CATEGORY: COMPUTER VISION & MACHINE VISION - CV04
POSTER P5141
CONTACT NAME Carlos S. Bederian: bc@famaf.unc.edu.ar
GPU TECHNOLOGY CONFERENCE

UNC Universidad Nacional de Córdoba

Real-time FullHD Tracking-Learning-Detection on a 2-SMX GPU

Jorge Atala[1], Carlos Bederián[2], Andrés Bordese[2], Facundo Gaich[2], Gastón Ingaramo[2], Julia Medina[2], Maximiliano Rossetti[1], Jorge Sánchez[2], Matías Tealdi[2] and Nicolás Wolovick[2]

[1]INVAP S.E., Argentina. [2]FaMAF, Universidad Nacional de Córdoba, Argentina.

Introduction

Object tracking is an important problem in computer vision and motion analysis. It can be defined as the estimation of the observed location (and scale) of a given object as it moves relative to the camera. This is a very challenging problem since one has to deal with changes in the object appearance, presence of background clutter, out-of-camera motions, etc.

TLD Overview

The TLD algorithm proposed by Kalal et al. [2] relies on the interplay between a reliable and fast appearance-based tracker [1] and a robust but slow multi-stage (cascaded) detector. It has shown great tracking performance in a variety of scenarios, although so far its application has been restricted to low-resolution imagery due to its complexity.

