Fast stixel computation for fast pedestrian detection

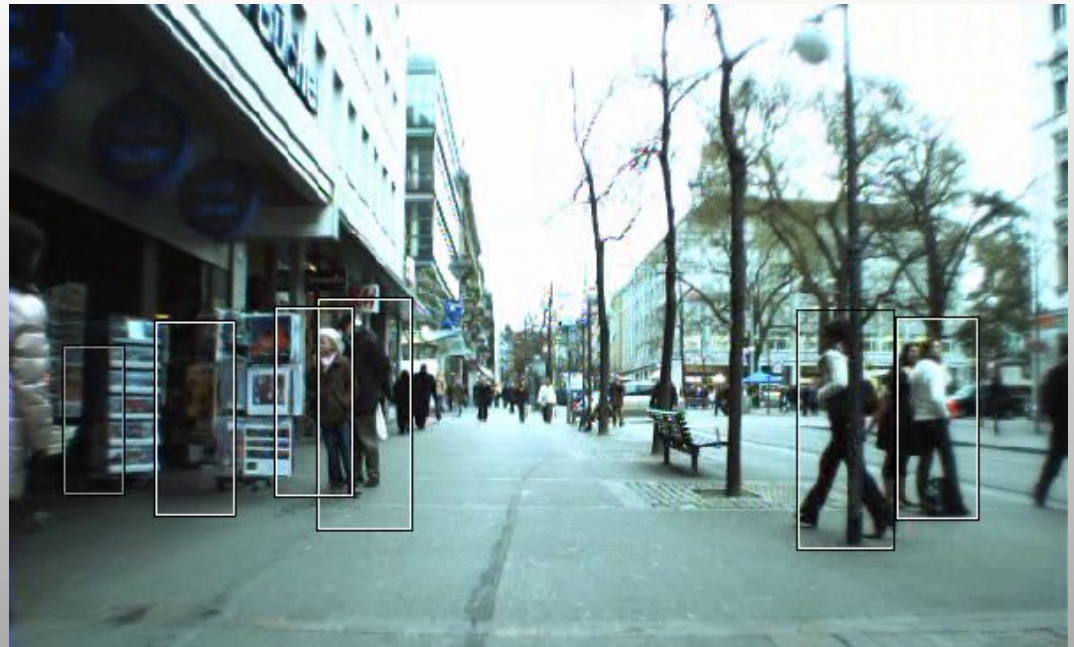
R. Benenson, M. Mathias, R. Timofte and L. Van Gool




KATHOLIEKE UNIVERSITEIT
LEUVEN

VISICS

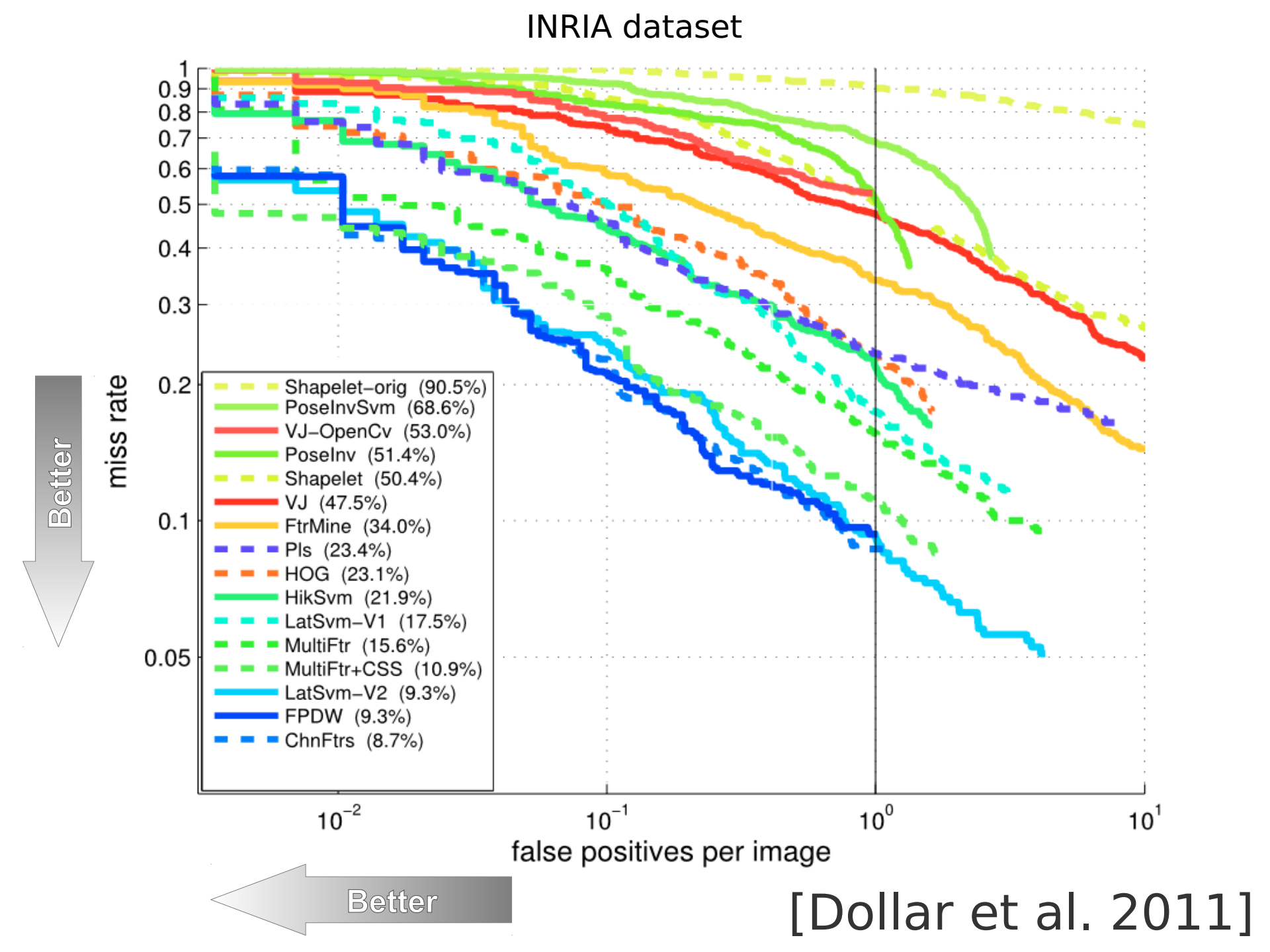
We want *very* fast pedestrian detections



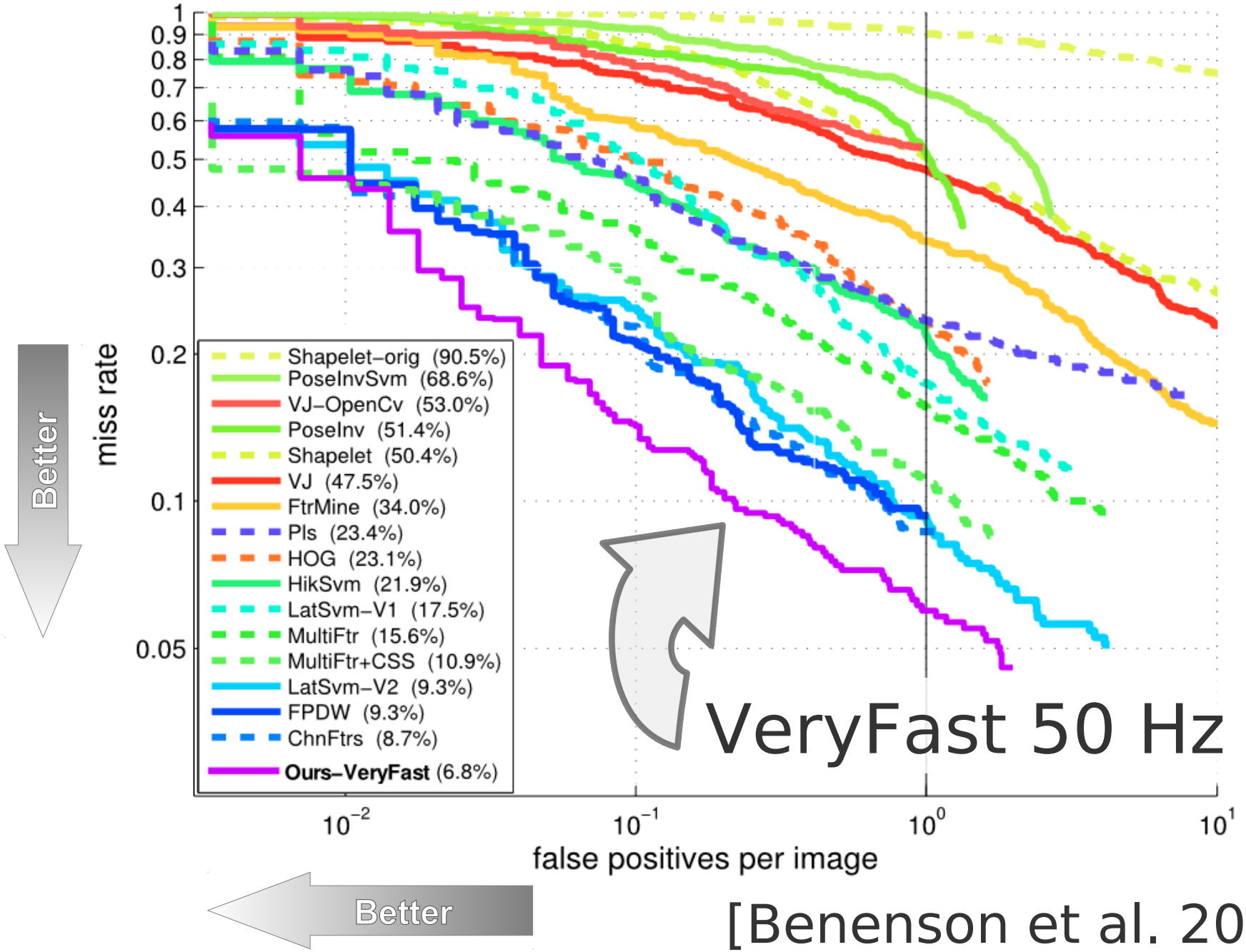
In this paper,
we reach *high quality*
pedestrian detection *at 165 Hz*

Why 165 Hz ?

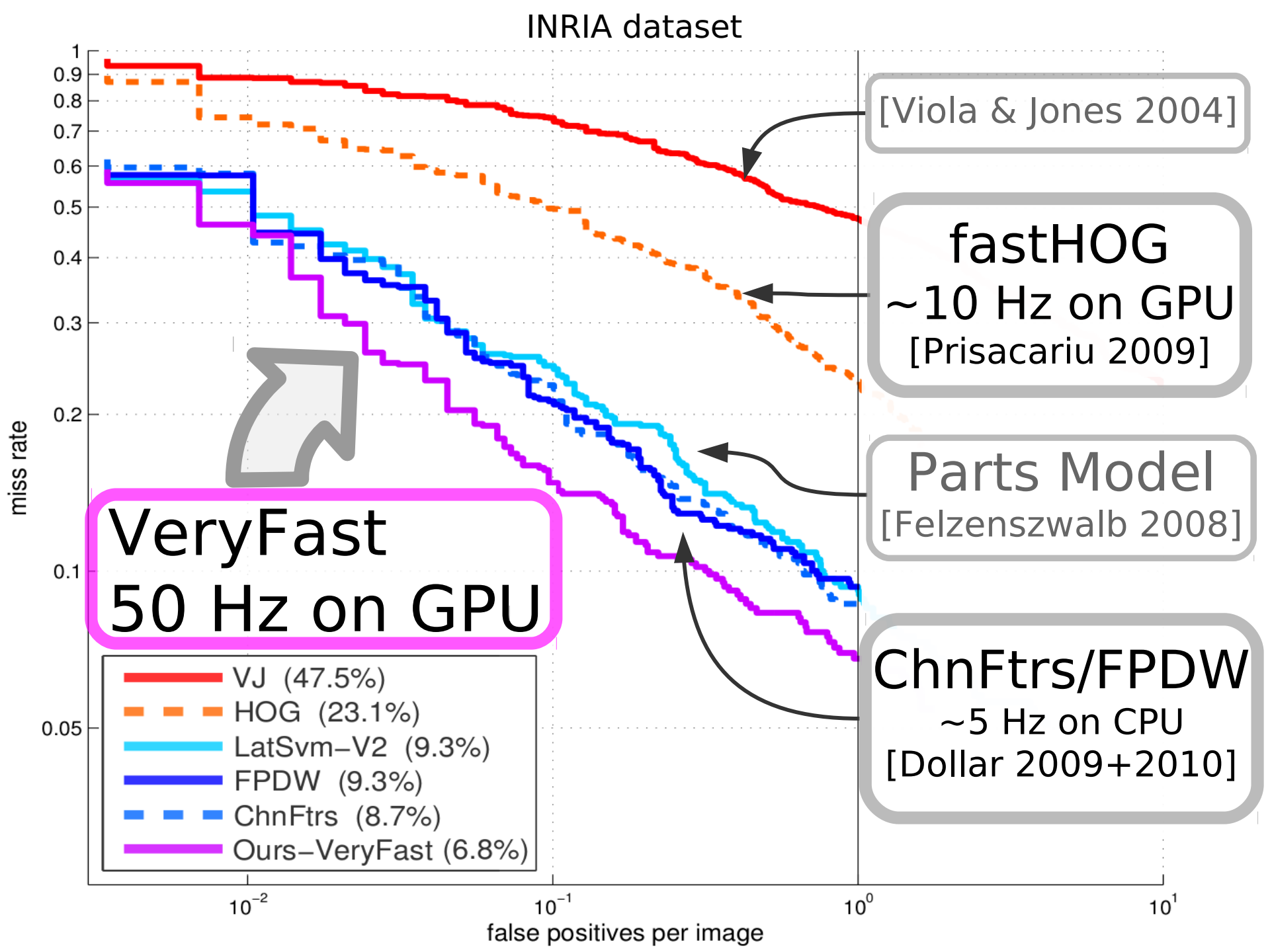
- Detection is one module amongst many
- Less computational power
- *Latency matters*


