Computational photography & the Stanford Frankencamera

Marc Levoy

(for a more complete survey lecture on computational photography, but without the Frankencamera, see http://graphics.stanford.edu/talks/compphot-publictalk-may08.pdf)

Stanford Computer Graphics Laboratory
http://graphics.stanford.edu

Executive summary

- faster computing + communications will revolutionize digital photography, creating new markets
  - computational photography points the way
- research & commercialization of computational photography is being hampered by the lack of programmable cameras
  - traditional cameras are closed platforms
- open-source cameras will benefit the research community and accelerate the revolution
  - 3rd party developers, plugins, apps

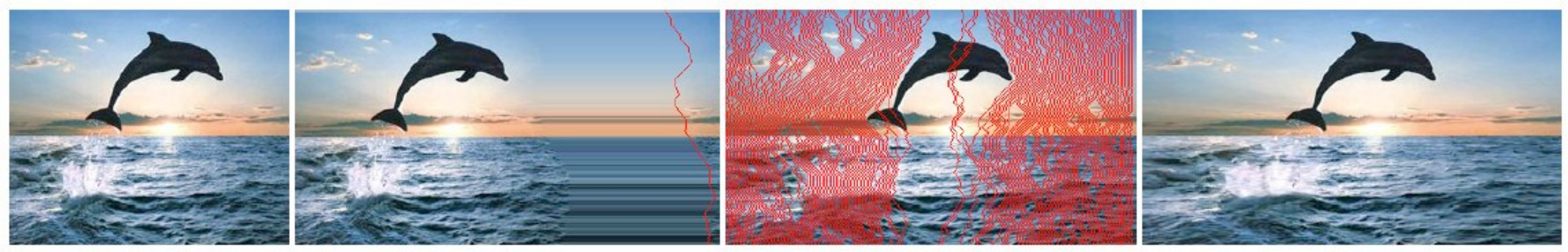
Some (tentative) definitions

- *computational imaging*
  - –any image formation method that requires a digital computer
- *computational photography*
  - –computational imaging techniques that enhance or extend the capabilities of digital photography
  - –output is an ordinary photograph, but one that could not have been taken by a traditional camera

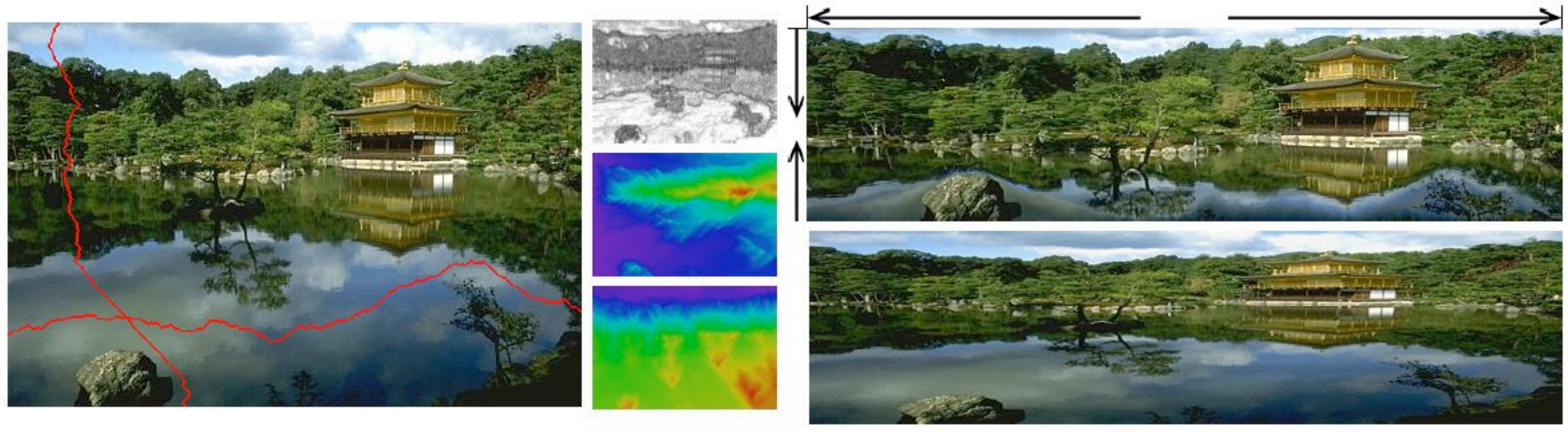
Film-like Photography with bits	Computational Photography			
	Computational Camera			Smart Light
Digital Photography	Computational Processing	Computational Imaging/Optics	Computational Sensor	Computational Illumination
Image processing applied to captured images to produce better images.	Processing of a set of captured images to create new images.	Capture of optically coded images and computational decoding to produce new images.	Detectors that combine sensing and processing to create smart pixels.	Adapting and Controlling Illumination to Create revealing image
Examples: Interpolation, Filtering, Enhancement, Dynamic Range Compression, Color Management, Morphing, Hole Filling, Artistic Image Effects, Image Compression, Watermarking.	Examples: Mosaicing, Matting, Super-Resolution, Multi-Exposure HDR, Light Field from Mutiple View, Structure from Motion, Shape from X.	Examples: Coded Aperture, Optical Tomography, Diaphanography, SA Microscopy, Integral Imaging, Assorted Pixels, Catadioptric Imaging, Holographic Imaging.	Examples: Artificial Retina, Retinex Sensors, Adaptive Dynamic Range Sensors, Edge Detect Chips, Focus of Expansion Chips, Motion Sensors.	Examples: Flash/no flash, Lighting domes, Multi-flash for depth edges, Dual Photos, Polynomial texture Maps, 4D light source

