika Iqbal$^{\star,+}$, *Saquib Siddiqui**, *Joshua Martin*$^{+}$, *Sameeksha Katoch*$^{\star,+}$,
Andreas Spanias$^{\star,+}$, *Daniel Bliss**, *Suren Jayasuriya*$^{\star,\dagger,+}$

* School of Electrical, Computer and Energy Engineering, Arizona State University
† School of Arts, Media and Engineering, Arizona State University
$^{+}$SenSIP Center, Arizona State University

ABSTRACT

Image sensors with programmable region-of-interest (ROI) readout are a new sensing technology important for energy-efficient embedded computer vision. In particular, ROIs can subsample the number of pixels being readout while performing single object tracking in a video. In this paper, we develop adaptive sampling algorithms which perform joint object tracking and predictive video subsampling. We utilize an object detection consisting of either mean shift tracking or a neural network, coupled with a Kalman filter for prediction. We show that our algorithms achieve mean average precision of 0.70 or higher on a dataset of 20 videos in software. Further, we implement hardware acceleration of mean shift tracking with Kalman filter adaptive subsampling on an FPGA. Hardware results show a 23$\times$ improvement in clock cycles and latency as compared to baseline methods and achieves 38FPS real-time performance. This research points to a new domain of hardware-software co-design for adaptive video subsampling in embedded computer vision.

Index Terms— FPGA acceleration, embedded computer vision, single object tracking, adaptive subsampling

1. INTRODUCTION

Embedded systems typically trade-off generality in order to perform specific tasks at high fidelity, often via hardware specialization and optimization. In particular, embedded computer vision is an emerging field where embedded systems and image sensors are jointly optimized for image and video processing. This has tremendous potential for applications such as autonomous driving, drones, and robotic platforms.

However, the key challenge to realizing embedded computer vision is to enable low-power processing of visual data such that it can be supported by the battery life of the mobile/embedded platform. While there have been several efforts in reducing the computational load of vision algorithms through hardware acceleration, there has been less focus on the costs of image sensing energy as the front end sensor in the pipeline. For example, the Google Glass runs out of battery in 45 minutes running continuous face detection, and consumes 50% of that power on image sensing alone [1]. Thus there is an opportunity to jointly design image sensors and algorithms for embedded computer vision platforms.

Existing image sensors can reduce power in the analog domain via reading out smaller regions of interest (ROIs). Not only does this reduce image quantization and bandwidth costs, but the other pixels not in the ROI can be switched off or power-gated for additional savings. In addition, the smaller ROI image can help speed up digital processing on-board the embedded platform by decreasing latency and clock cycles. In this paper, we leverage the ROI capabilities of sensors to design a new class of adaptive video subsampling algorithms, which only readout salient pixels to perform an end vision task. Our main application is single object tracking, where only a small ROI around a moving object is necessary for tasks including surveillance and autonomous driving.

Our specific contributions are the design of video adaptive subsampling algorithms along with an embedded system implementation onto an FPGA. We utilize off-the-shelf detectors, both mean shift tracking as well as the TinyYOLO deep neural network, coupled with a Kalman filter to jointly track an object while subsampling frames of a video. The algorithm thus sends a control signal to the image sensor to intelligently adapt its ROI to the moving object. We validate our algorithms on a small video dataset, and show that mean-shift tracking slightly outperforms the tiny-YOLO network in this task. Further, we accelerate the mean shift tracking + Kalman filter algorithm onto a Xilinx Zynq FPGA to measure the performance on real hardware. We show that our algorithms achieve mean average precision of 0.70 or higher and hardware acceleration achieves a 23$\times$ improvement in clock cycles and latency as compared to baseline methods and achieves 38FPS real-time performance.

2. RELATED WORK

Energy-efficient object tracking has been studied extensively [2–6]. Similar to our paper, other works have also exploited the Kalman filter for tracking objects [7–10]. State-of-the-art