INRIA dataset
miss rate
false positives per image
Better
Better
Shapelet−orig (90.5%)
PoseInvSvm (68.6%)
VJ−OpenCv (53.0%)
PoseInv (51.4%)
Shapelet (50.4%)
VJ (47.5%)
FtrMine (34.0%)
Pls (23.4%)
HOG (23.1%)
HikSvm (21.9%)
LatSvm−V1 (17.5%)
MultiFtr (15.6%)
MultiFtr+CSS (10.9%)
LatSvm−V2 (9.3%)
FPDW (9.3%)
ChnFtrs (8.7%)
[Dollar et al. 2011]

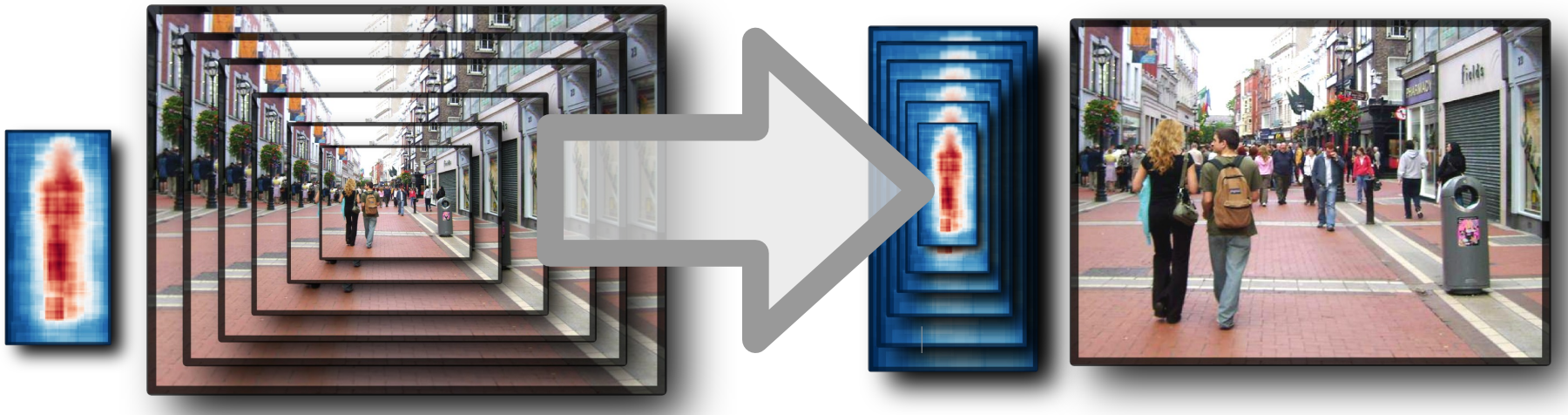
INRIA dataset



[Benenson et al. 2012]


INRIA dataset
miss rate
false positives per image
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.05
10^{-2}
10^{-1}
10^{0}
10^{1}
VJ (47.5%)
HOG (23.1%)
LatSvm−V2 (9.3%)
FPDW (9.3%)
ChnFtrs (8.7%)
Ours−VeryFast (6.8%)
VeryFast
50 Hz on GPU
[Viola & Jones 2004]
fastHOG
~10 Hz on GPU
[Prisacariu 2009]
Parts Model
[Felzenszwalb 2008]
ChnFtrs/FPDW
~5 Hz on CPU
[Dollar 2009+2010]

Detecting objects without image resizing



1 model,
50 image scales

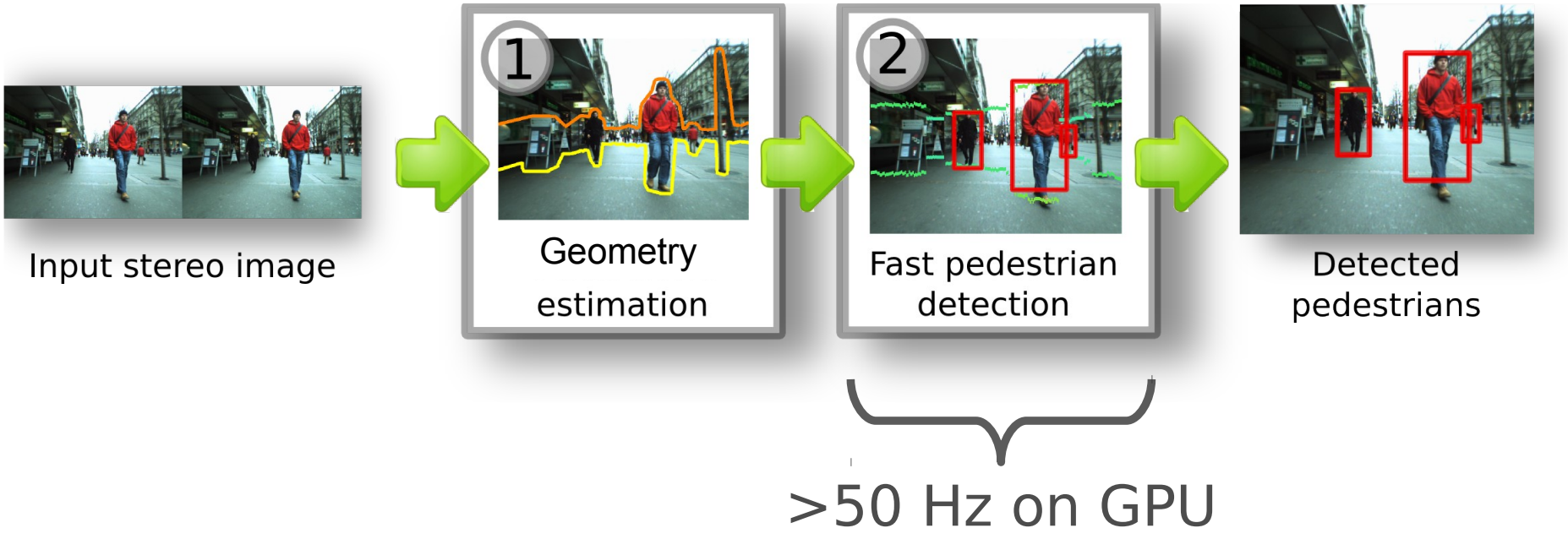
50 models,
1 image scale

[Benenson et al. 2012]

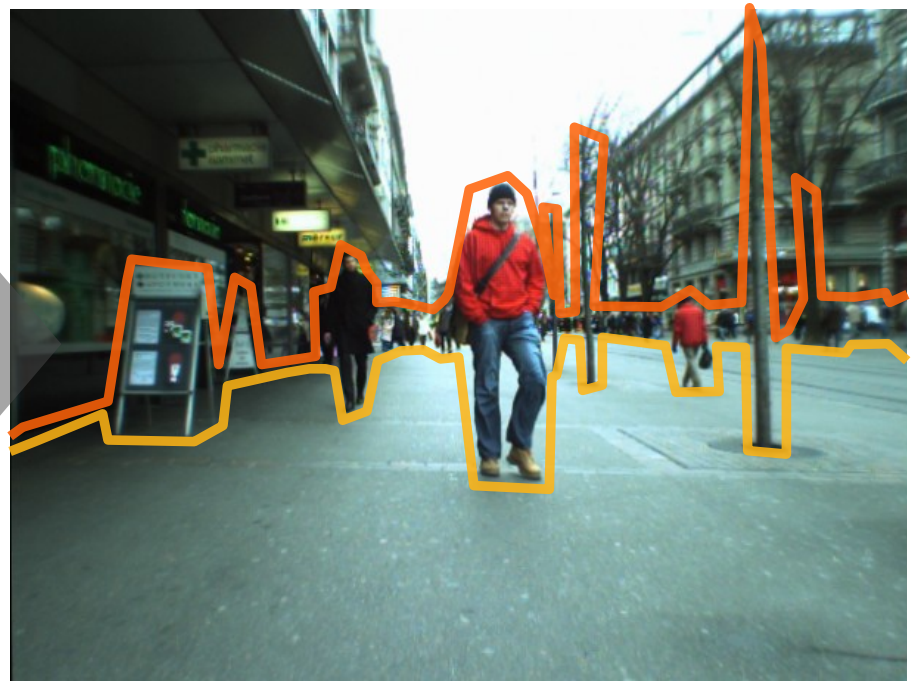
We want to use scene geometry to guide the detections

Monocular

Stereo


1
2
Input stereo image
Geometry estimation
Fast pedestrian detection
Detected pedestrians
>50 Hz on GPU

