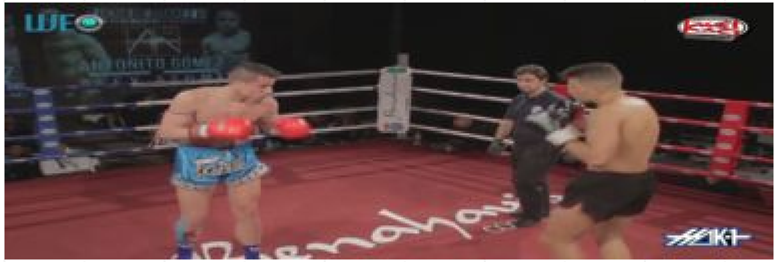
Deep Motion:
A Convolutional Neural Network for Frame Interpolation

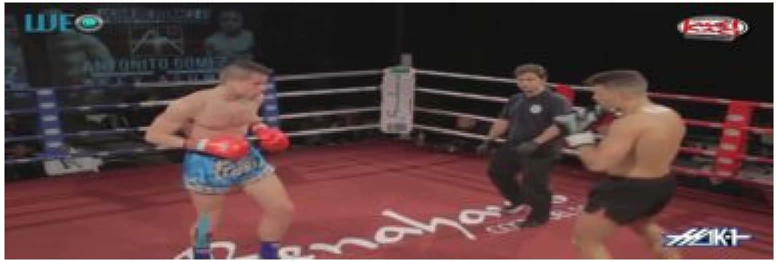
Basics of Frame Interpolation

Videos consist of frames. The number of frames displayed per second impacts video smoothness.

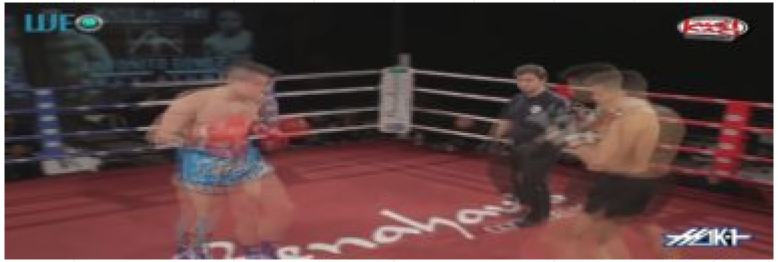
But how can we increase the FPS without speeding up the video?

Basics of Frame Interpolation





We can take two consecutive frames and try to form a middle frame.



Doing this for every frame tuple in a video, we can double the FPS.

Applications of Frame Interpolation

Conventional 60Hz
LG TruMotion 240Hz
60Hz
240Hz
LG's TruMotion 240Hz technology provides unsurpassed motion detail even in fast moving sports and movie scenes.
TruMotion
240Hz

TVs often try to use interpolation to reduce display motion blur.

Applications of Frame Interpolation

Doing stacked interpolations can generate pseudo-slow-motion video

(although the interpolation error can also stack)

Resources

- Python 2.7
  - Quick and easy prototyping, industry standard for Deep Learning
- Keras - Python Deep Learning Framework
  - Minimalistic and highly modular, which can run on top of Google’s TensorFlow
- FFMPEG
  - Tool for fast frame extraction from video
- Several Other Python Libraries
  - NumPy / SciPy - Very useful for mathematical and scientific computing
  - Matplotlib - For plotting graphs, viewing images

Data

KITTI

- Over 80,000 frames of driving footage
- Used to train self-driving cars, but benefits us by having good uniform motion

YouTube 8M

- Newly released library of 8 million YouTube videos classified and labelled
- We use just the raw video data for fine-tuning

Data Collection & Preprocessing

For KITTI image sequences

- Resized all images to 384x128, to improve network training speed
- Separated into training and validation sets

For YouTube-8M

- Used a video scraper script to download 20,000 videos from the dataset, using their YouTube IDs
  - Requirements: Must be 720p resolution, with .mp4 file extension (for consistency)

Created batch generator functions, to extract batches of random frame triplets from either of the datasets