[Nayar, Tumblin]

Content-aware image resizing
[Avidan SIGGRAPH 2007]



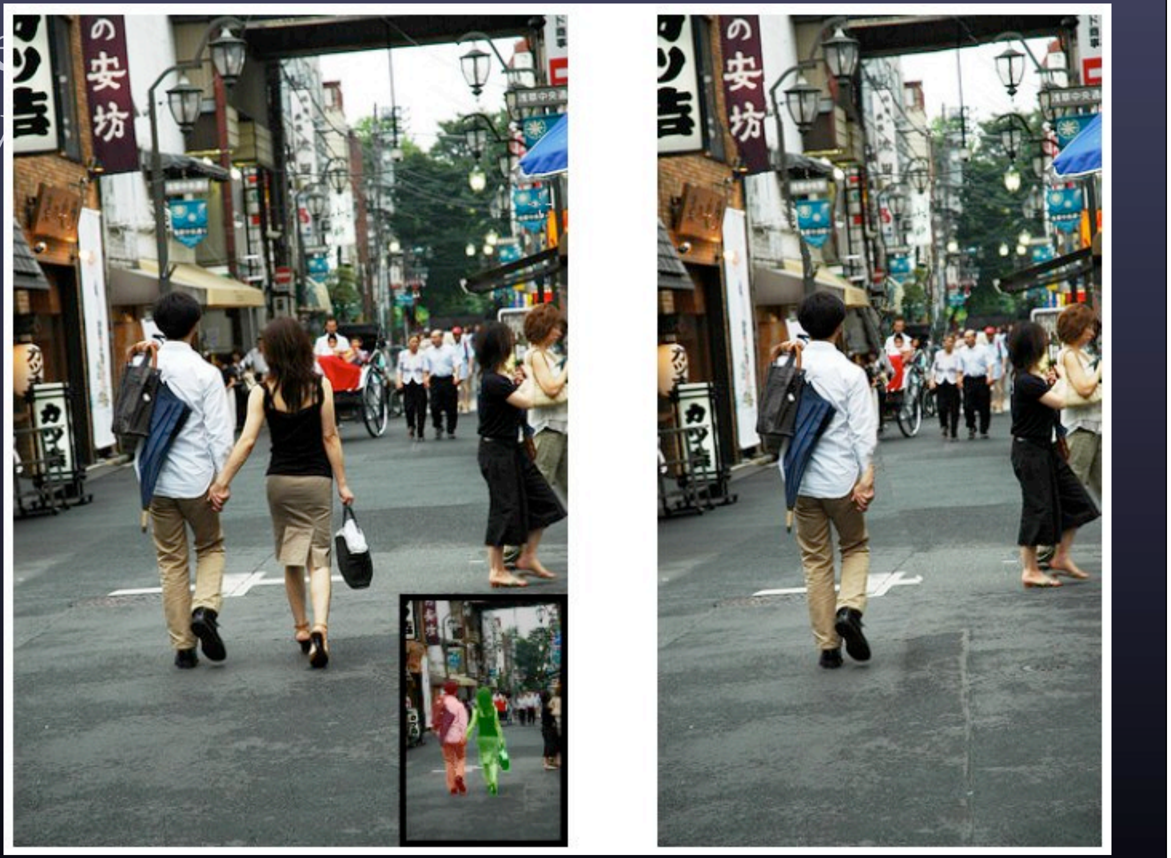
- to expand: insert pixels along seams that, if removed in order, would yield the original image