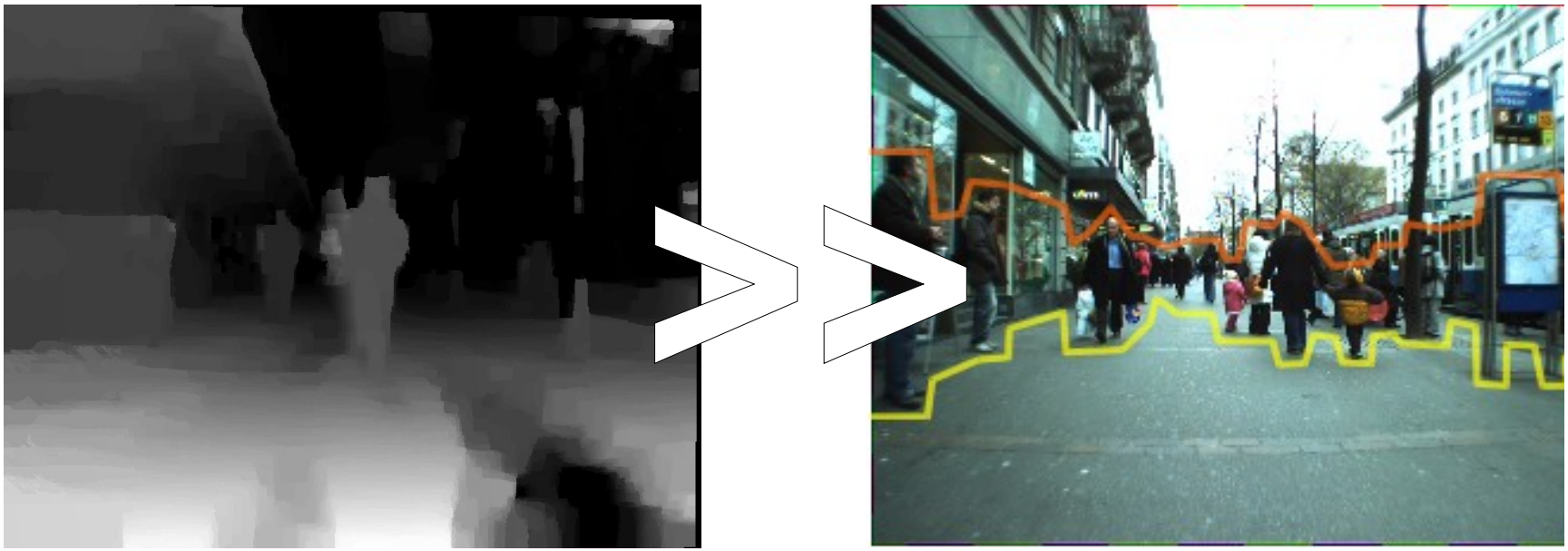
Depth maps are slow to compute



<50 Hz on CPU

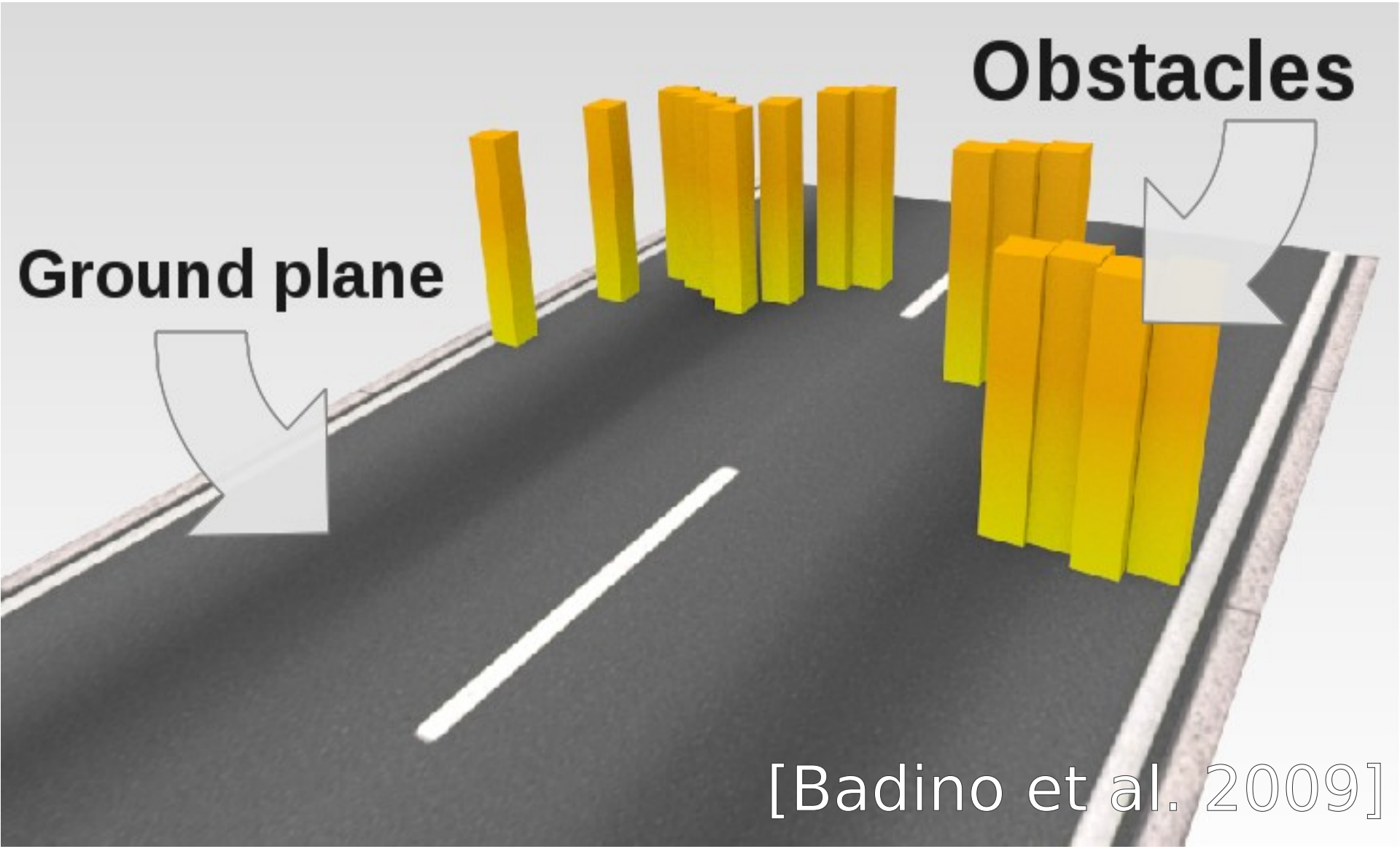
What to do when
depth map computation
is much slower than detection?

Depth maps compute more information that we need



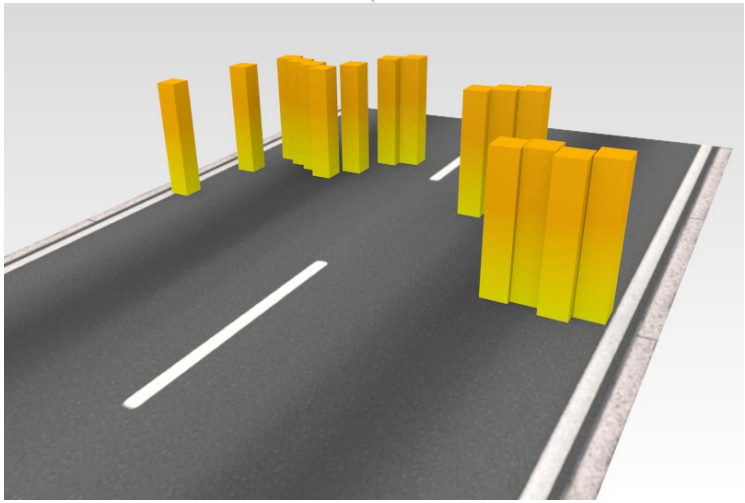
Key insight

Stixel world



Stixel world without depth map

90 Hz on CPU

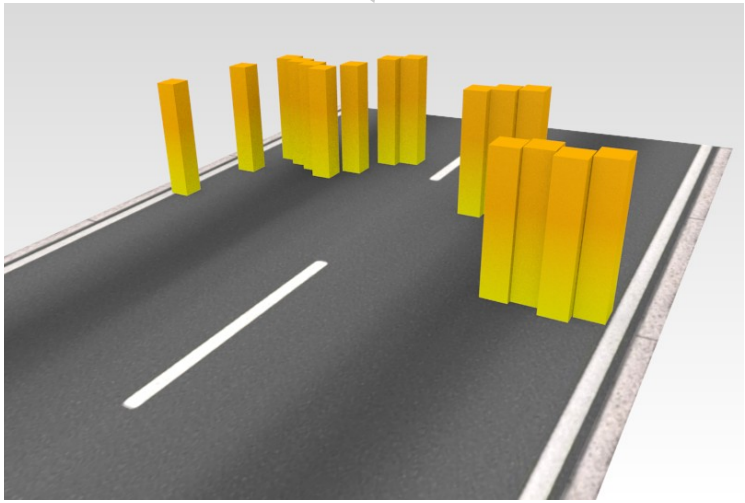


[Benenson et al. 2011]

Stixel world without depth map

135 Hz

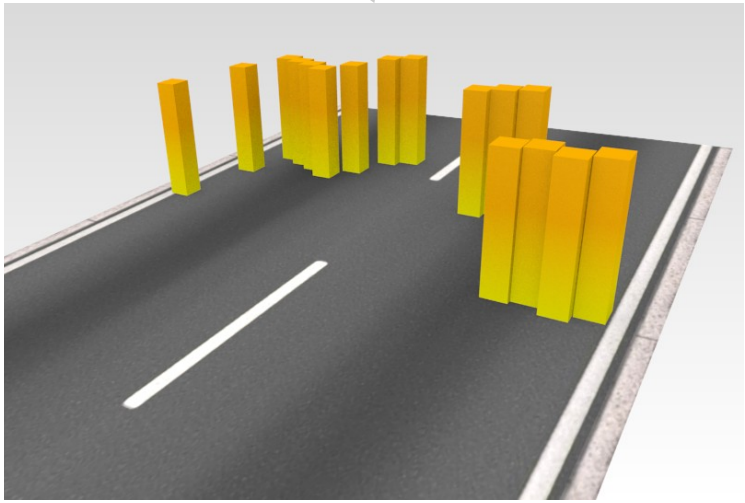
~~**90 Hz**~~ on CPU



[Benenson et al. 2011 & 2012]

Stixel world without depth map

300 Hz on CPU



CVVT@ICCV 2011

"we can extract scene geometry from stereo at 90 Hz"

CVPR 2012

"we can detect pedestrian at 50 Hz,
135 Hz when using scene geometry"

CVVT@ECCV 2012

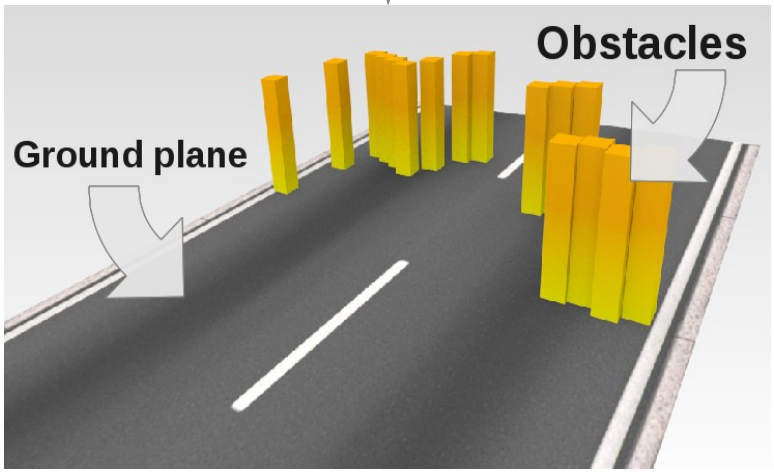
"we can extract **scene geometry from stereo at ~300 Hz**,
and thus detect pedestrian at 165 Hz"

Is (direct) stixel world a good idea?

It opens new possibilities

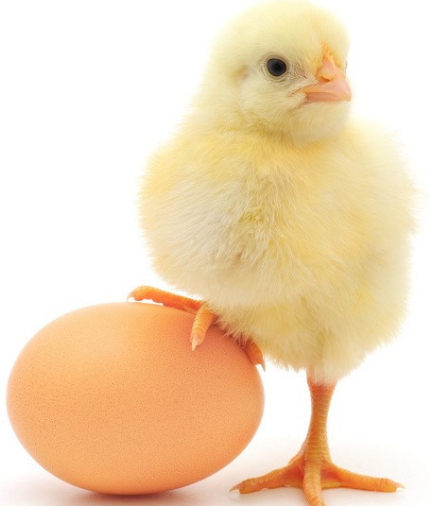
Monocular

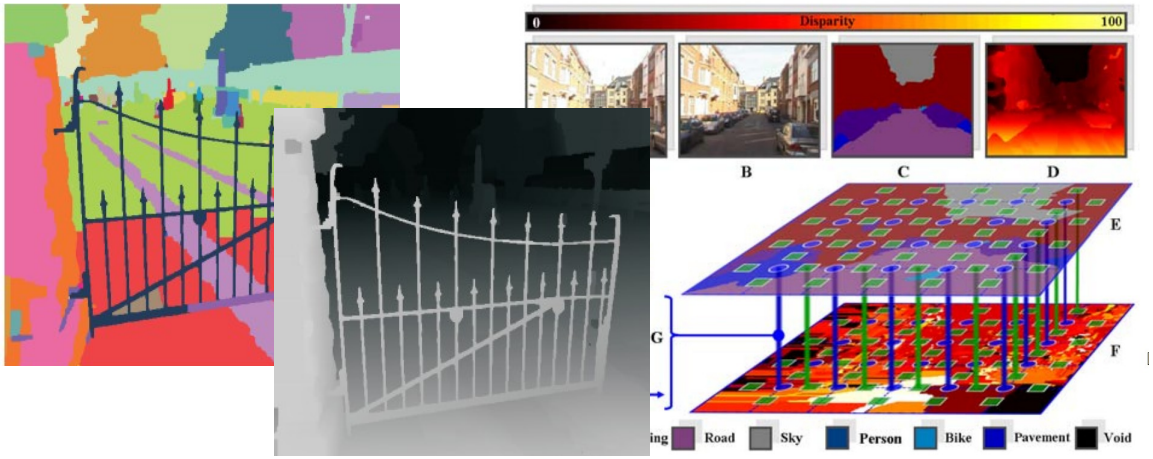
Pixel-wise depth map

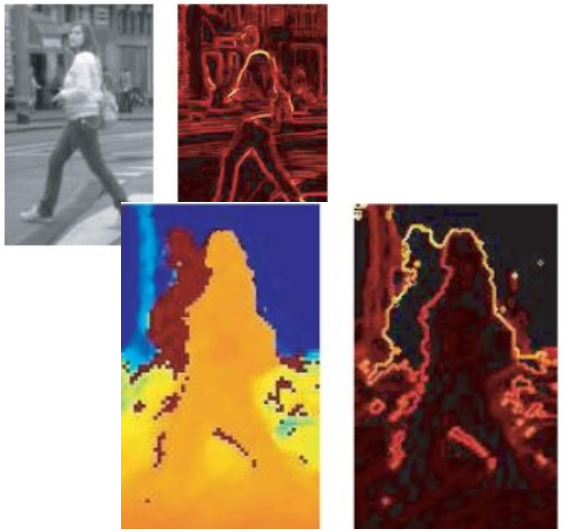


Stixel world

Stereo and object detection is a chicken and egg problem!