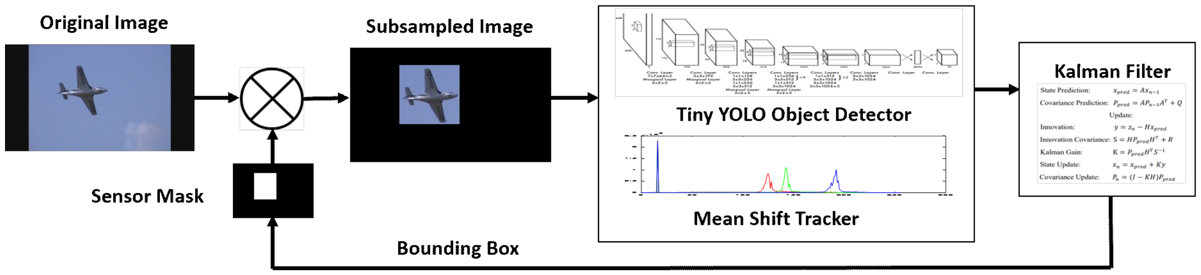

Original Image
Subsampled Image
Sensor Mask
Tiny YOLO Object Detector
Mean Shift Tracker
Kalman Filter
State Prediction: $x_{pred} = Ax_{n-1}$
Covariance Prediction: $P_{pred} = AP_{n-1}A^T + Q$
Update:
Innovation: $y = z_n - Hx_{pred}$
Innovation Covariance: $S = HP_{pred}H^T + R$
Kalman Gain: $K = P_{pred}H^TS^{-1}$
State Update: $x_n = x_{pred} + Ky$
Covariance Update: $P_n = (I - KH)P_{pred}$
Bounding Box

Fig. 1: Proposed FPGA-based image subsampling algorithm featuring an object detector coupled with a Kalman filter.

object tracking utilizes deep neural networks [11, 12]. However, none of these works deal with adaptive image subsampling.

There is growing interest in the notion of region of interest (ROI) and adaptive subsampling. ROI refers to the part of an image frame which contains the object of interest, and adaptive subsampling refers to the discarding of spatially distributed pixels outside of the ROI. ROIs have been exploited for image compression [13, 14]. The work most closely aligned to ours is objectness-based subsampling [15]. The algorithm localizes the object of interest by utilizing a feature called objectness to compute a heat map of likely object locations, and then using Otsu's threshold to determine an ROI and subsampling mask. However, this algorithm does not use tracking or motion information explicitly, and must sample a full frame when significant motion occurs. In our paper, we adaptively change the ROI with moving objects using both mean shift tracking as well as a neural network, and predictively estimate the next ROI to sample in future frames.

3. METHODS

Our tracking algorithm is comprised of two key components: a predictor (Kalman filter) which features ROI-ing capabilities, and an object detector (mean shift or neural network) as the measurement generator for the predictor. We alternate between detection and prediction with a fixed ratio. In hardware, this ratio is fixed at $1 : 5$ and in software we switch between detection and prediction every other frame.

Kalman Filter Tracking. The Kalman filter is used to keep track of the moving objects and predict their locations in subsequent frames [16]. In the **update phase**, the object detector algorithm outputs the bounding box of the moving object and this is fed as input to the Kalman filter. The filter accepts this input as a new measurement and it self-updates its internal state estimate vector based on the incoming bounding box. In the **prediction phase**, the filter's state space matrix is used to predict the position of the moving object in the adjacent frame. The predicted position gives the new bounding box dimensions and this new box is used to generate a mask that turns pixels off outside the area containing the object of interest.

Tiny-YOLO CNN. We utilize the pre-trained tiny-YOLO convolutional neural network (CNN) object tracker for the measurement phase of the Kalman filter for predicting future regions of interest. The YOLO neural network architecture proposed by Redmon et al. [12] is a state-of-the-art real-time object detector and is a good candidate for tracking applications. Redom et al. have modeled the object detection task as a regression problem, and outputs the bounding box along with the class probability. Tiny-YOLO can optimize the size of its ROI by changing the width and height of the bounding box frame to frame. Additionally, its lightweight architecture and its end-to-end optimization framework make it an attractive object detector for mobile and real-time applications.

Mean Shift Tracking. The mean shift tracking algorithm estimates how the color histogram values of the ROI move as a cluster over time in the image using non-parametric methods [17]. We note that previous research has also proposed coupling mean shift tracking with Kalman filtering [18]. However, they do not consider explicitly the case of adaptive subsampling, and they do not explore hardware acceleration of this algorithm as we do.

One limitation of mean shift includes requiring an object detector for the first iteration to select the initial ROI. However, once it receives the initial bounding box it computes the color histogram for every frame and uses that to find the centroid of the ROI. Note that that the width and height do not change iteratively which means constant energy savings.