Content-aware image resizing
[Avidan SIGGRAPH 2007]

- to compress: remove pixels along lowest-energy seams, ordered using dynamic programming
- to expand: insert pixe in order, w
- application to object removal

NOW AVAILABLE IN PHOTOSHOP !!



Removing camera shake
[Fergus SIGGRAPH 2006]

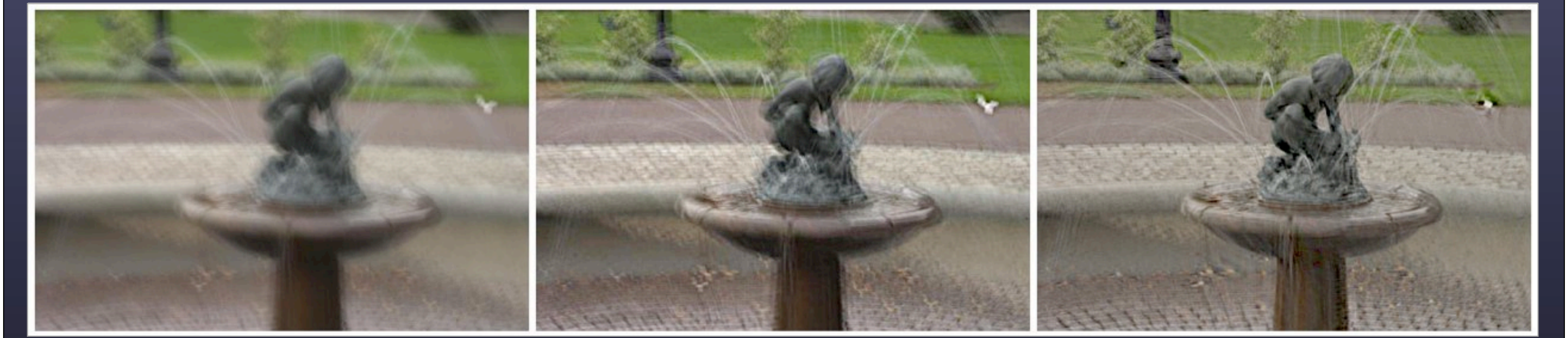
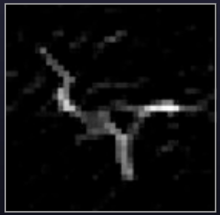