[Ladicky et al. 2010], [Bleyer et al. 2011]

Stixel world without depth map

Stixel world without depth map

1) Ground plane estimation

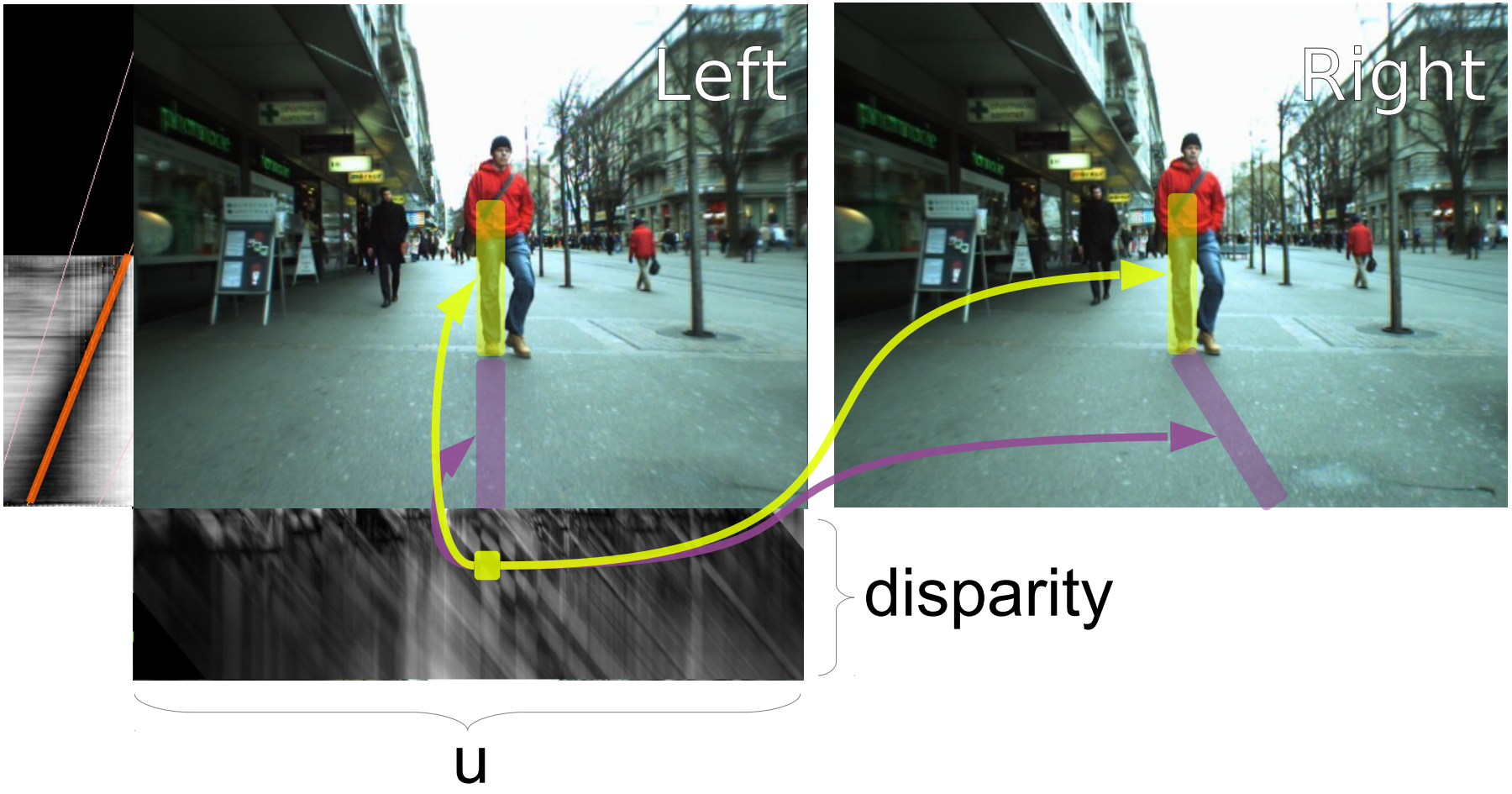
Stixel world without depth map

1) Ground plane estimation

Stixel world without depth map

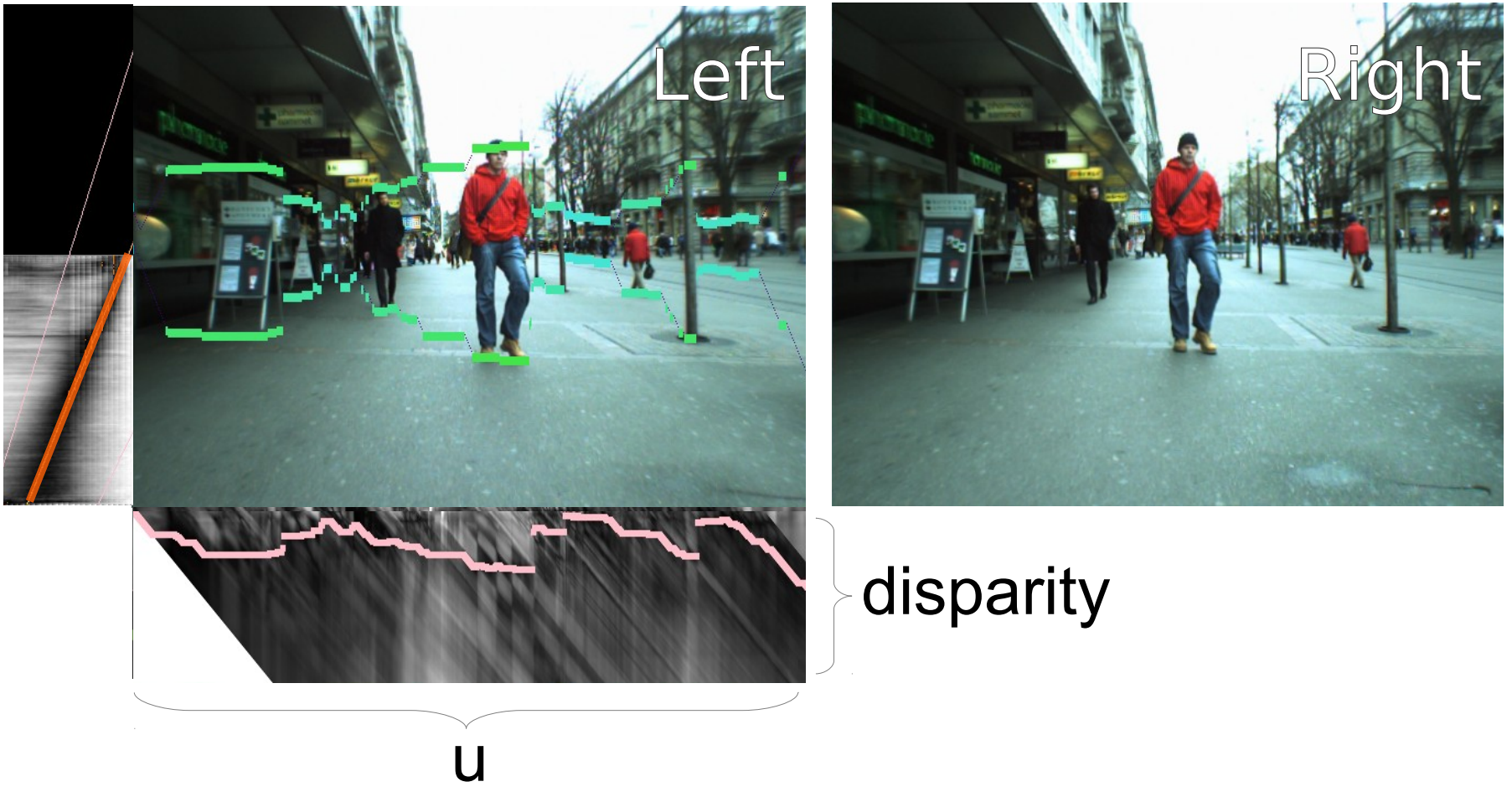
2) Stixel distance estimation

Stixel world without depth map



2) Stixel distance estimation

Stixel world without depth map



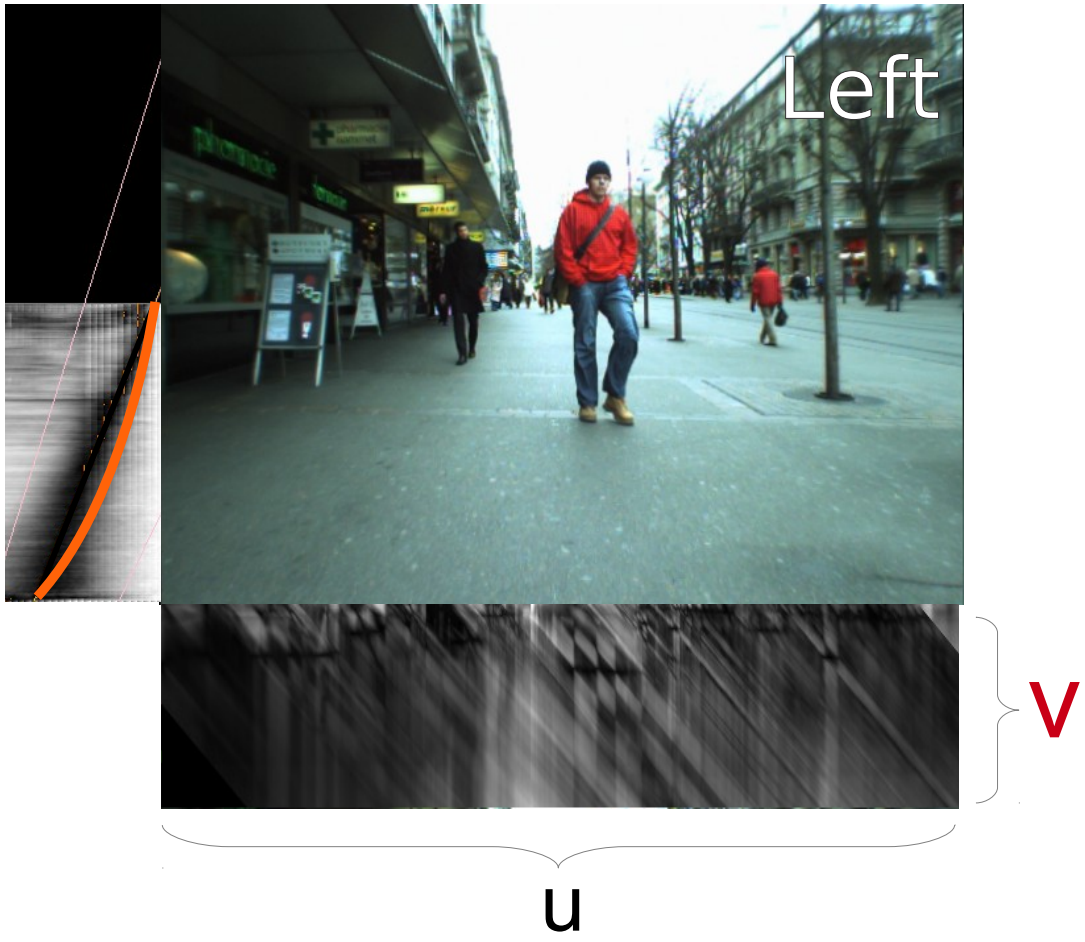
2) Stixel distance estimation @ **135 Hz CPU**

