Object Detection in Real-time Systems: Going Beyond Precision

Anupam Sobti IIT Delhi

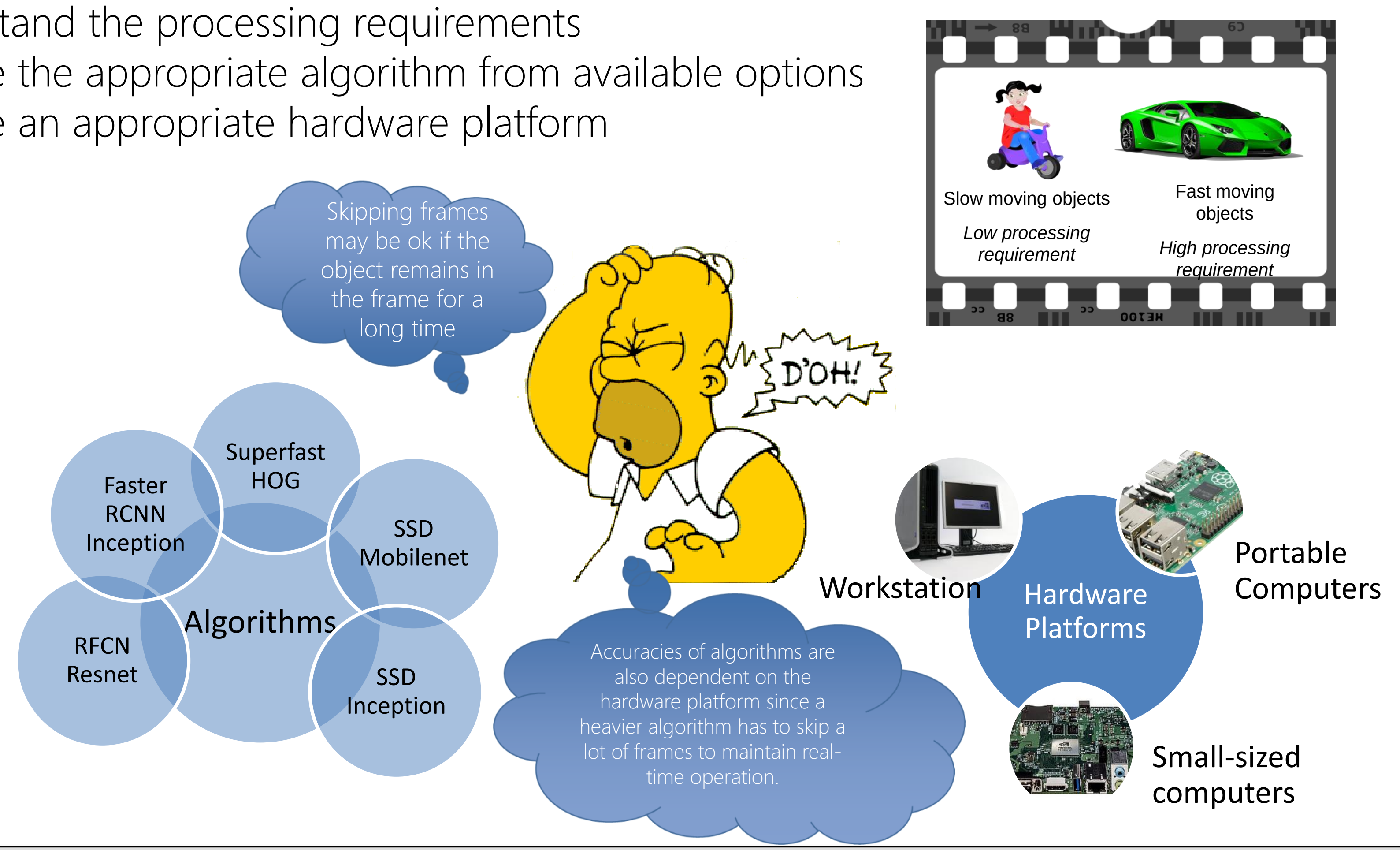
Chetan Arora IIIT Delhi

M. Balakrishnan IIT Delhi

The problem

A real-time object detection system designer has to

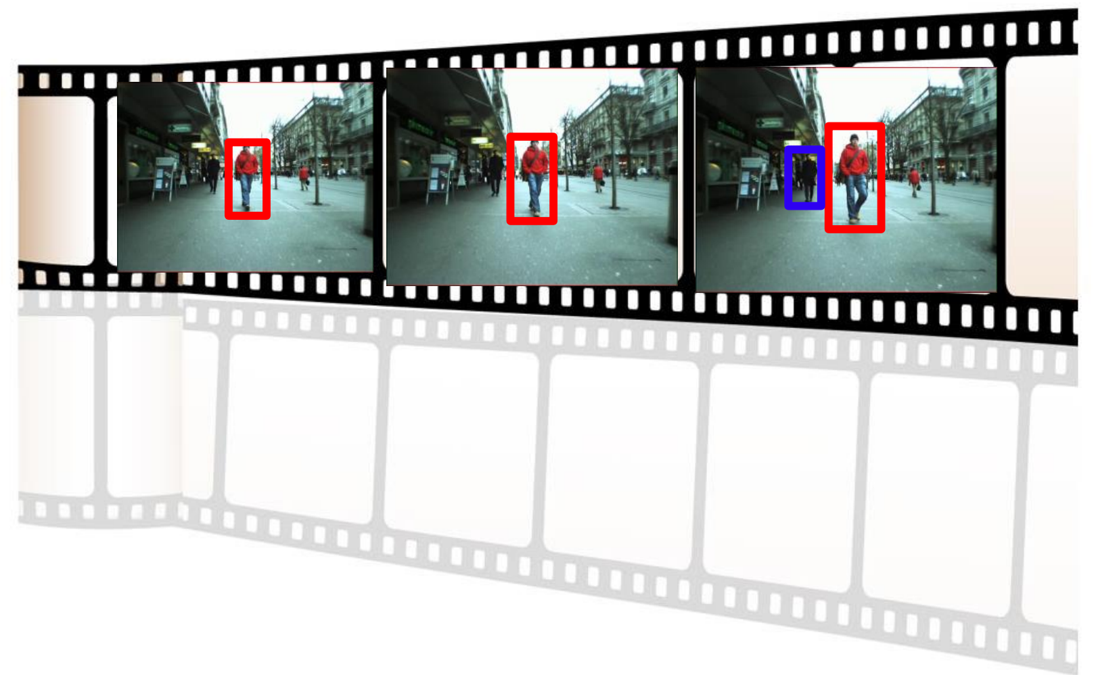
- Understand the processing requirements
- Choose the appropriate algorithm from available options
- Choose an appropriate hardware platform


Slow moving objects
Low processing requirement
Fast moving objects
High processing requirement
Skipping frames may be ok if the object remains in the frame for a long time
D'OH!
Faster RCNN Inception
Superfast HOG
SSD Mobilenet
Algorithms
RFCN Resnet
SSD Inception
Accuracies of algorithms are also dependent on the hardware platform since a heavier algorithm has to skip a lot of frames to maintain real-time operation.
Workstation
Hardware Platforms
Portable Computers
Small-sized computers

Current Object Detector Evaluations

Evaluation

True Positives, False positives etc. are simply accumulated over all frames of the video. Therefore, a detector with higher mAP (mean average precision) gives better results.



Drawbacks

- No measure of how quickly objects move in the real-time video. The number of frames which can be skipped is therefore unknown.
- If an object is detected, tracking and detection can work together for upcoming frames. Thus, detecting the object in any one viewpoint becomes important.
- The number of objects is not included in evaluation, which intuitively shouldn't be the case. A frame-wise evaluation doesn't give the right picture.