4. FPGA IMPLEMENTATION

We choose field programmable gate array (FPGAs) as our hardware platform due to a host of attractive qualities including reprogammability, low latency, high bandwidth, and onboard memory for hardware acceleration. In addition to this, one particular application of interest for object tracking and adaptive subsampling is identifying targets on space imaging platforms where radiation is a concern. Rad-hard FPGAs is a growing market for space-bound applications where flexibility and dataflow-intensive processing is desired. Considering latency as our metric for evaluation, FPGAs have been known to come out ahead of both CPUs and GPUs for imaging applications [19]. While ASICs can potentially alleviate

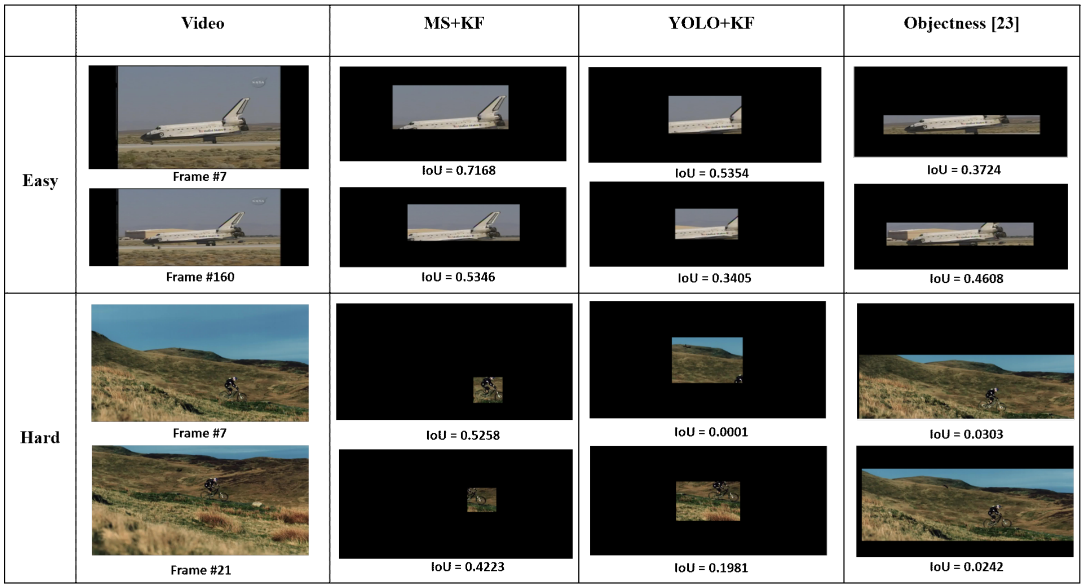


Fig. 2: Visualization of the three algorithms tracking objects with image subsampling.

these issues, the hardware benefits come at the cost of programmability and flexibility of use. Further, there is a lack of fully developed, user friendly high level synthesis toolflows for ASICs to accelerate vision algorithms.

For the hardware implementation of our algorithm, we use Xilinx's Zynq Ultrascale+ MPSoC ZCU102 board. The toolkit used to program the board is Xilinx's SDSoC (Software Defined System on Chip Platform). The SDSoC environment includes a full-system optimizing C/C++ compiler that provides automated software acceleration in programmable logic combined with automated system connectivity generation. As such, simply writing out our code in C++ sufficed. Verilog code for the input/output port connectivity is auto-generated when we synthesize our C++ code in the SDSoC environment. Verilog code generation, placing, routing and bitstream generation all happen in the SDSoC environment. In order to communicate with the FPGA, we set up the SD card boot mode and use an SD card to load our program and input videos onto the board.

5. RESULTS

Dataset. In order to evaluate our algorithm, we used 20 single object tracking videos from the ILSVRC2015 dataset [20]. We picked only single object videos for our experiments and we avoided videos with excessive clutter and blur. This is a limitation of our method, and remains future work to handle these cases effectively. The set of videos we picked included simple test cases as well difficult ones. A few examples of the videos we used are - a turtle walking slowly (easy), a boat speeding away in water (medium), a small animal running fast (difficult).

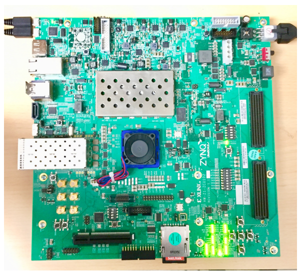
Baseline. We compare against the baseline objectness-based adaptive video subsampling technique [15]. The inherent limitations of the algorithm are reflected quite clearly in Table 1. Even when the algorithm narrows in on the object of interest, it shrinks the enclosing bounding box to capture just parts of the object. Both our algorithms outperformed the baseline and motivated hardware acceleration for MS+KF.

Metrics	MS+KF	NN+KF	OB
mAP Mean	0.7677	0.7385	0.4598
mAP Std.	0.1960	0.3437	0.3194
Mean Pixel Off Ratio	0.8412	0.6930	0.9467

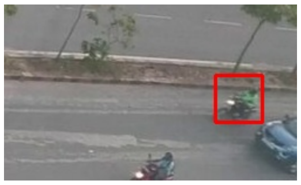
Table 1: Performance Evaluation on ILSVRC2015 (20 Videos).