100 Hz is too slow!

Our initial proposal has (at least) two issues

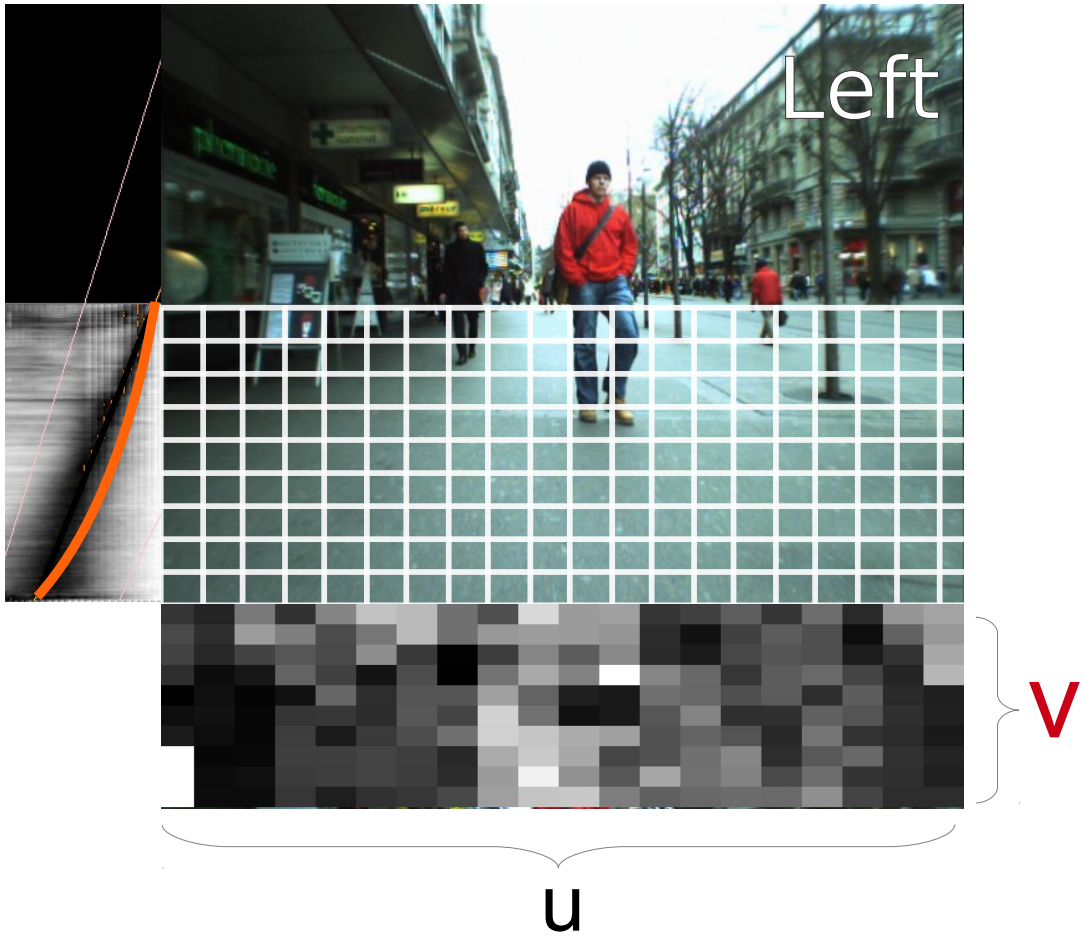
Too focused in u-disparity, instead of the image space

Too focused in u-disparity, instead of the image space



We will **quantize** both rows and columns

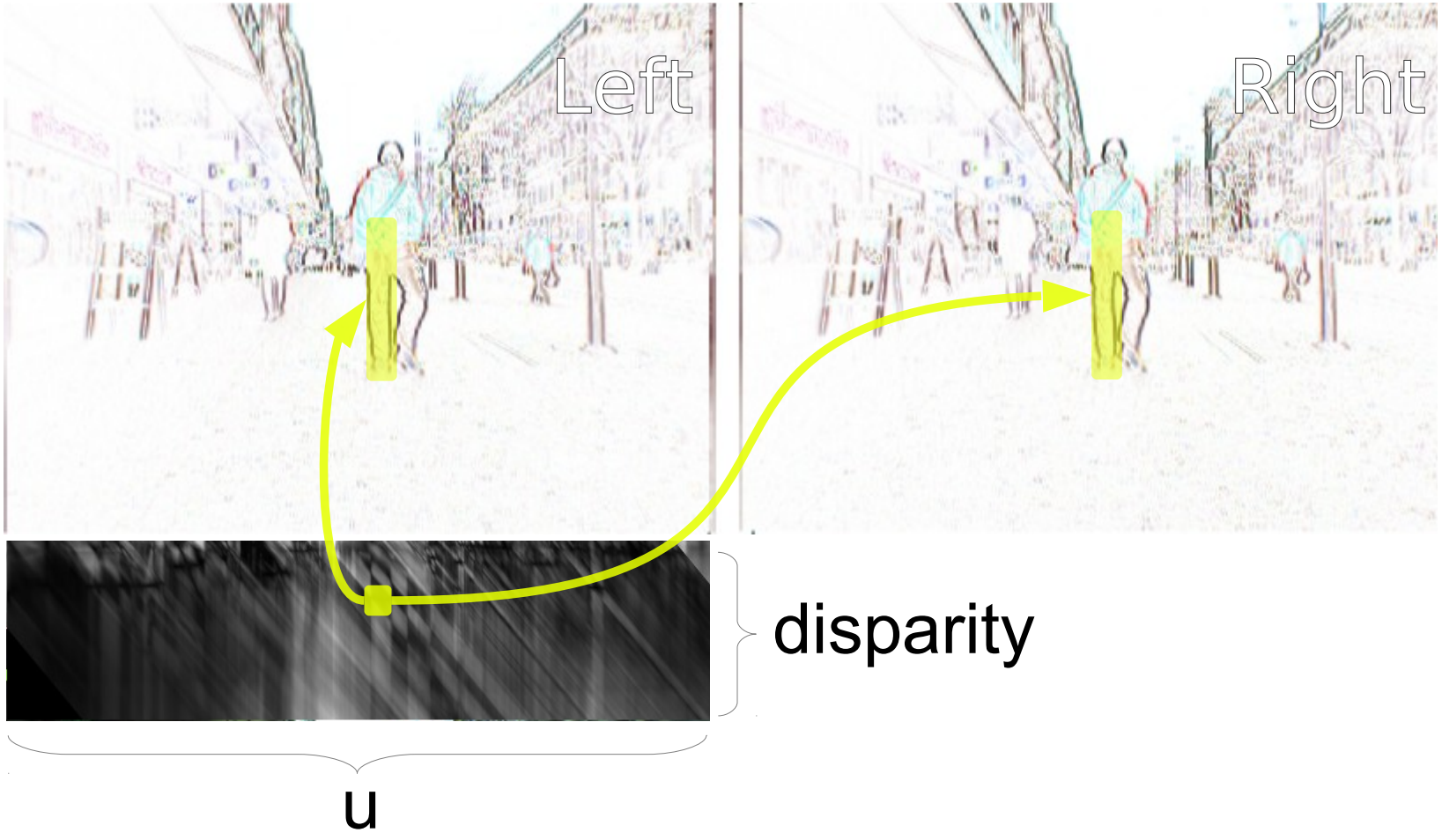
Too focused in u-disparity, instead of the u-v space