Suggested Evaluation Method

Count the number of objects detected (irrespective of which video frame), instead of enforcing object detection in every frame.

Estimating the processing requirement : Entropy

If an object stays in the video for a long time, detection is easier since any one of the frames can be used for detection.

Evaluation in device independent way : Infinite Resource Setting (IRS)

Irrespective of how many resources are put in, some algorithms can only do so much! In a device independent way, we identify the algorithms which would be able to provide the minimum application requirement.

Evaluation in resource-constrained setting : Resource Constrained Setting (RCS)

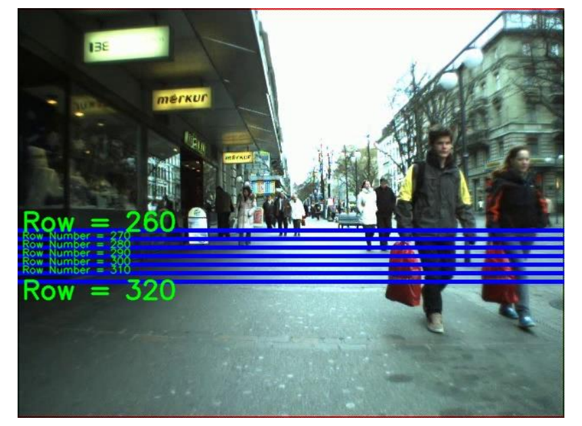
A slower running detector which has a higher accuracy has to skip frames in order to maintain real-time operation. When too many frames have to be skipped to keep up with the real-time performance, a lot of pedestrians go undetected resulting in a poor performance on the application level.

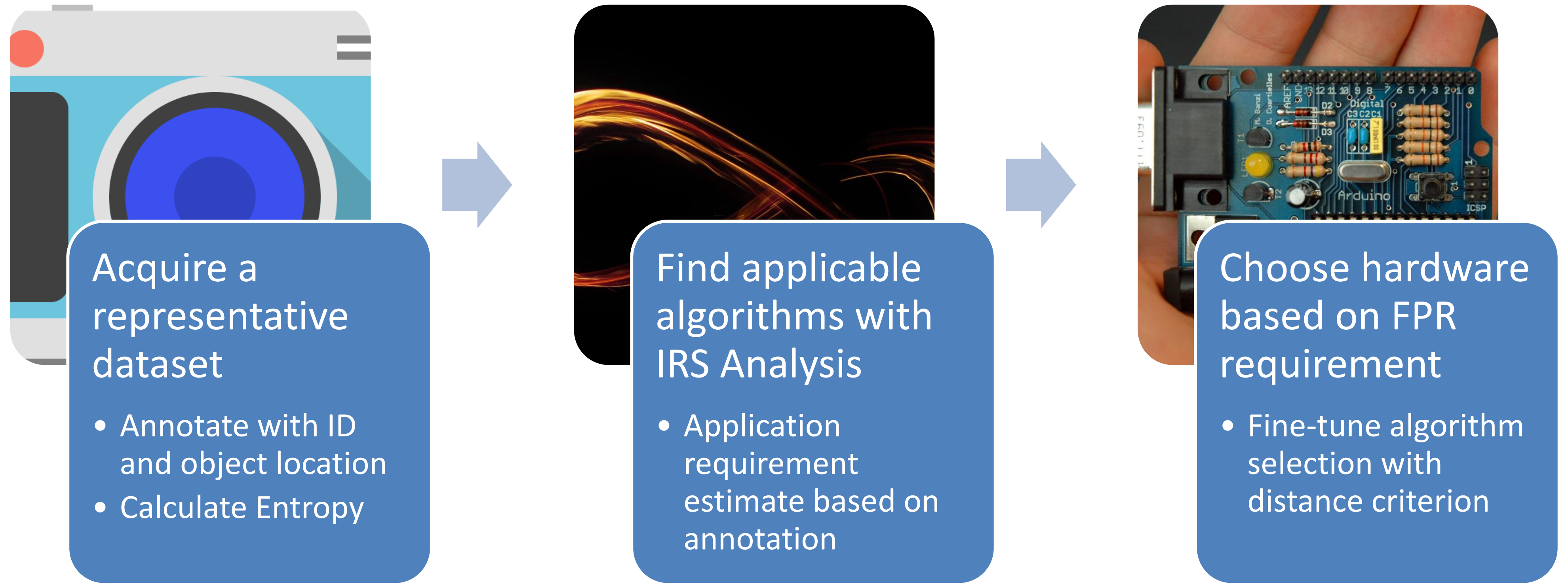
A fast running detector with lower accuracy is able to run detections on all frames but unable to detect all pedestrians in the frame. Nevertheless, it may still outperform a slower detector.