Metrics. Our metrics for evaluation were the mean average precision (mAP) and the ratio of pixels switched off. The mAP was determined by counting the number of frames in a video for which the intersection over union (IoU) of the predicted bounding box and ground truth bounding box is greater than 0.5. The mAP gives a measure of how accurately an object in motion is being tracked. For adaptive subsampling performance, we also compute and report the ratio of pixels switched off. This metric is determined by dividing the number of pixels outside of the predicted ROI by the total number of frame pixels. We note that image sensor energy corresponds to number of pixels turned off (see a useful breakdown of image sensing energy here [21]).

Software Results. In Table 1, we report the performance for the mean shift plus Kalman filter (MS+KF), the tiny-YOLO tracking method (NN+KF) and the objectness algorithm (OB). For the subset of videos that we have used for running our experiments, the MK+KF algorithm outperformed the other two in terms of precision (0.7677 ver-



(a) Hardware Setup



(b) FPGA Output

Fig. 3: Hardware implementation of our object tracking algorithms on (a) Xilinx Zynq ZCU102 board, with the resulting output displayed in (b).

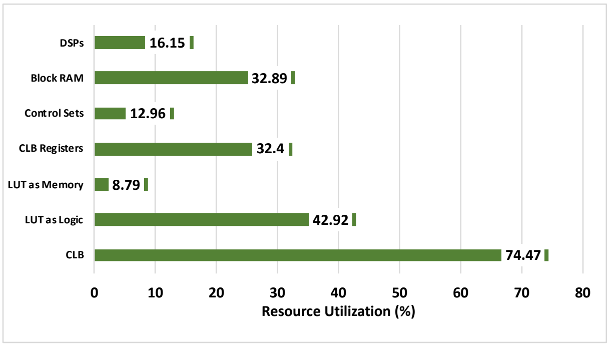


Fig. 4: FPGA resource utilization

sus 0.7385, 0.4598). The energy savings associated with switching pixels off for the MS+KF technique is also high (0.8412). While there is not too much gap between the mAP for MS+KF and NN+KF, we note that the neural network has a larger standard deviation in precision than the MS+KF.

Even if an algorithm identifies the object, the tracking precision could end up being poor if the algorithm picks an ROI smaller than the object. Such an instance of poor ROI selection by the algorithm has been illustrated in Figure 2. For the easy case, we see that the MS+KF algorithm achieves higher IoU for both frames than the NN+KF algorithm. The neural network appears to be emphasizing on the writing on the plane wing as the object of interest whereas the MS+KF is capturing more accurate ROIs for both the cases. Analysing the hard case in Figure 2,we see that MS+KF identifies the small target, while NN+KF and objectness progressively do worse (respectively).

Overall, the performance of the mean shift based algorithm is comparable to the neural network based tracker and superior to the objectness method. To estimate the speed of each method, we evaluated the algorithm execution time in software. Running mean shift on a single frame takes approximately 0.5389 seconds. On the other hand, tiny-YOLO and objectness take 1.7813 and 6.2506 seconds respectively. In our hardware acceleration, we thus chose the mean shift plus Kalman filter algorithm due to its performance, execution time in software, and availability of computer vision libraries for Xilinx FPGAs which can easily be adapted to implement this algorithm. We note that it is very difficult to accelerate neural networks on FPGAs because of associated resource constraints, although it remains future work to explore the DNNDK library for achieving neural network acceleration.

Hardware Results. Figure 3a and Figure 3b demonstrate our hardware setup and an FPGA generated image respectively. Figure 4 demonstrates the resource utilization for the MS+KF algorithm and that resource constraints of the board were satisfied.

We have also found that the image resolution affects the speed of the FPGA. With a resolution of 250×150, the algorithm runs at 6.24×10^5 clock cycles and with a resolution of 1920×1080, the clock frequency goes up to 1.5×10^6. This is a promising result in the context of adaptive subsampling. Truncating images on the basis of ROIs implies fewer pixels for the tracking algorithm to process.

The mean shift run in software requires 1.4×10^8 clock cycles whereas in hardware it requires just 1.5×10^7 cycles. The Kalman filter on hardware requires 6.24×10^5 clock cycles. Coupling the two, the requirement is 34.99×10^5 cycles on average, which corresponds to an effective latency of 26 ms and real-time frame rate of **38 FPS** with a clock frequency of 75 MHz on the FPGA. This corresponds to a **23x speedup** in hardware clock cycles.