- Allows us to train on live batches of images, and do preprocessing on-the-fly

CNN Architecture

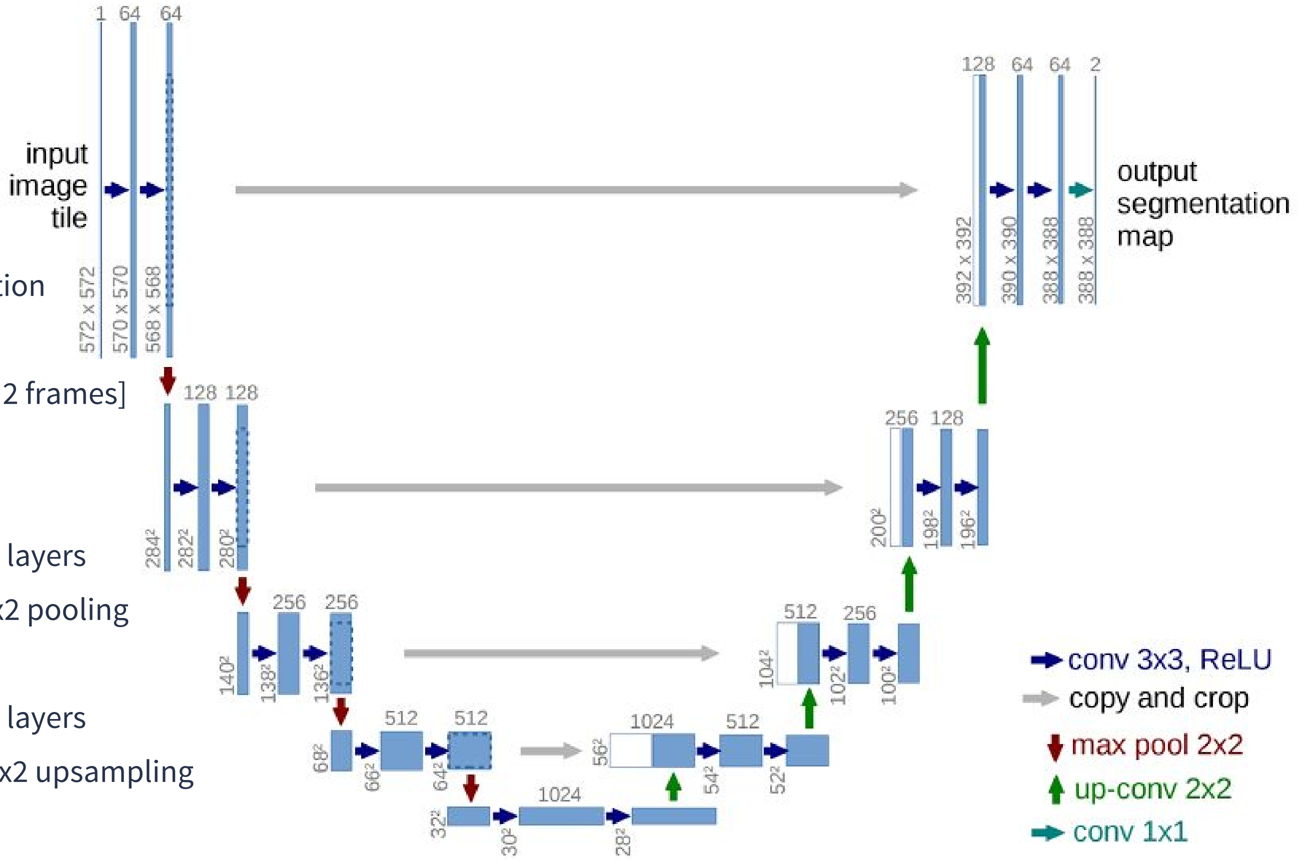
Modelled after U-Net (right), which is used for biomedical image segmentation

Input shape = (6, 128, 384) [6 because 2 frames]

Output shape = (3, 128, 384)

Five blocks of: Two 3x3 Convolutional layers of increasing filter size, followed by 2x2 pooling

Five blocks of: Two 3x3 Convolutional layers of decreasing filter size, followed by 2x2 upsampling