We will **quantize** both rows and columns

Only half of the image information is used

Only half of the image information is used



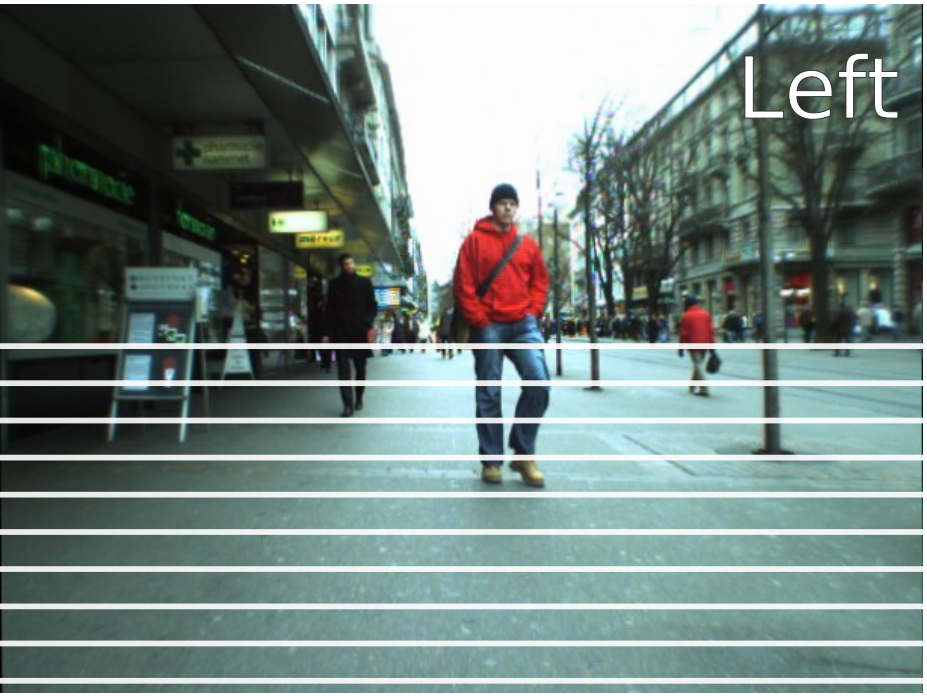
Only gradients along rows are used

Only half of the image information is used

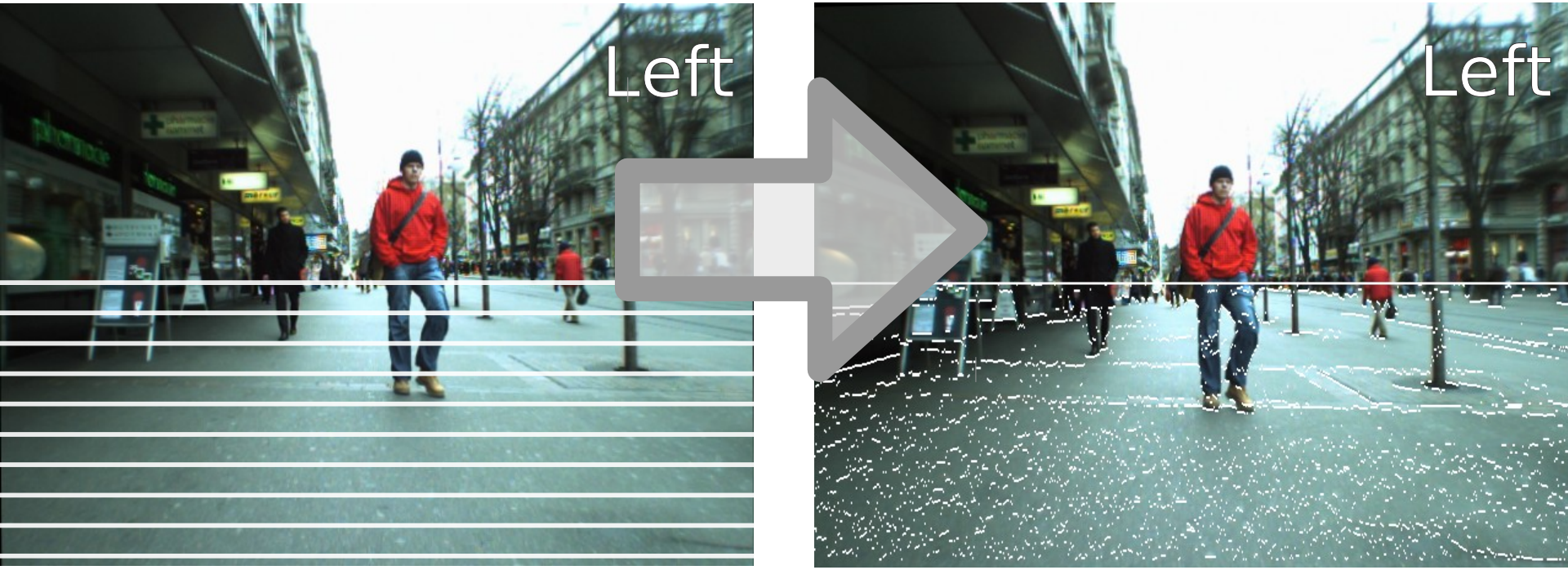
Gradients along columns are **ignored**

Now we fully exploit the left image

Now we fully exploit the left image

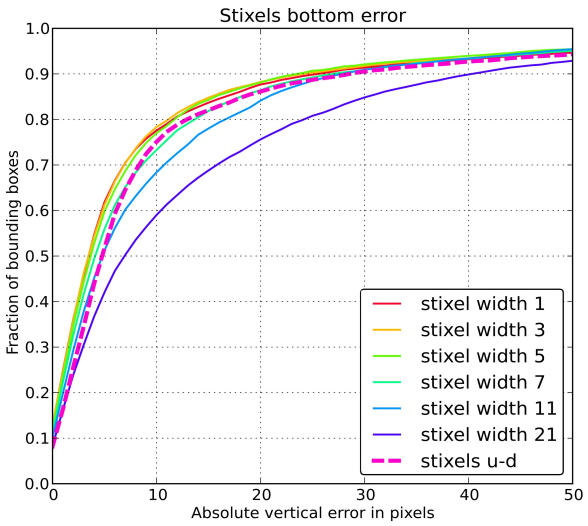


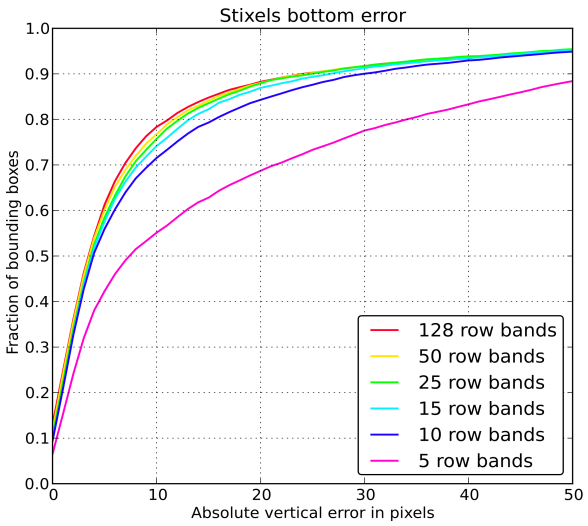
Now we fully exploit the left image

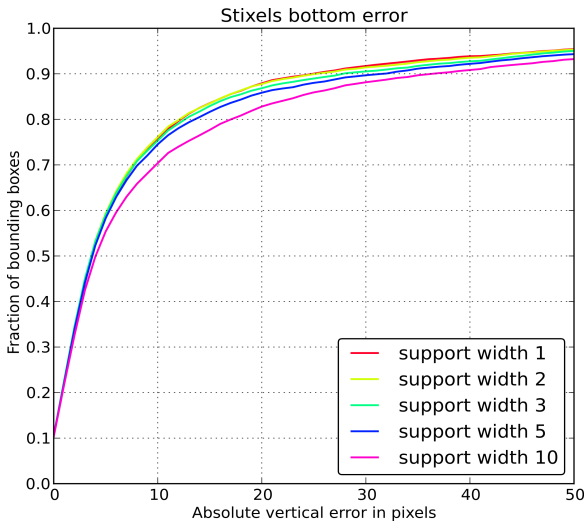


What happens when .... ?

- Stixels vertical sampling changes?
- Stixels horizontal sampling changes?
- Stixel support width changes?







What happens when .... ?

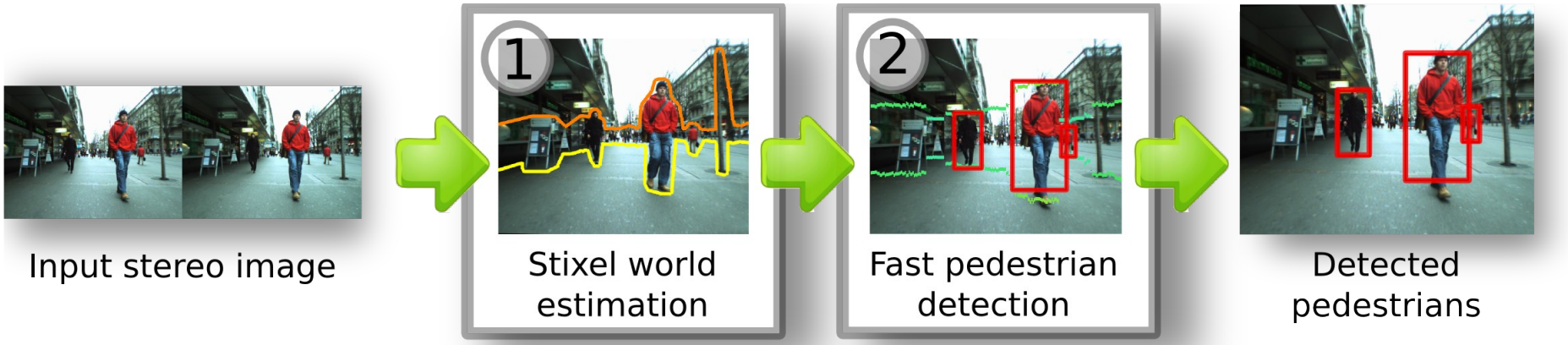
- Stixels vertical sampling changes?
  → each **3 columns** is good enough

- Stixels horizontal sampling changes?
  → **25 row bands** is good enough

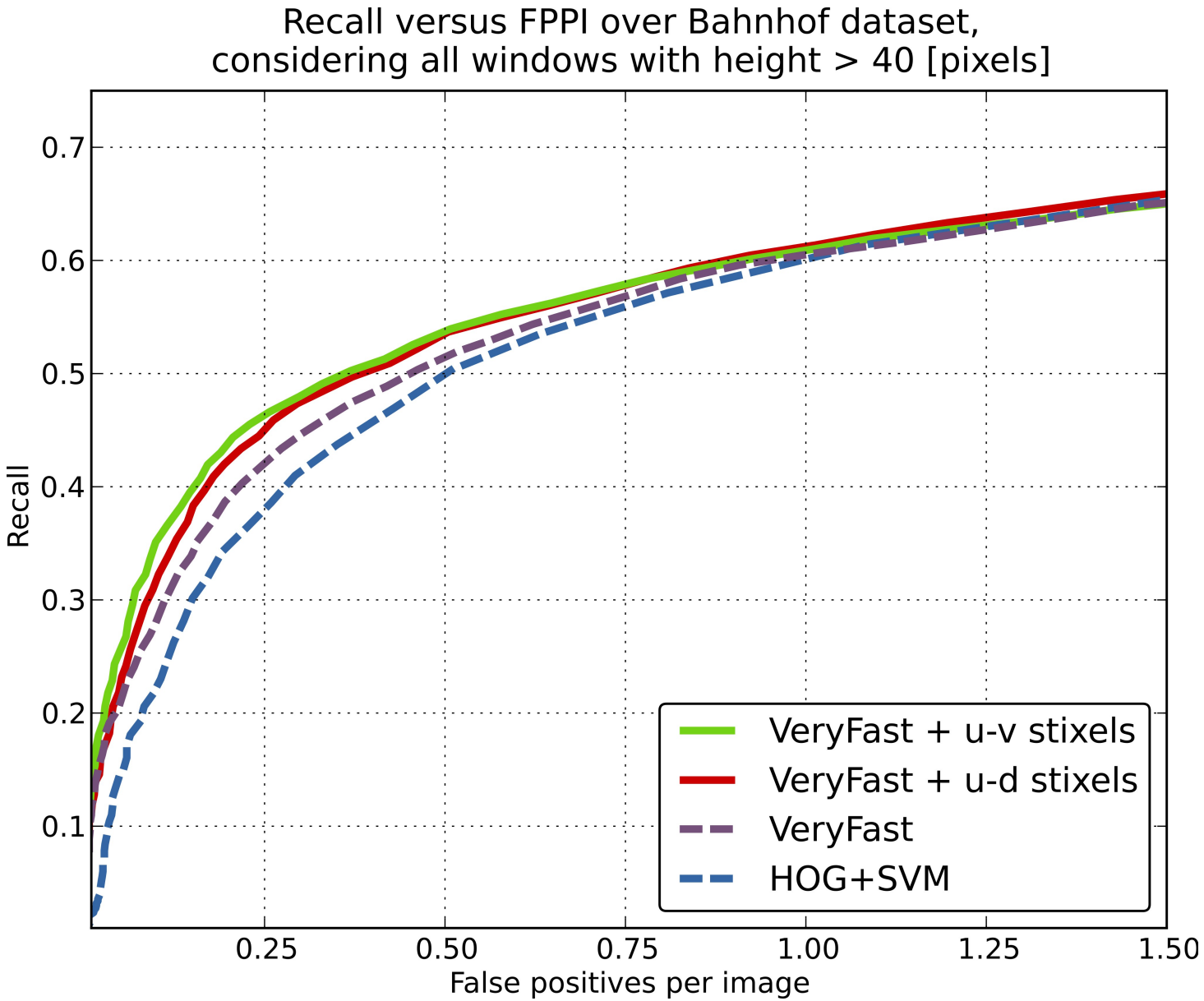
- Stixel support width changes?
  → **width 1 column** is best

~100 Hz ⇨ **~300** Hz on CPU

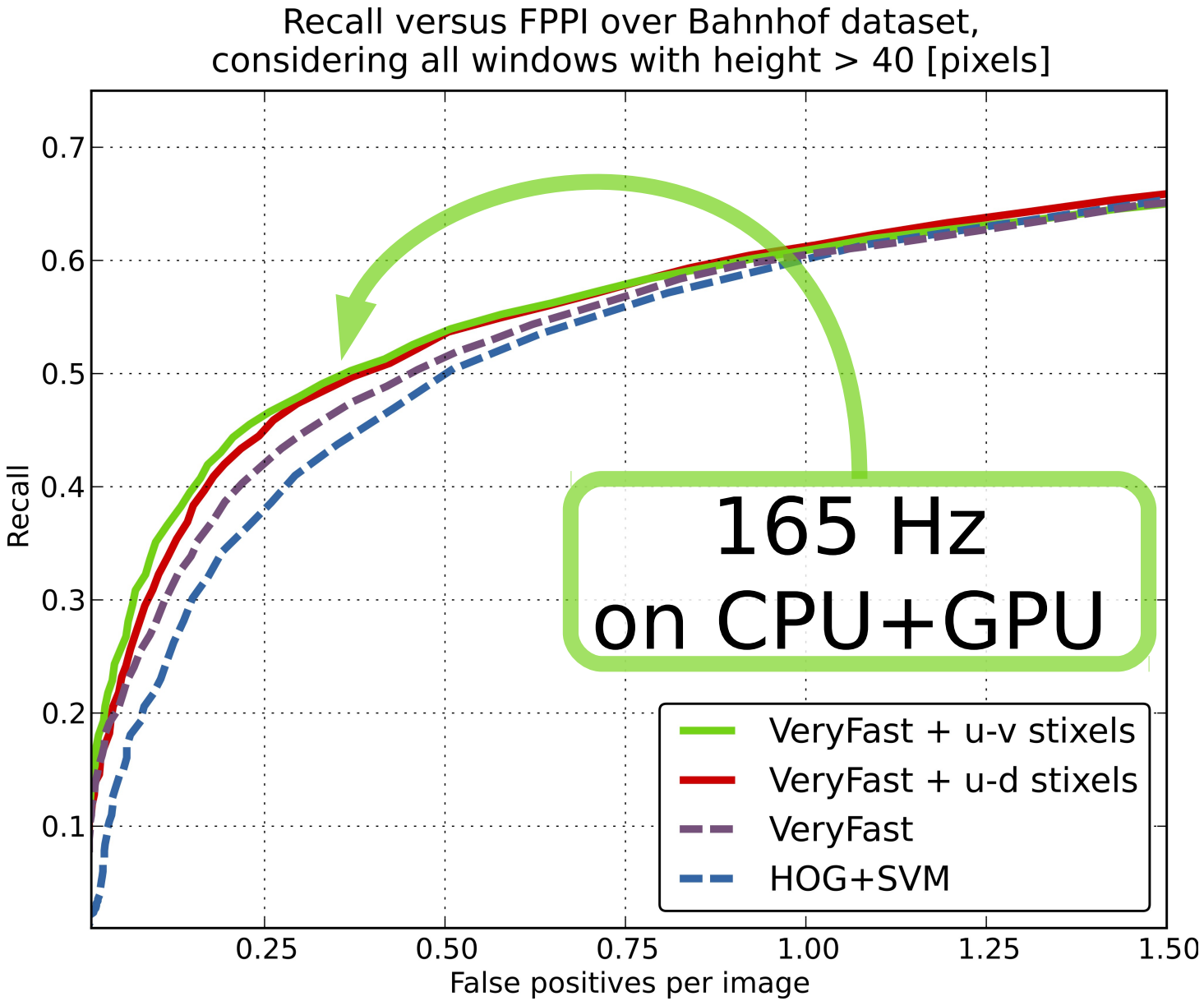
(on 640x480 pixels images)