image with camera shake

Photoshop Unsharp Mask

deconvolution

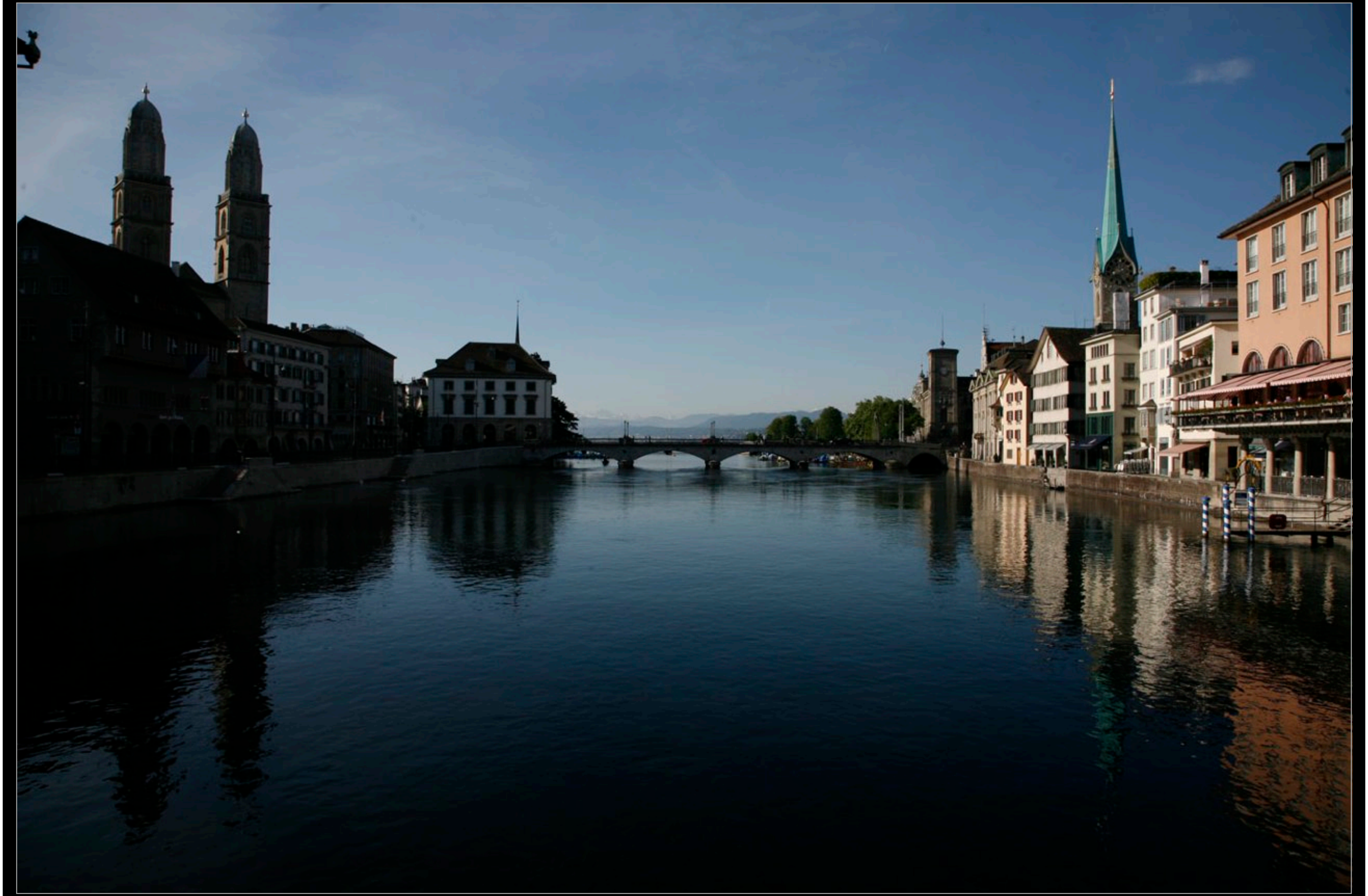


blur kernel

Film-like Photography with bits	Computational Photography			
	Computational Camera			Smart Light
Digital Photography	Computational Processing	Computational Imaging/Optics	Computational Sensor	Computational Illumination
Image processing applied to captured images to produce better images.	Processing of a set of captured images to create new images.	Capture of optically coded images and computational decoding to produce new images.	Detectors that combine sensing and processing to create smart pixels.	Adapting and Controlling Illumination to Create revealing image
Examples: Interpolation, Filtering, Enhancement, Dynamic Range Compression, Color Management, Morphing, Hole Filling, Artistic Image Effects, Image Compression, Watermarking.	Examples: Mosaicing, Matting, Super-Resolution, Multi-Exposure HDR, Light Field from Mutiple View, Structure from Motion, Shape from X.	Examples: Coded Aperture, Optical Tomography, Diaphanography, SA Microscopy, Integral Imaging, Assorted Pixels, Catadioptric Imaging, Holographic Imaging.	Examples: Artificial Retina, Retinex Sensors, Adaptive Dynamic Range Sensors, Edge Detect Chips, Focus of Expansion Chips, Motion Sensors.	Examples: Flash/no flash, Lighting domes, Multi-flash for depth edges, Dual Photos, Polynomial texture Maps, 4D light source