6. DISCUSSION

In this paper, we introduced an adaptive subsampling algorithm coupling object detectors with a predictive Kalman filter for single object tracking. For mean shift and tiny-YOLO, we show mean average precision of 0.7677 and 0.7385 on a selection of 20 videos from the ILSVRC2015 dataset. We accelerated the mean shift with Kalman filter algorithm on an FPGA, and achieved $23\times$ speedup compared to object tracking alone, with real-time performance of 38 FPS. Our method is advantageous in that it allows high object tracking precision while saving large amounts of pixels (84% turned off) and achieving real-time performance. One limitation of our technique is the inability to handle multi-object tracking, clutter, and blur. Future work includes developing neural network solutions such as deep reinforcement learning to improve performance for these challenging cases. In addition, we plan to use emerging FPGA libraries to accelerate these networks on devices. We hope that this research opens up new capabilities for embedded object tracking systems in the future.

Acknowledgements: This work was supported by the NSF REU Site CCF-1659871, NSF CCF-1909663, the SenSIP Center at ASU, and Qualcomm. We also thank Mohit Malu from SenSIP Center as well as Sai Medapuram and Matt Engelman from Alphacore for useful discussions.

7. 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 *Proceedings of 5th Asia-Pacific Workshop on Systems*, 2014, pp. 1–7.

[2] Y. Xu and W.-C. Lee, "On localized prediction for power efficient object tracking in sensor networks," in *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.* IEEE, 2003, pp. 434–439.

[3] M. Casares and S. Velipasalar, "Adaptive methodologies for energy-efficient object detection and tracking with battery-powered embedded smart cameras," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 10, pp. 1438–1452, 2011.

[4] J.-M. Hsu, C.-C. Chen, and C.-C. Li, "Poot: An efficient object tracking strategy based on short-term optimistic predictions for face-structured sensor networks," *Computers & Mathematics with Applications*, vol. 63, no. 2, pp. 391–406, 2012.

[5] J. A. Fuemmeler and V. V. Veeravalli, "Energy efficient multi-object tracking in sensor networks," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3742–3750, 2010.

[6] L. Yang, J. Cao, W. Zhu, and S. Tang, "Accurate and efficient object tracking based on passive rfid," *IEEE Transactions on Mobile Computing*, vol. 14, no. 11, pp. 2188–2200, 2015.

[7] X. Li, K. Wang, W. Wang, and Y. Li, "A multiple object tracking method using kalman filter," in *The 2010 IEEE International Conference on Information and Automation.* IEEE, 2010, pp. 1862–1866.

[8] J. Black, T. Ellis, and P. Rosin, "Multi view image surveillance and tracking," in *Workshop on Motion and Video Computing, 2002. Proceedings.* IEEE, 2002, pp. 169–174.

[9] L. Marcenaro, M. Ferrari, L. Marchesotti, and C. S. Regazzoni, "Multiple object tracking under heavy occlusions by using kalman filters based on shape matching," in *Proceedings. International Conference on Image Processing*, vol. 3. IEEE, 2002, pp. III–III.

[10] D. Y. Kim and M. Jeon, "Data fusion of radar and image measurements for multi-object tracking via kalman filtering," *Information Sciences*, vol. 278, pp. 641–652, 2014.

[11] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5620–5629.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[13] W. Lin and L. Dong, "Adaptive downsampling to improve image compression at low bit rates," *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2513–2521, 2006.

[14] R. A. Belfor, M. P. Hesp, R. L. Lagendijk, and J. Biemond, "Spatially adaptive subsampling of image sequences," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 492–500, 1994.

[15] D. Mohan, S. Katoch, S. Jayasuriya, P. Turaga, and A. Spanias, "Adaptive video subsampling for energy-efficient object detection," in *Asilomar Conference on Signals, Systems, and Computers*, 2019.

[16] R. E. Kalman, "A new approach to linear filtering and prediction problems," vol. 82, pp. 32–45.

[17] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.

[18] J. Ren and J. Hao, "Mean shift tracking algorithm combined with kalman filter," in *2012 5th International Congress on Image and Signal Processing.* IEEE, 2012, pp. 727–730.

[19] S. Asano, T. Maruyama, and Y. Yamaguchi, "Performance comparison of fpga, gpu and cpu in image processing," in *2009 International Conference on Field Programmable Logic and Applications.* IEEE, 2009, pp. 126–131.

[20] 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.

[21] V. Kodukula, S. Katrawala, B. Jones, C.-J. Wu, and R. LiKamWa, "Stagioni: Temperature management to enable near-sensor processing for energy-efficient high-fidelity imaging," *arXiv preprint arXiv:2001.01580*, 2019.