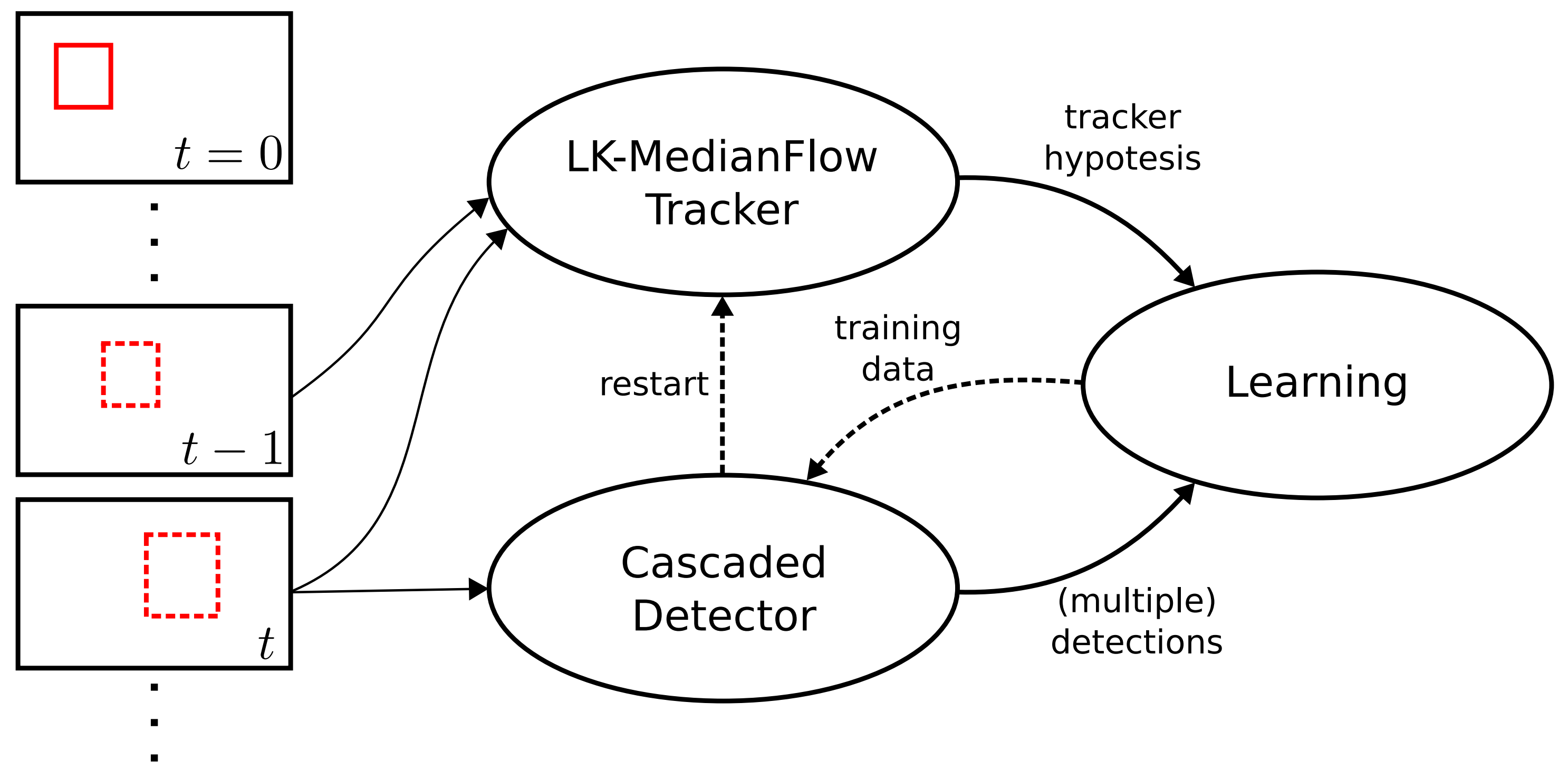
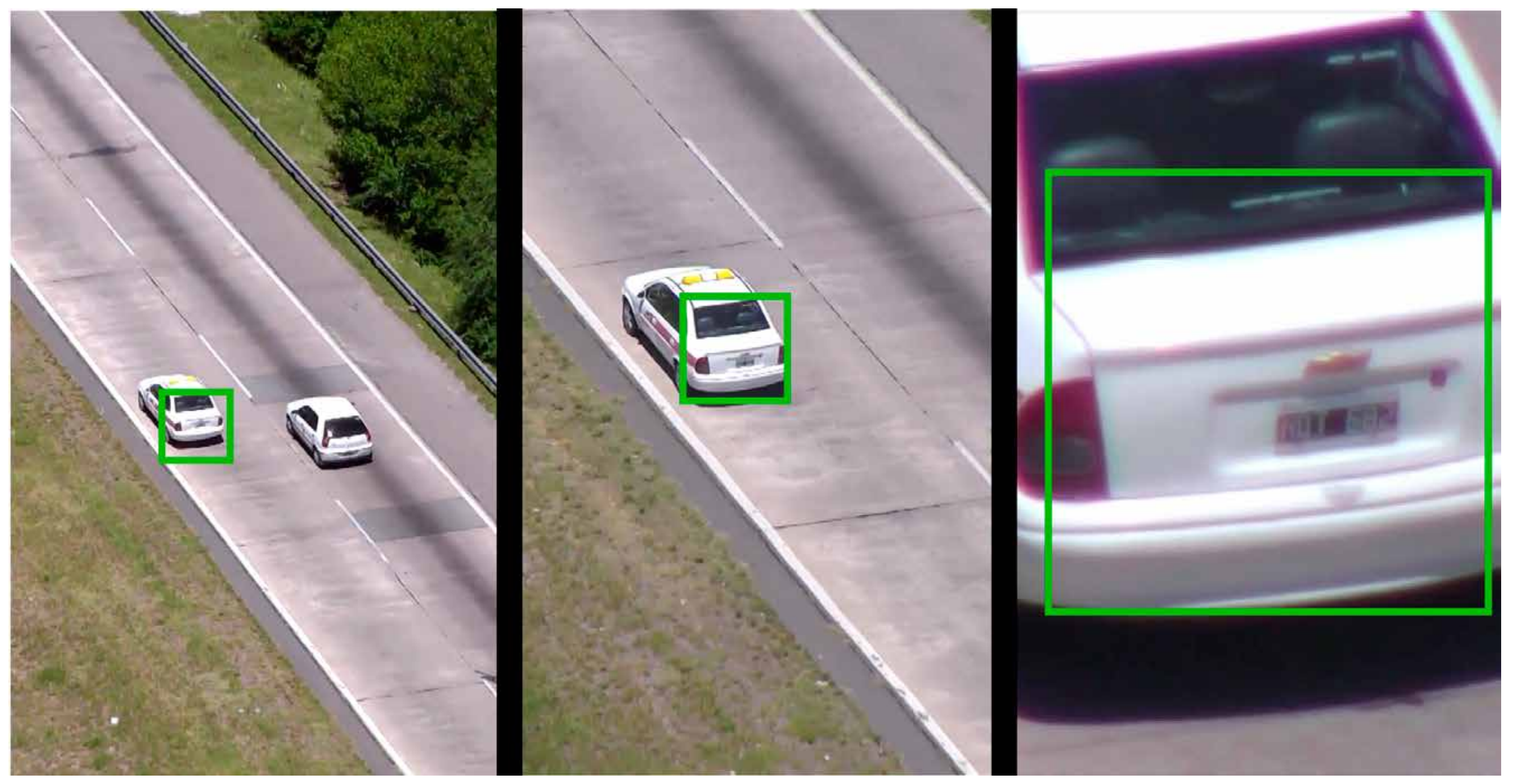


Fig. 1: An overview of the TLD algorithm.

Fig. 1 schematically depicts the main blocks of the algorithm.

- *LK-MedianFlow Tracker* [1]: Fast estimation, but not robust against out-of-camera motions and abrupt changes in visual appearance.
- *Cascaded Detector*: Slow but robust sliding window detector which corrects the tracker if necessary.
- *Learning*: Responsible of training data generation, detector error estimation and model updates.



Motivation

Our initial parallel and vectorized CPU implementation required downsampling the 1080p input video stream to a quarter of the resolution in order to perform above 30 frames per second. Other open-source TLD implementations performed similarly.
The parallelizability of a large portion of the TLD pipeline, combined with processor usage and power constraints on our target platform, prompted research into a CUDA port running on a mid-range GPU.