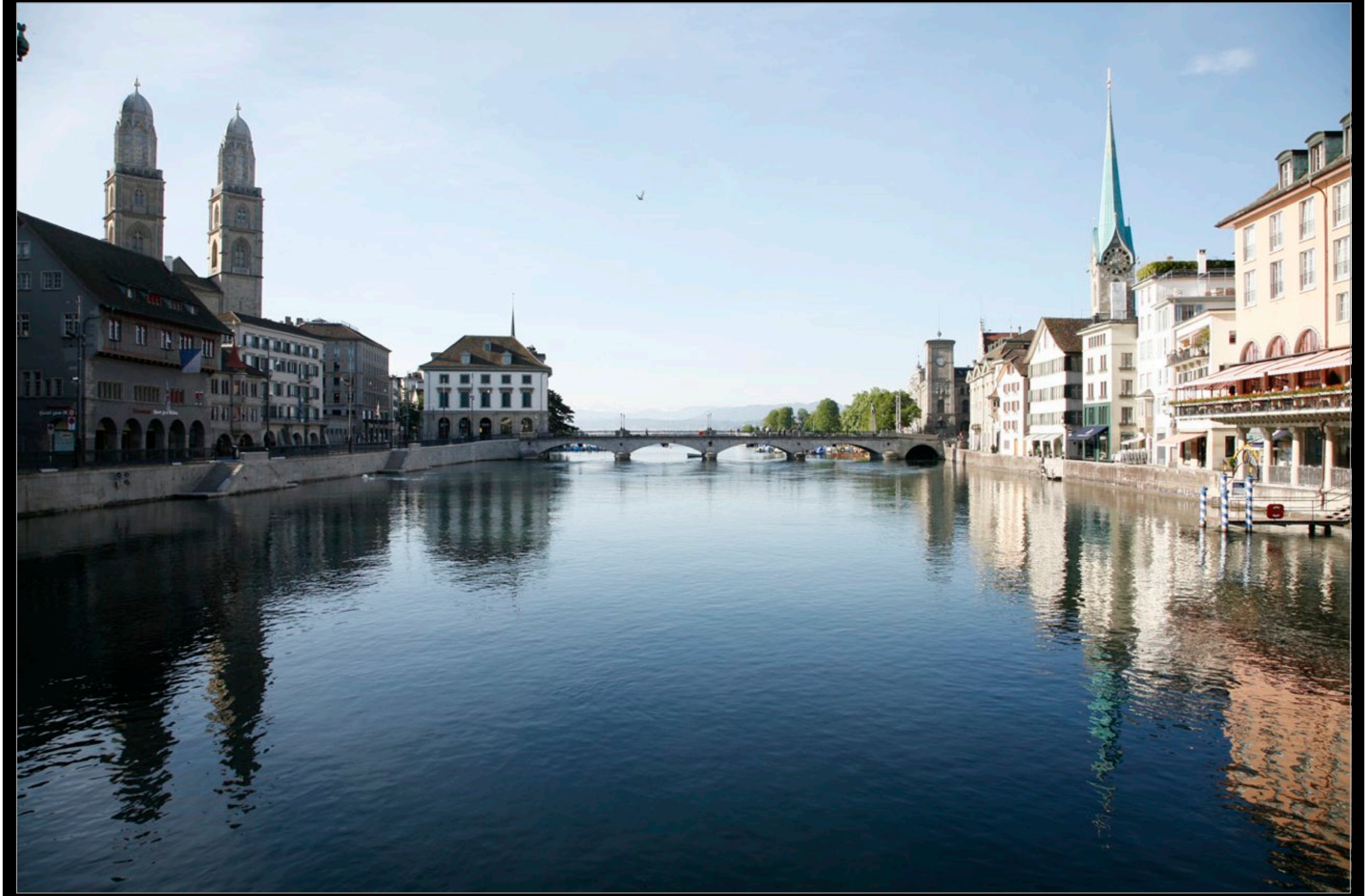
[Nayar, Tumblin]