Different resources allow a different frame processing rate (FPR) for a specific algorithm. A slower FPR leads to skipping a lot of frames to maintain real-time operation.

Looking Ahead : The earlier the detection, the better the system

Even if a slower detector detects all the objects, a faster detector may detect the objects from sufficient distance. Applications like autonomous driving, assistive devices benefit from detecting objects from a larger distance. A pixel distance approximation is used as a special case of no vertical motion recording.





Acquire a representative dataset

- Annotate with ID and object location
- Calculate Entropy

Find applicable algorithms with IRS Analysis

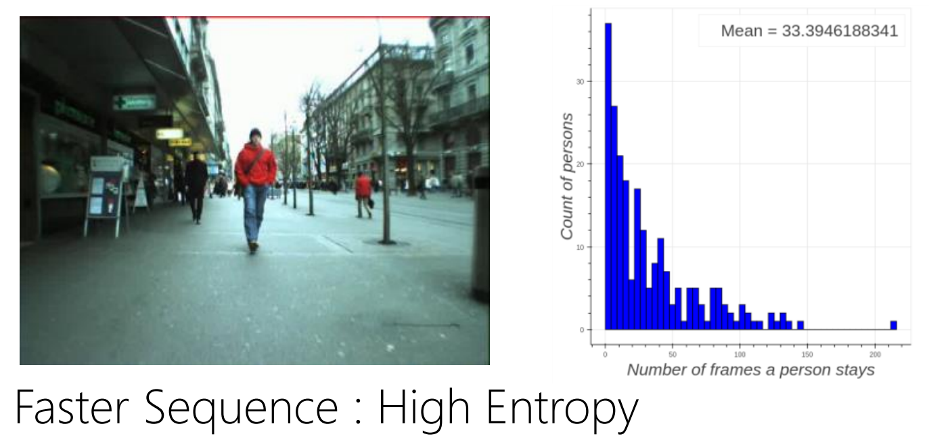
- Application requirement estimate based on annotation

Choose hardware based on FPR requirement

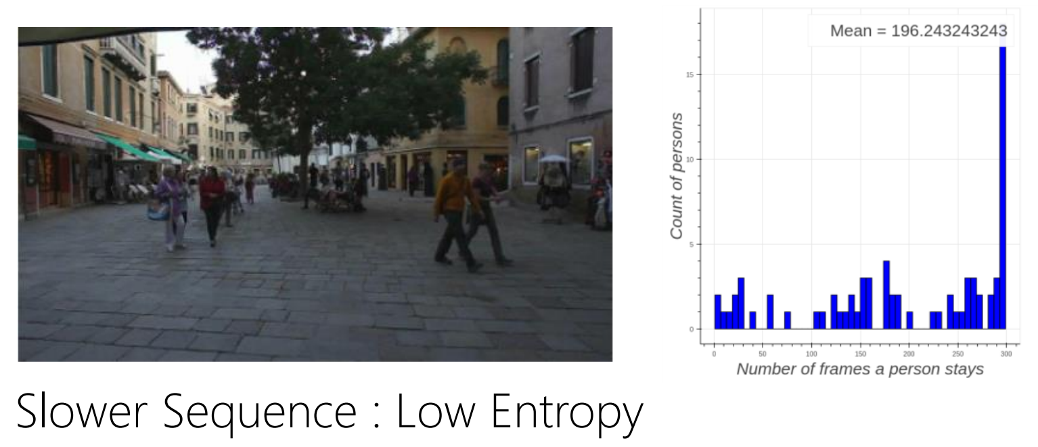
- Fine-tune algorithm selection with distance criterion