CNN Architecture

Utilizes Merge layers (gray arrows),

to concatenate output of previous layers,

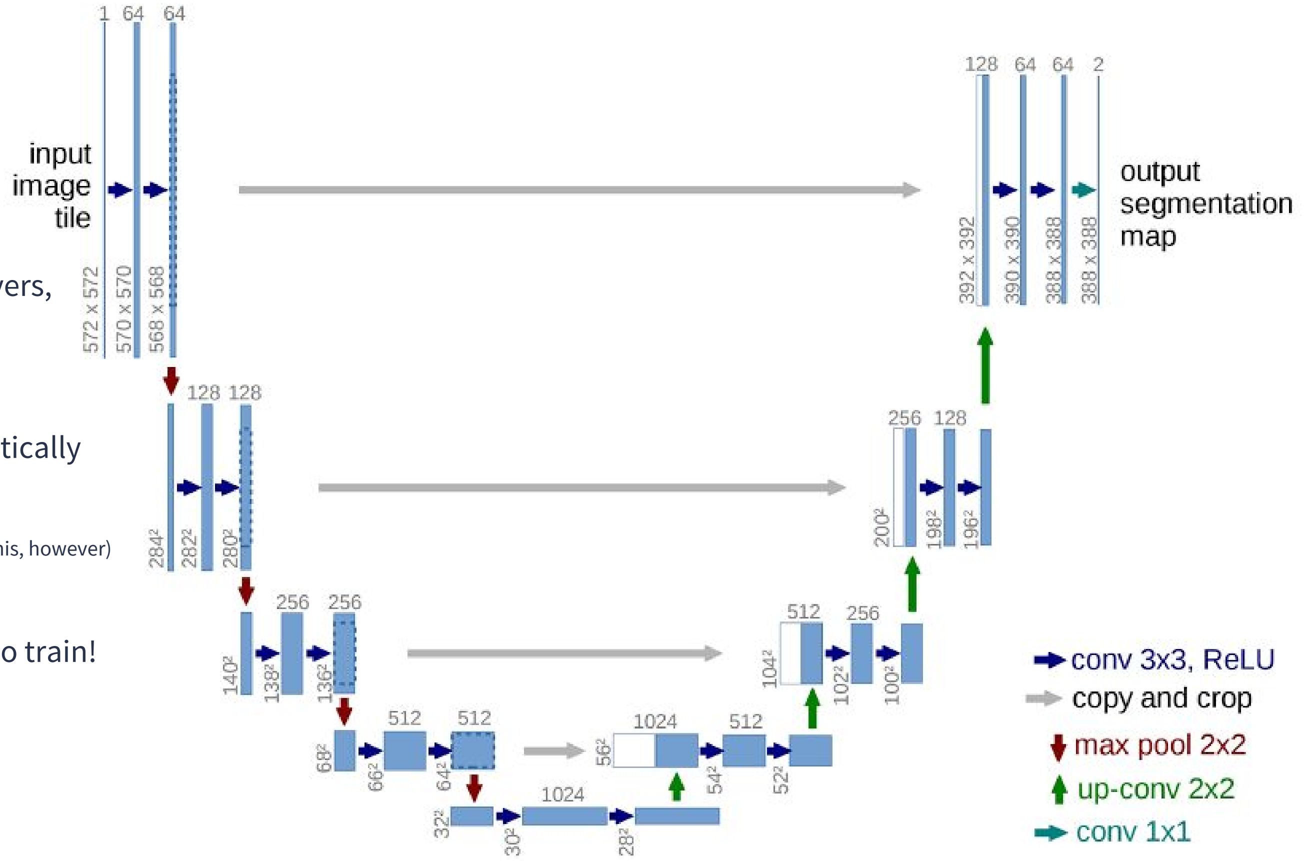
in order to preserve fine-details.

Fully Convolutional, so should theoretically

support images of any size

(our inclusion of a Batch Normalization layer prevents this, however)

Over 19 million network parameters to train!



Training Configuration

Loss Function: Charbonnier - $\rho(x) = \text{sqrt}(x^2 + \varepsilon^2)$, where we set $\varepsilon = 0.01$

- Similar to Mean Squared Error Loss, but depending on ε, it increases the loss when it should be very small.

Optimizer: Adam, default params

Learning Rate = 0.001, which is multiplied by 0.1 when training loss stagnates

Batch Size = 16

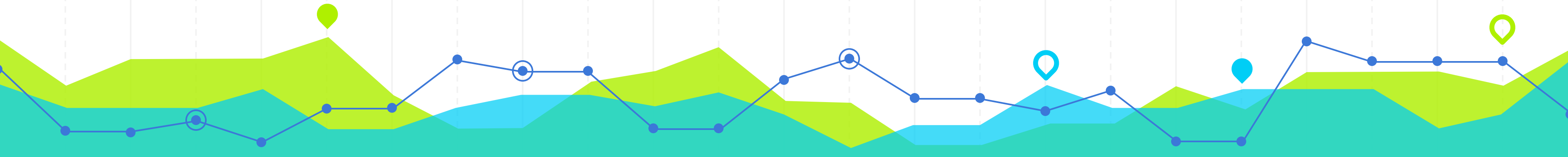
Training Methodology

We first trained over 153,600 samples in KITTI, which is almost 2 epochs

- Took roughly 10 hours on a NVidia 980 Ti GPU
- Model weights were saved only when validation loss increased (validation set evaluated every 800 samples)