High dynamic range (HDR) imaging



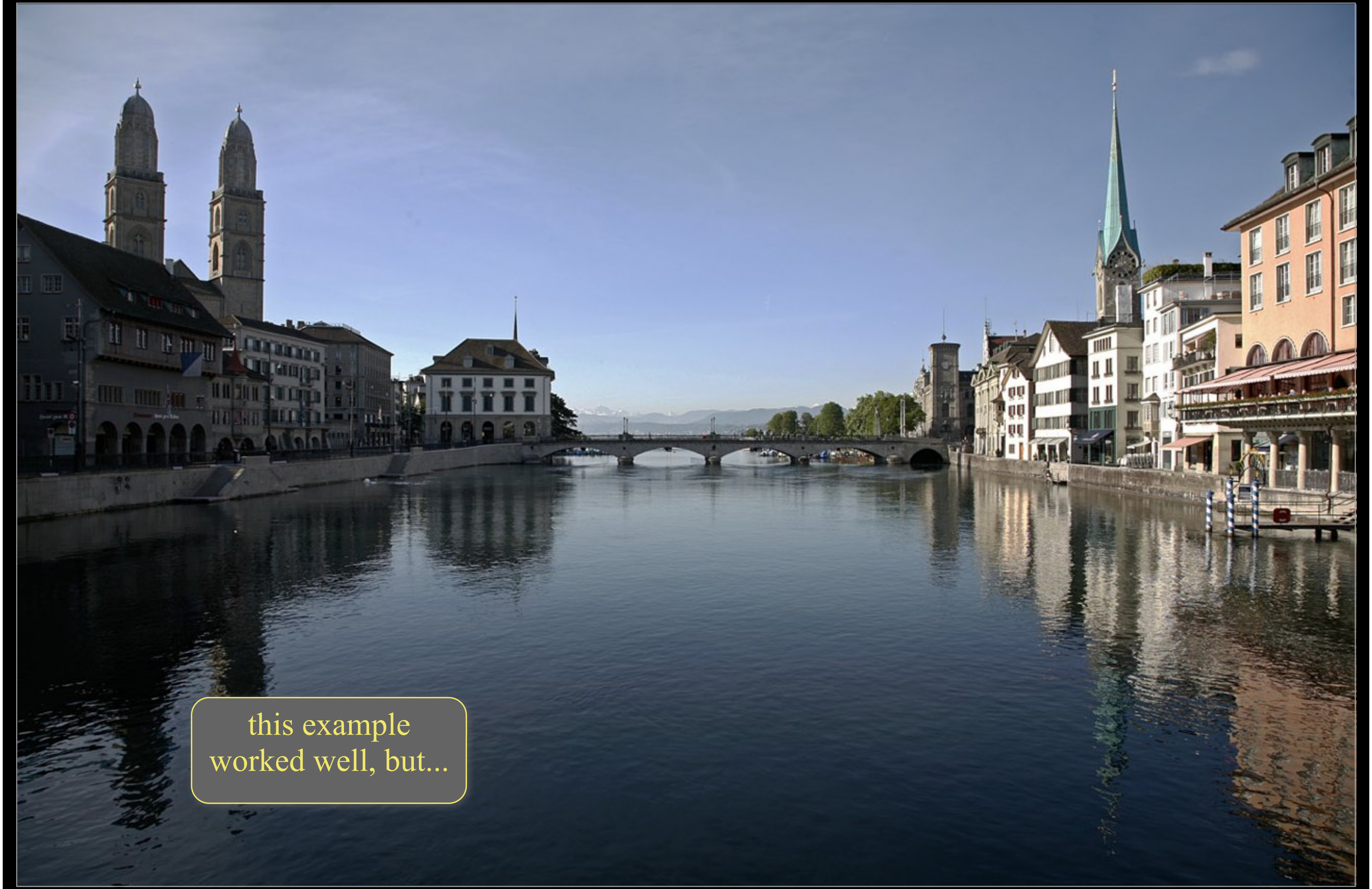
Too dark

High dynamic range (HDR) imaging