1
2
Input stereo image
Stixel world estimation
Fast pedestrian detection
Detected pedestrians

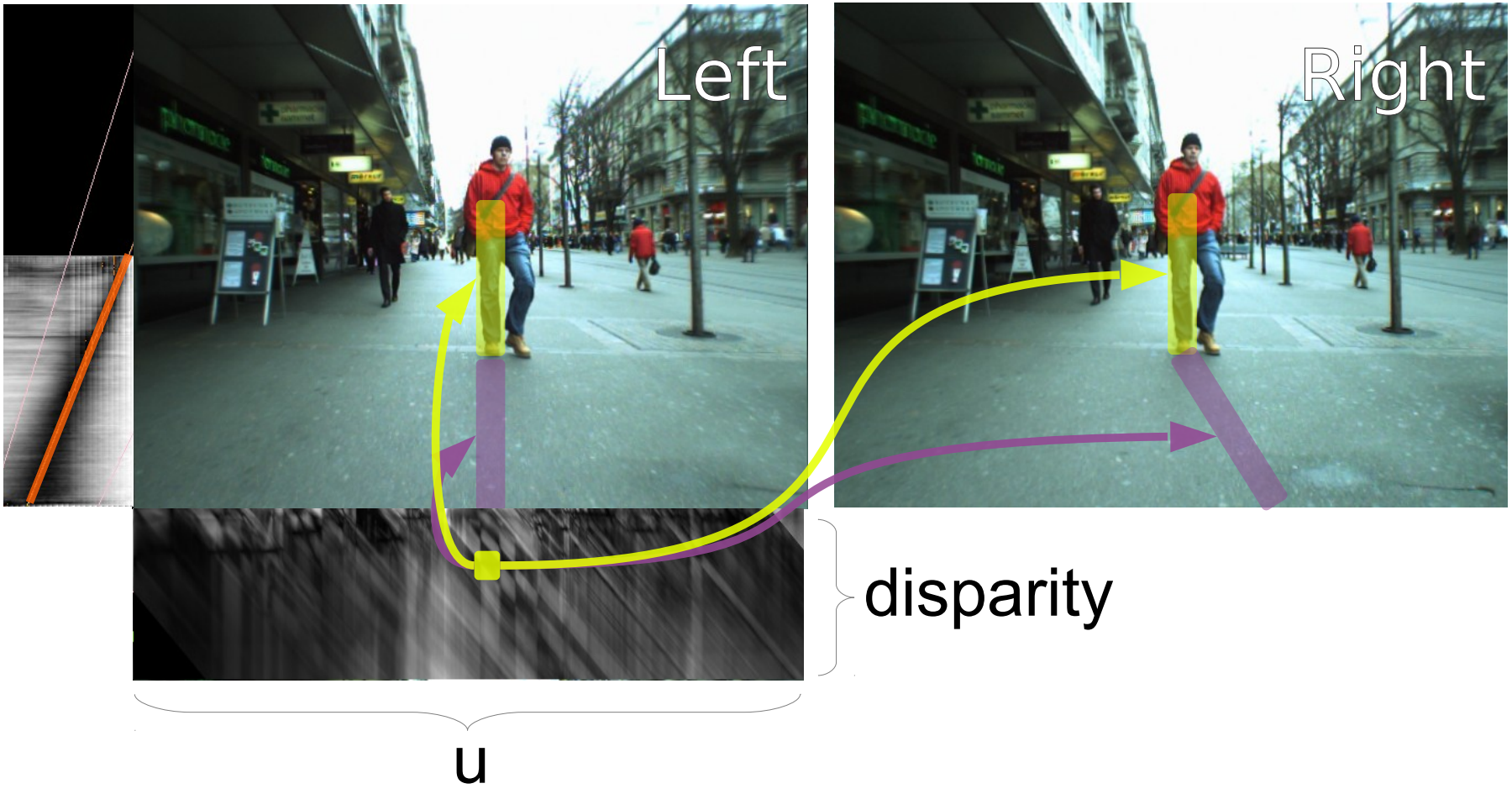
More speed, same quality



More speed, same quality

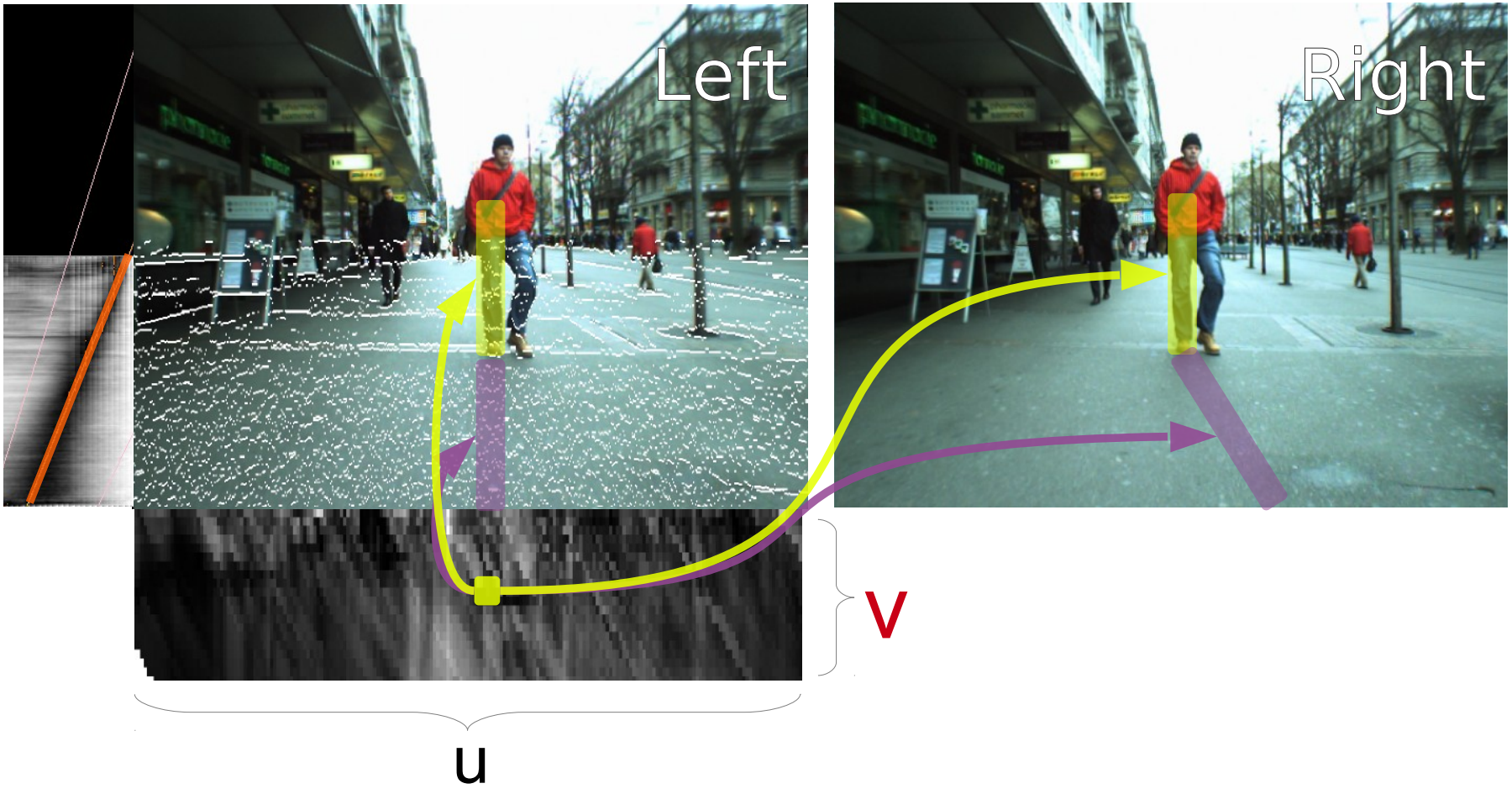


Stixel estimation in u-disparity space is good



Stixel distance estimation @ 135 Hz CPU

Stixel estimation in u-v space is better



2) Stixel distance estimation **@ 300 Hz CPU**

Take-away message:

Stixels are useful

and

they can be computed very fast

(300 Hz on CPU, less than 4 ms)

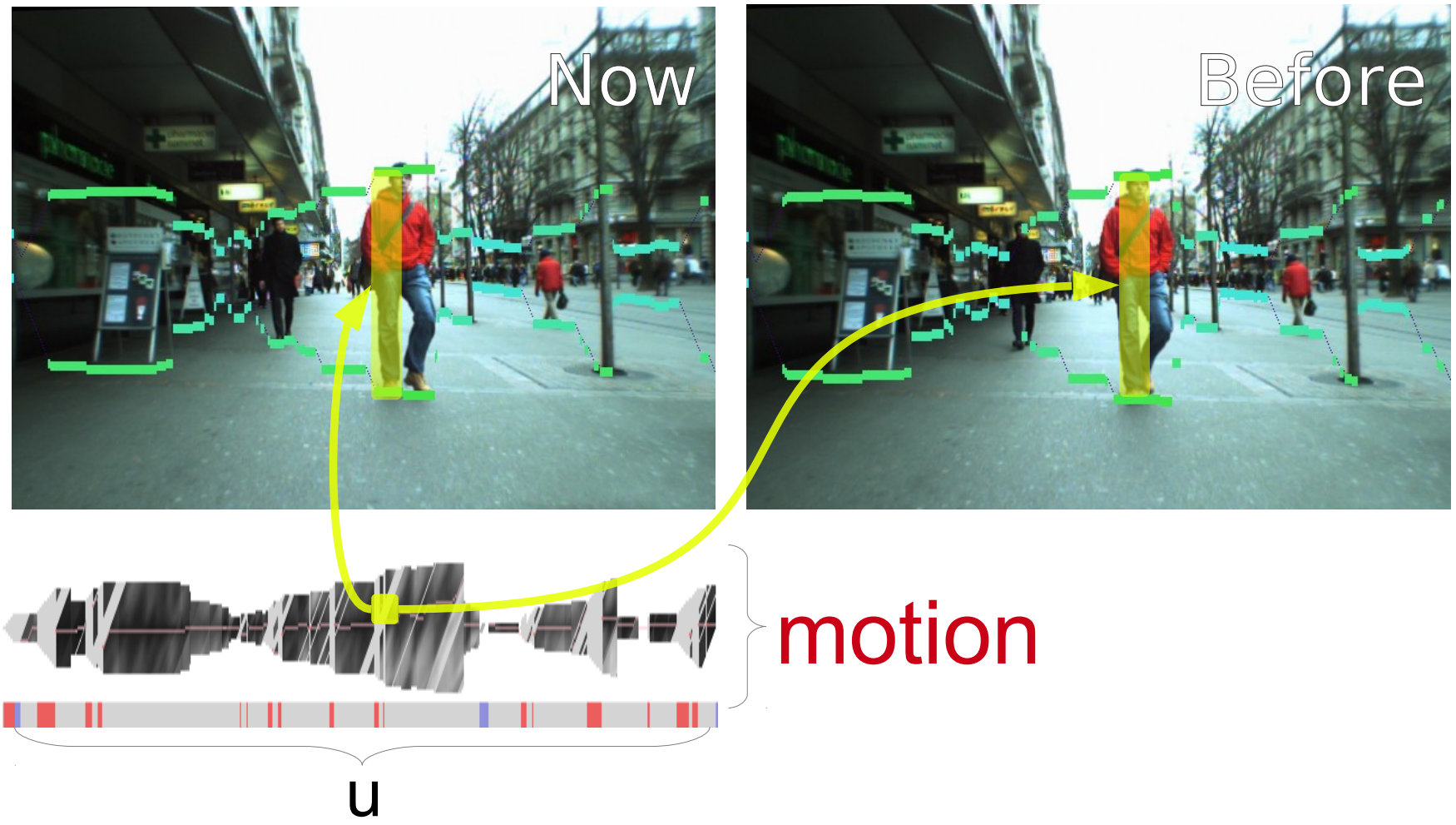
Source code available at
http://rodrigob.github.com