Results

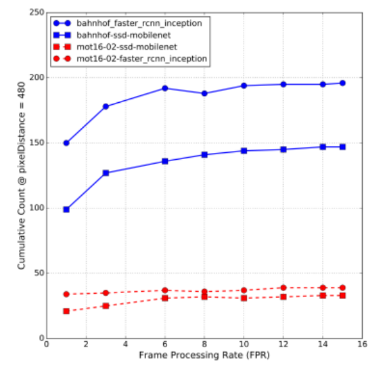
The effect of Entropy : how quickly will I need to process?



Faster Sequence : High Entropy
Objects are present in the video for a short time

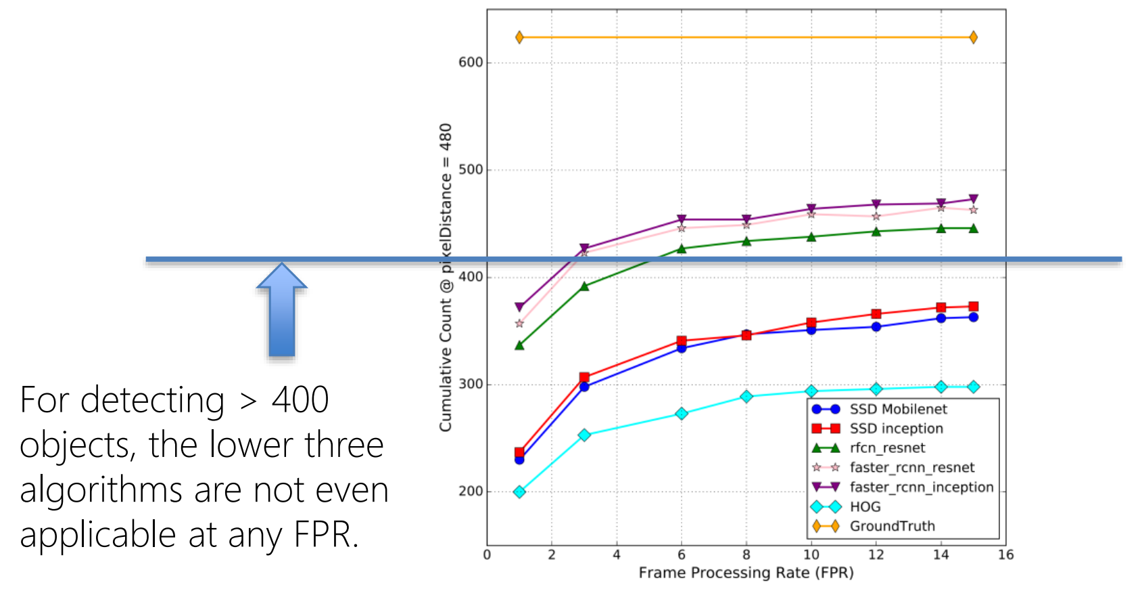


Slower Sequence : Low Entropy
Objects are present in the video for a long time



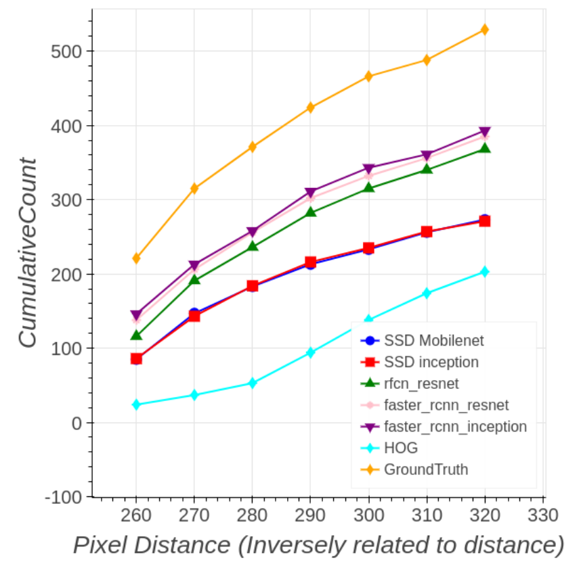
In a low entropy video, accuracy is maintained even at a low FPR.

Infinite Resource Setting (IRS) : what would even work?