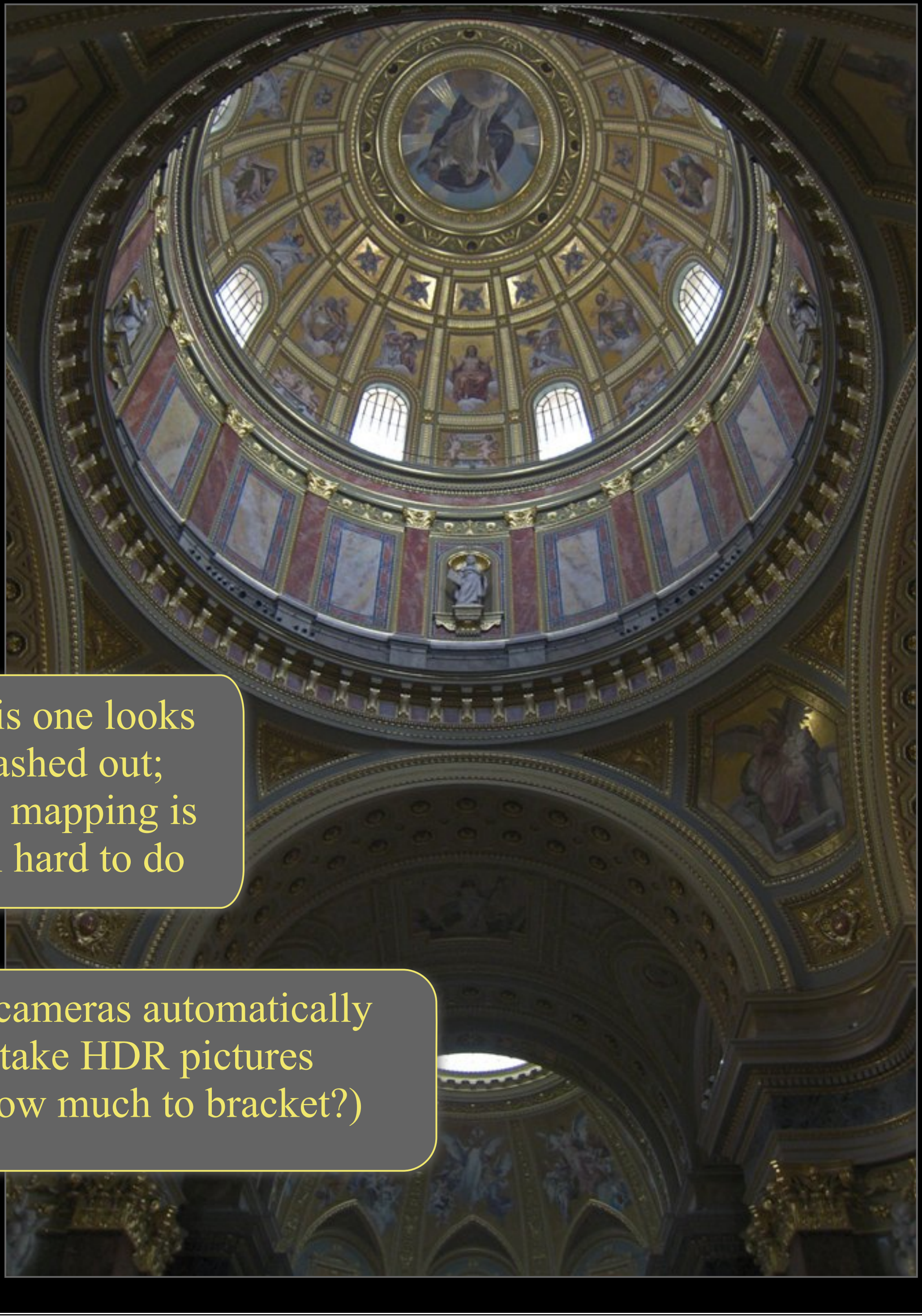
Too bright

High dynamic range (HDR) imaging



this example worked well, but...