Rodrigo Benenson

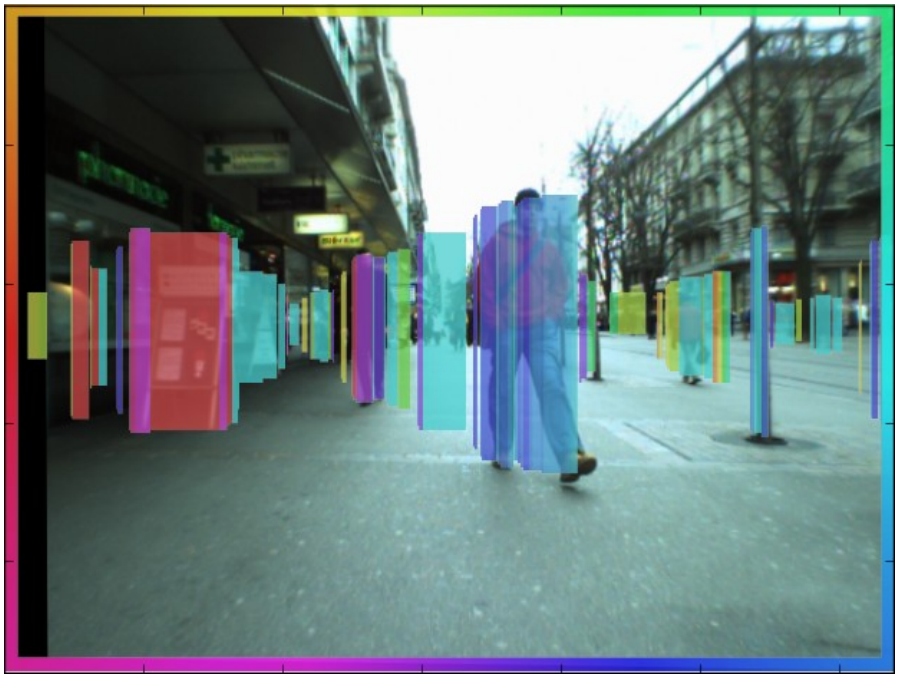
KATHOLIEKE UNIVERSITEIT
LEUVEN

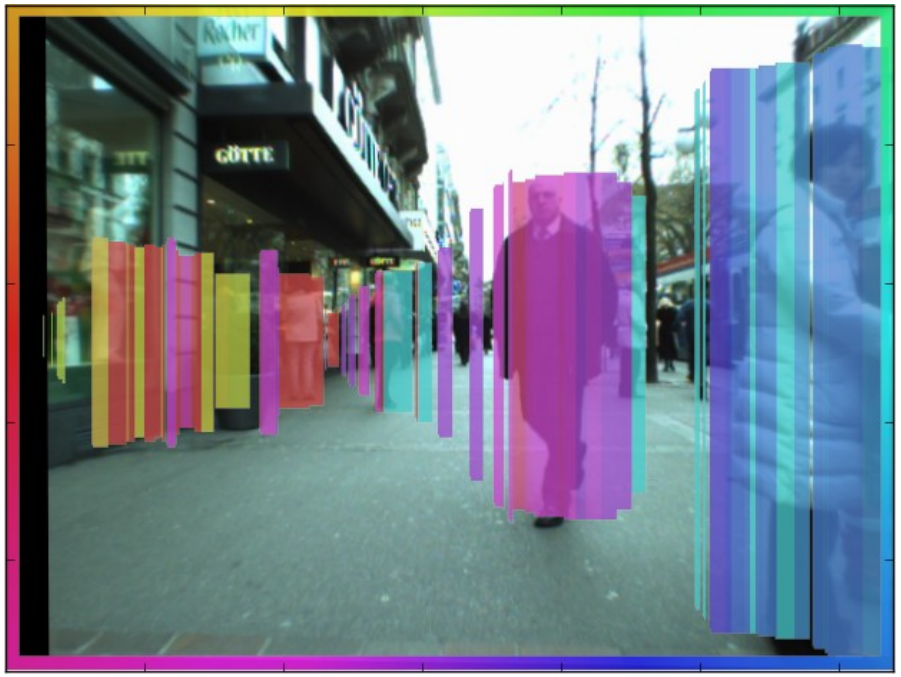
VISICS

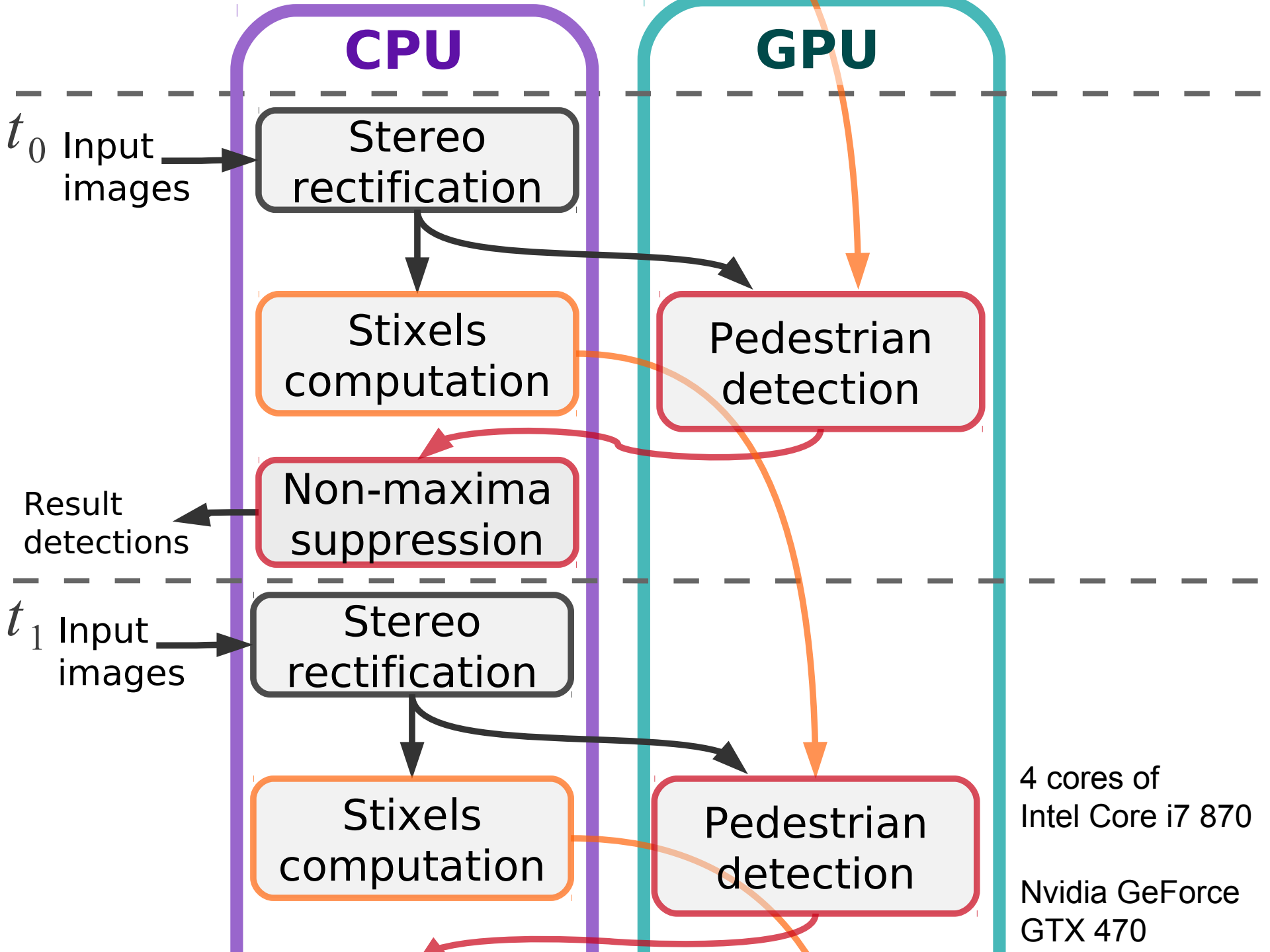
Direct stixels motion estimation

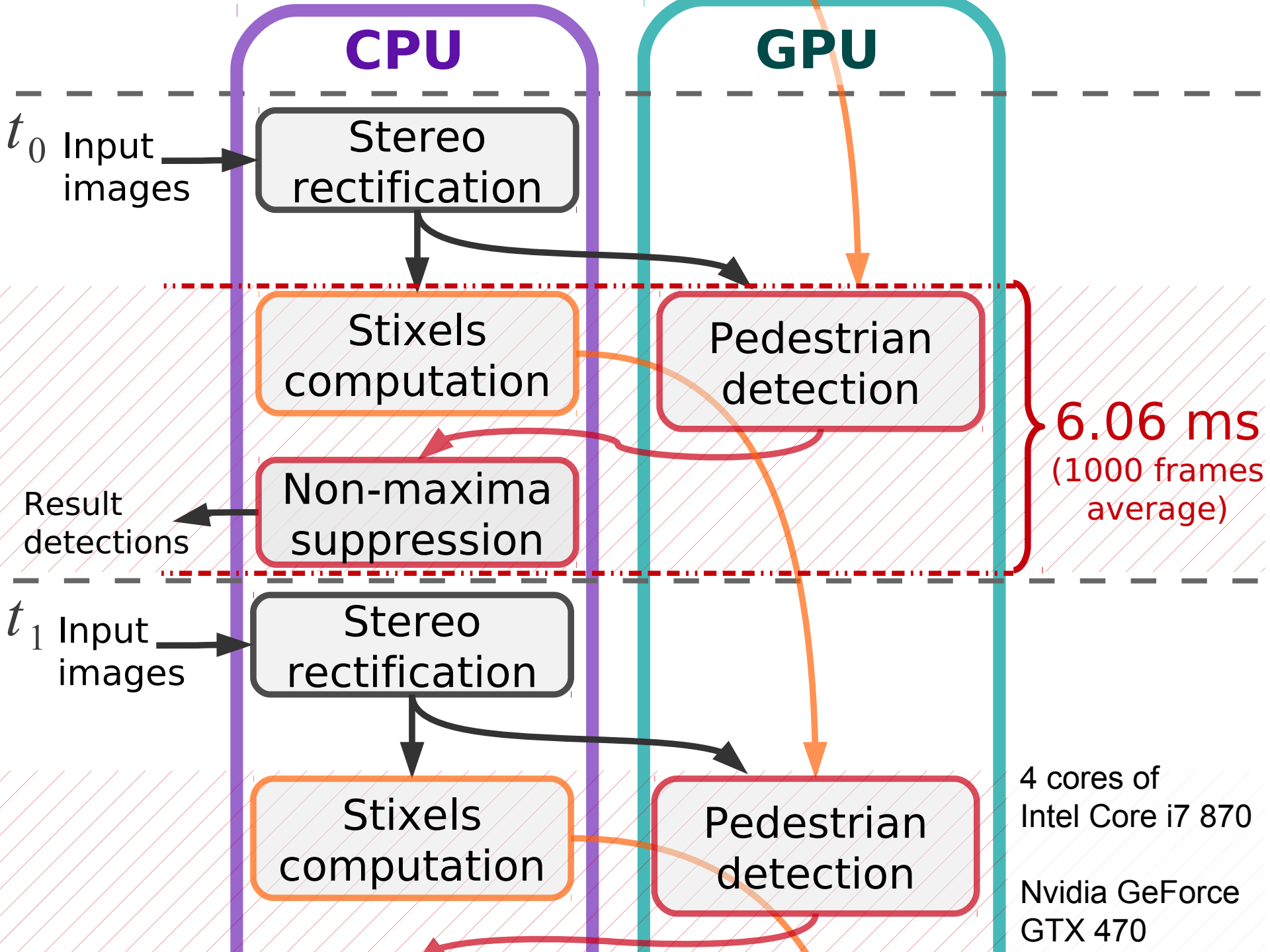


Direct stixels motion estimation






CPU
GPU
t_0 Input images
Stereo rectification
Stixels computation
Pedestrian detection
Non-maxima suppression
Result detections
t_1 Input images
Stereo rectification
Stixels computation
Pedestrian detection
4 cores of Intel Core i7 870
Nvidia GeForce GTX 470


CPU
GPU
t_0 Input images
Stereo rectification
Stixels computation
Pedestrian detection
Non-maxima suppression
Result detections
6.06 ms
(1000 frames average)
t_1 Input images
Stereo rectification
Stixels computation
Pedestrian detection
4 cores of Intel Core i7 870
Nvidia GeForce GTX 470