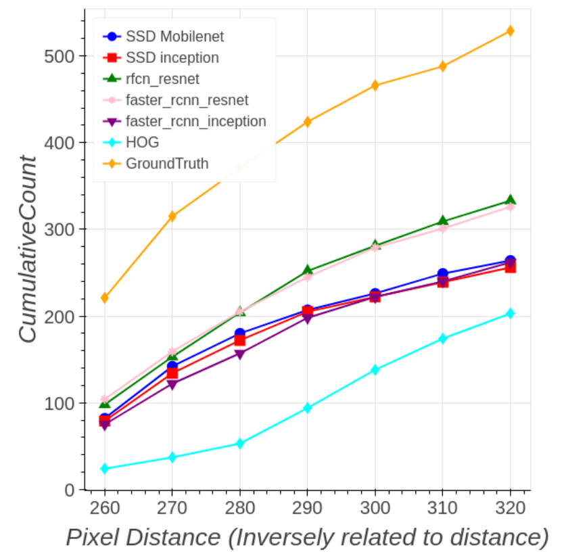


For detecting > 400 objects, the lower three algorithms are not even applicable at any FPR.

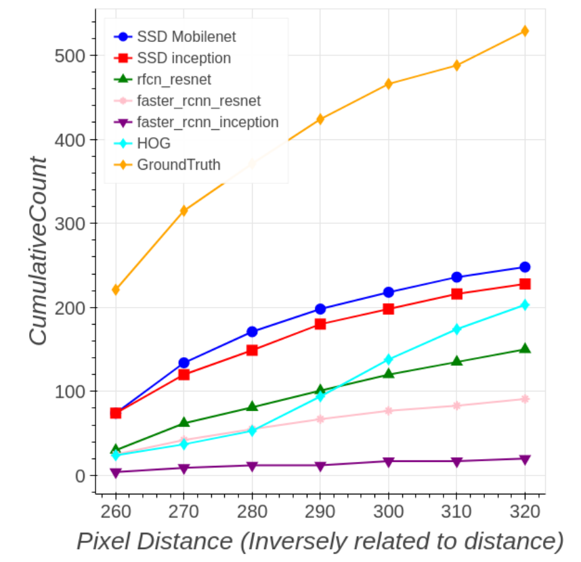
In the IRS setting, the number of objects detected at all distances follow the order of mean average precision (mAP) of the algorithms.
Therefore, if all frames can be processed, a more accurate algorithm always does better, given infinite resources.



Resource Constrained Setting (RCS) : Which hardware to pick? Which algorithm to run?