Then we fine-tuned the model over roughly 80,000 samples in YouTube-8M

- Trained for about 24 hours. Takes much longer because of live FFMPEG frame extraction from .mp4's
- Model weights were saved when training loss increased
    - Since there is no validation set (effective dataset is so large, ~0% chance of repeated samples)



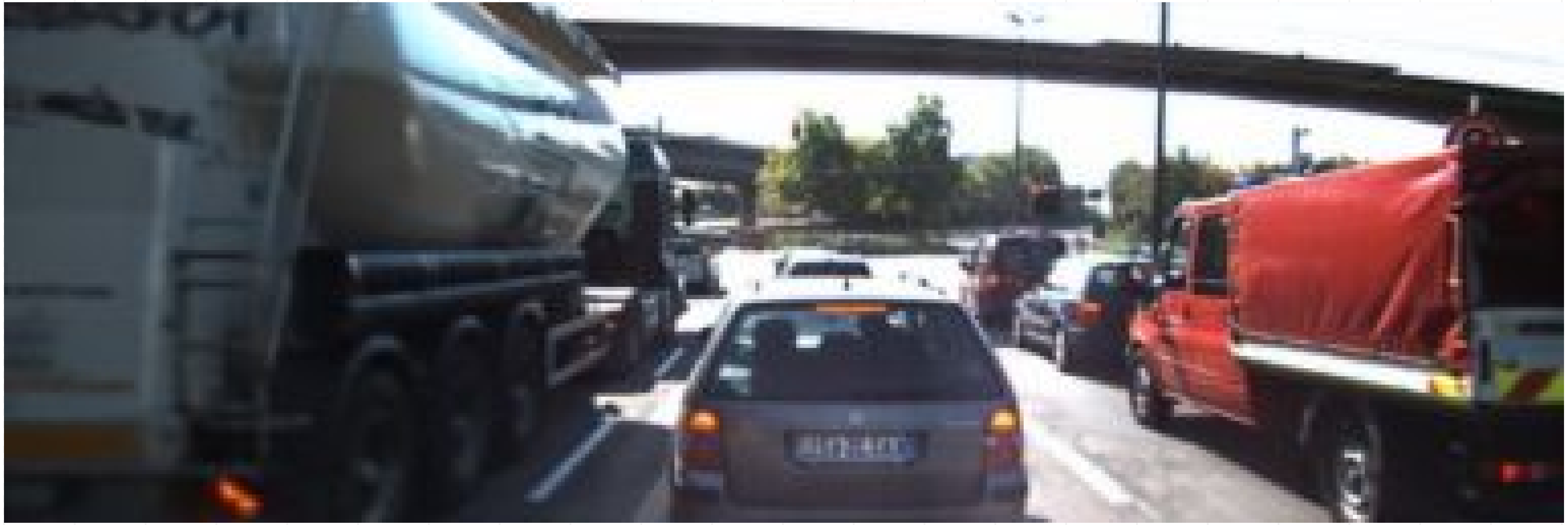
Results: KITTI - 1 (First Frame)

Results: KITTI - 1 (Middle Frame)

Results: KITTI - 1 (Last Frame)

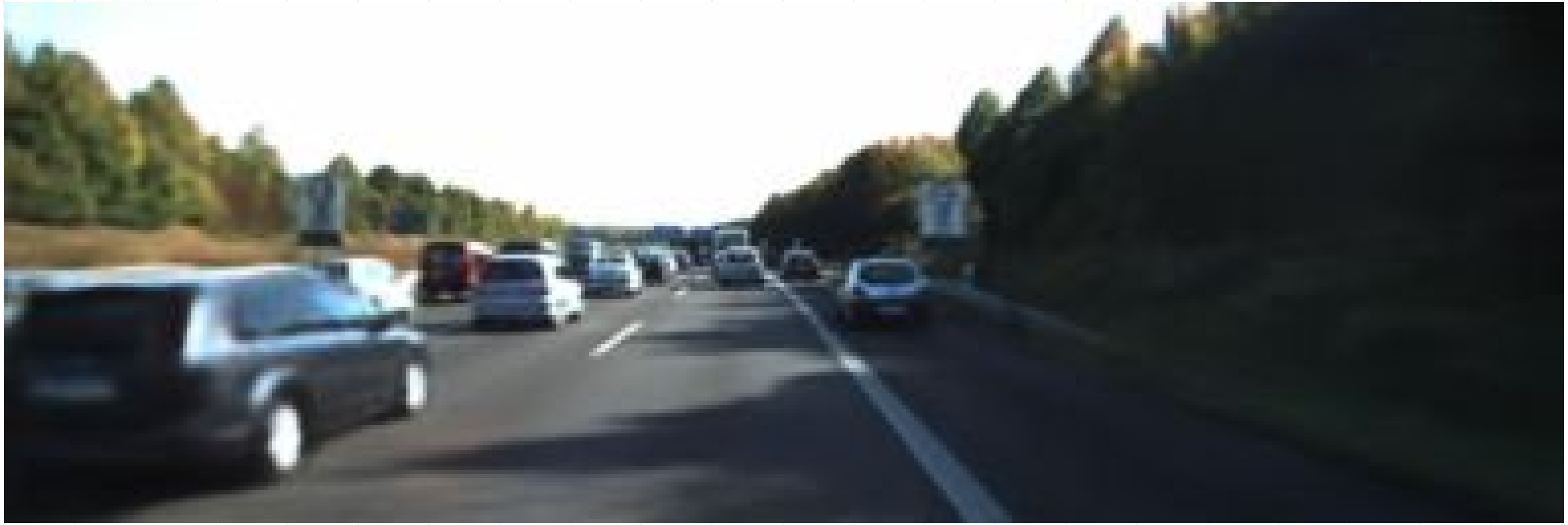
Results: KITTI - 1 (Blended)