Approach

The first stage of our work consisted in offloading the detector module to the GPU. As the work showed promise, producing a heterogeneous CPU-GPU implementation that exceeded 30 frames per second without input downsampling, the rest of the TLD algorithm was then ported to CUDA in order to free up CPU resources for other processes running in the target platform.
Development was kickstarted through rapid prototyping using the Thrust template library. Thrust was then replaced wherever possible by the higher performing CUB library, while critical sections of the pipeline, obtained through profiling, were replaced by optimized kernels which target Kepler and newer GPU architectures.
The port aims to reproduce CPU implementation output in an exact fashion. While this limits optimization opportunities, it also improved porting productivity by ensuring correctness.

Results

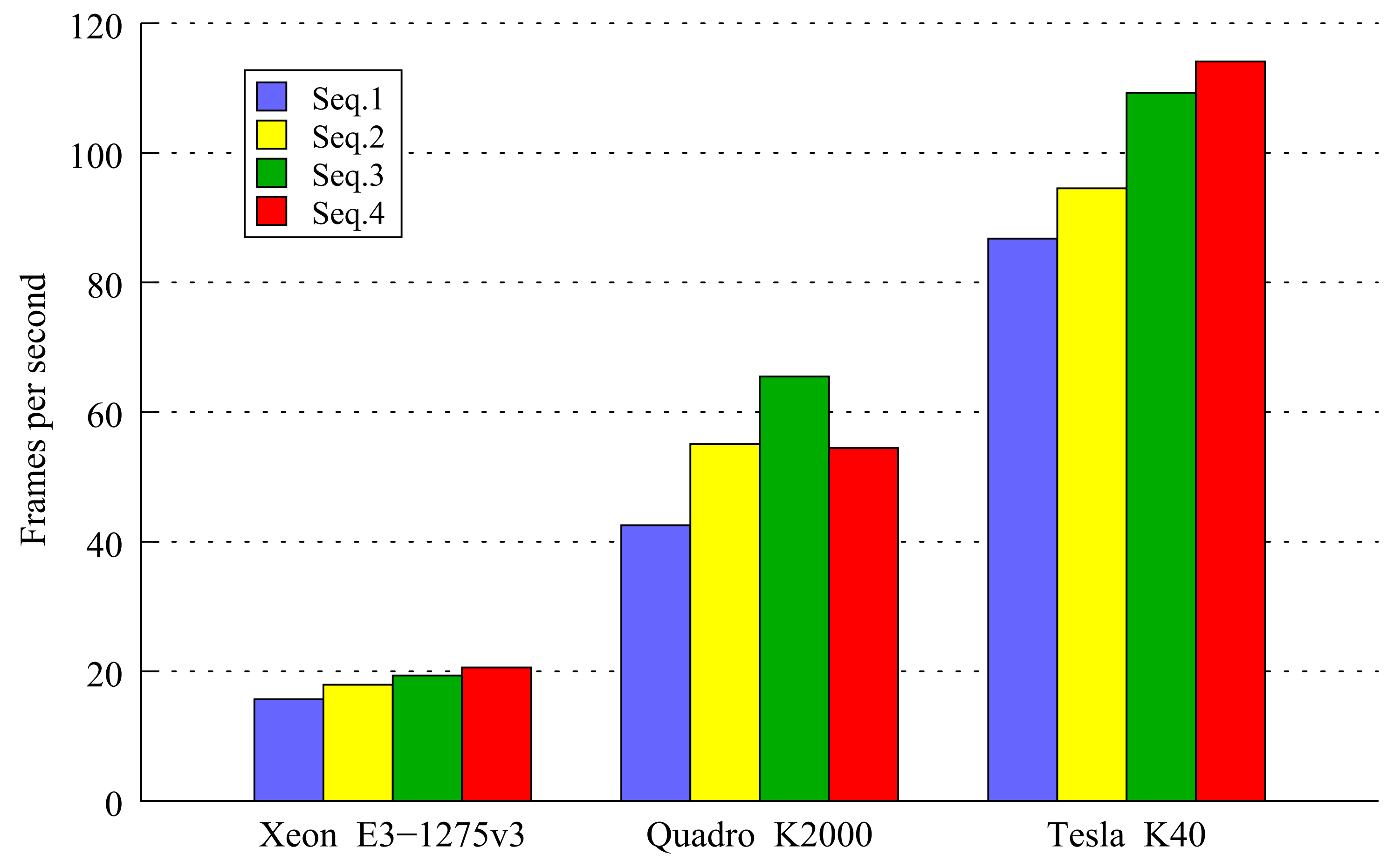


Fig. 2: Tracking performance on four 1080p video sequences.

Fig. 2 shows the performance of our TLD implementations on a set of 1080p video sequence samples with varying detection and tracking requirements. The tracker performs acceptably on the 384-core Quadro K2000 GPU of our target platform. Results on a high-end Tesla K40 GPU are also provided for comparison.

Future work

We aim to continue improving our implementation through more aggressive optimizations that drift from the original CPU code, in order to improve scaling on faster GPUs and also to be able to run TLD on embedded systems based on Tegra K1 or X1 SoCs.

References

[1] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *20th International Conference on Pattern Recognition, ICPR 2010, Istanbul, Turkey, 23-26 August 2010*, pages 2756–2759, 2010.

[2] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1409–1422, 2012.