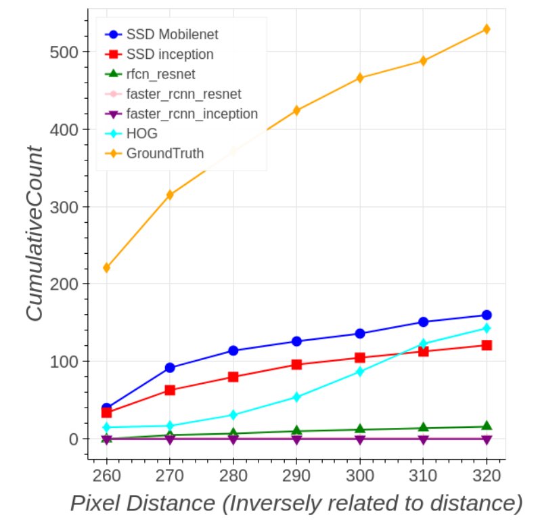
CPU + GPU



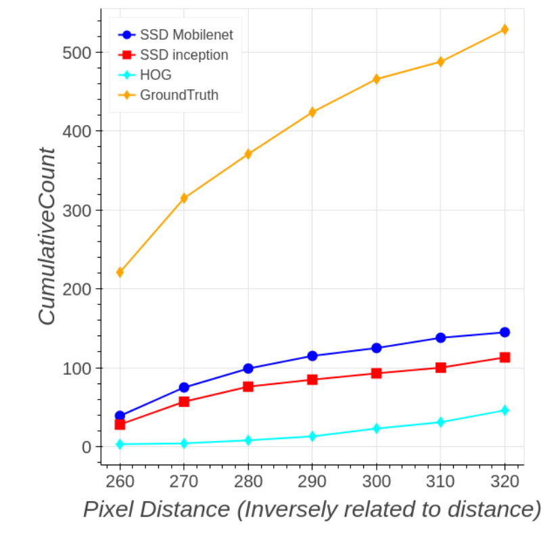
High-end CPU

There are three key observations here:

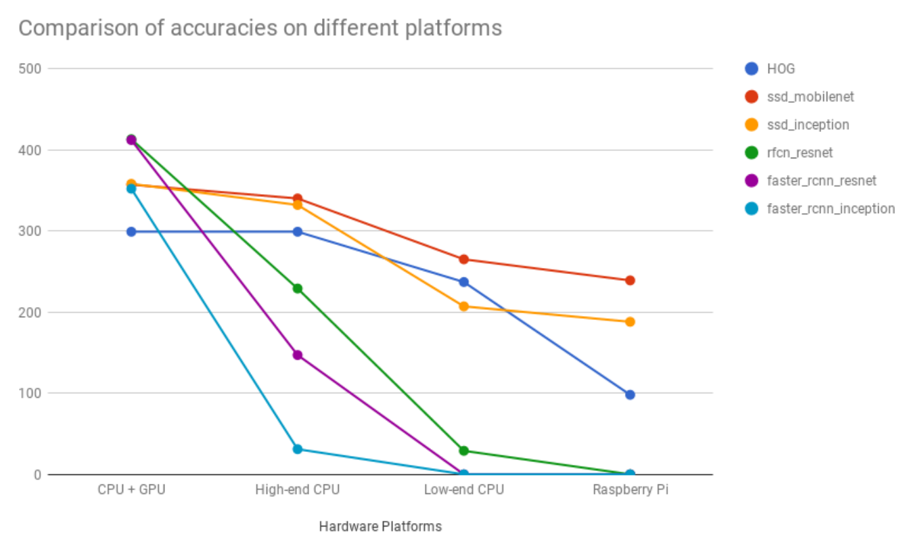
- Even in the most powerful configuration we have, results are different from the IRS Setting. Both rfcn_resnet and faster_rcnn_resnet algorithms perform better than faster_rcnn_inception.
- The difference is more evident in smaller platforms, where algorithms with 21 mAP are able to do better than algorithms with 34 mAP.
- The distance crossover in case of High-end CPU and Low-end CPU shows that depending on the distance at which you want to detect the objects, the optimal algorithm may vary.



Low-end CPU



Portable CPU



Acknowledgements: We would like to thank Rajesh Kedia for discussions which were helpful in materializing some of these ideas. Anupam Sobti has been supported by Visvesvaraya PhD Fellowship and travel is supported by Microsoft Research Travel Grant. Chetan Arora has been supported by Visvesvaraya Young Faculty Research Fellowship and Infosys Center for Artificial Intelligence.