Results: KITTI - 1 (Predicted Middle Frame)



Results: KITTI - 2 (Blended)

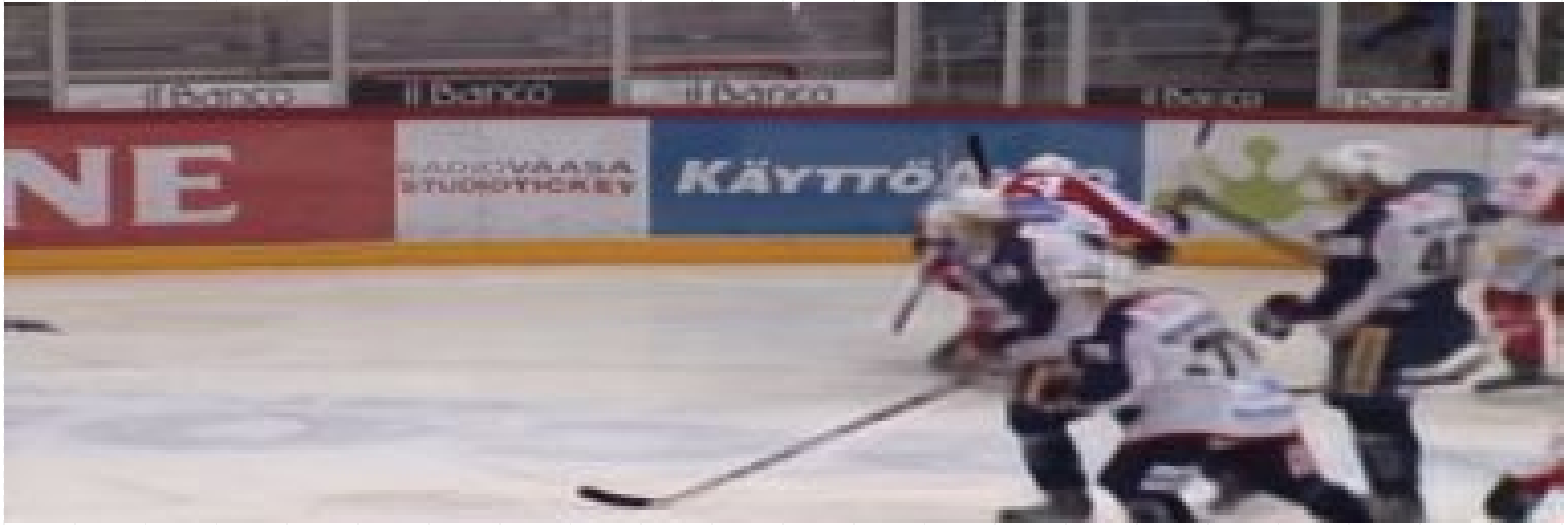
Results: KITTI - 2 (Predicted Middle Frame)



Results: KITTI - 2 (Middle Frame)

Results: YouTube-8M - 1 (Blended)

Results: YouTube-8M - 1 (Predicted Middle Frame)



Results: YouTube-8M - 1 (Middle Frame)

Results: YouTube-8M - 2 (Blended)

Results: YouTube-8M - 2 (Predicted Middle Frame | Pre-Fine-Tuning)

Results: YouTube-8M - 2 (Predicted Middle Frame | Post-Fine-Tuning)

Results: Video FPS Doubling Demo