Tone mapped combination

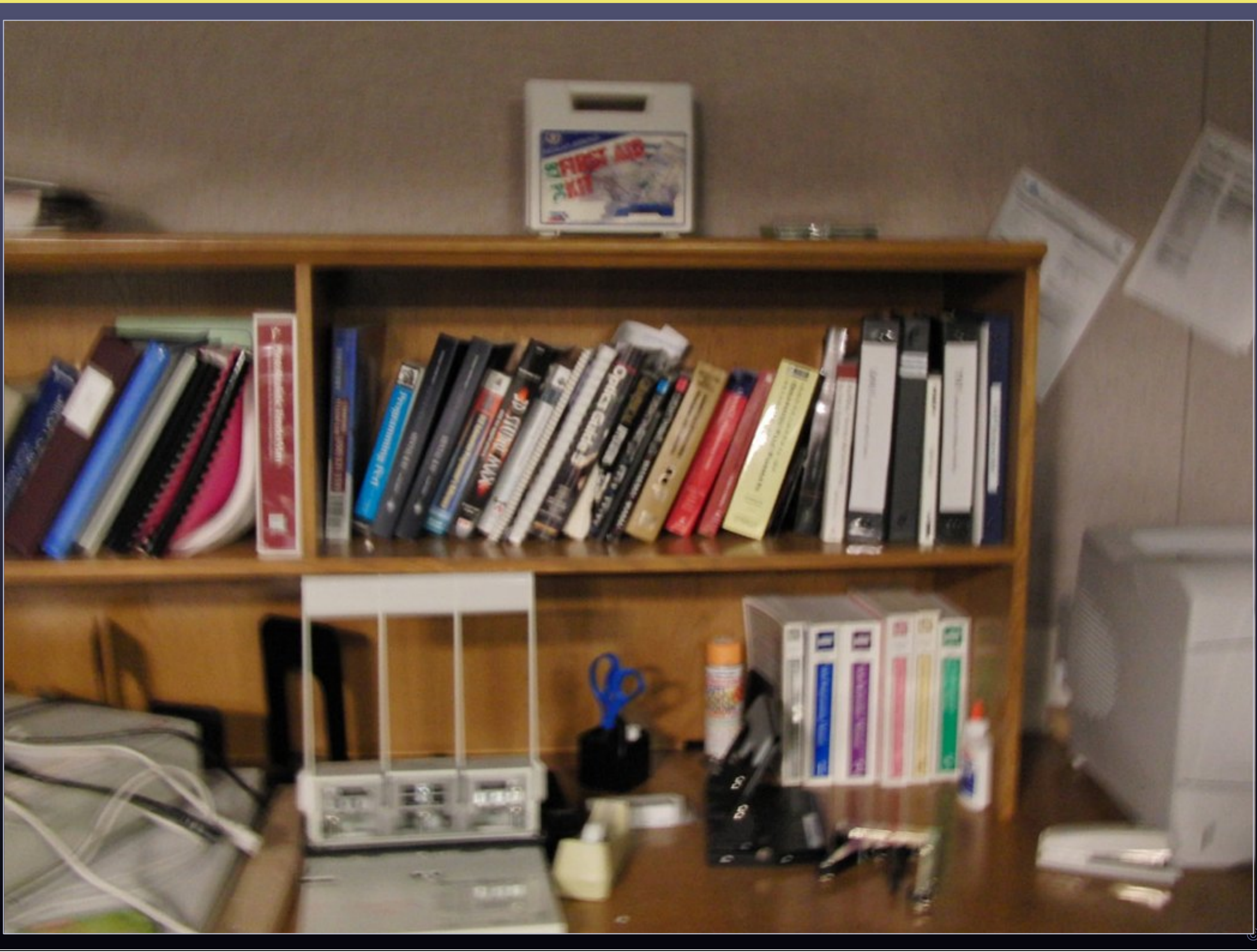


...this one looks washed out; tone mapping is still hard to do

no cameras automatically take HDR pictures (How much to bracket?)

Aligning a burst of short-exposure, high-ISO shots using the Casio EX-F1

1/3 sec

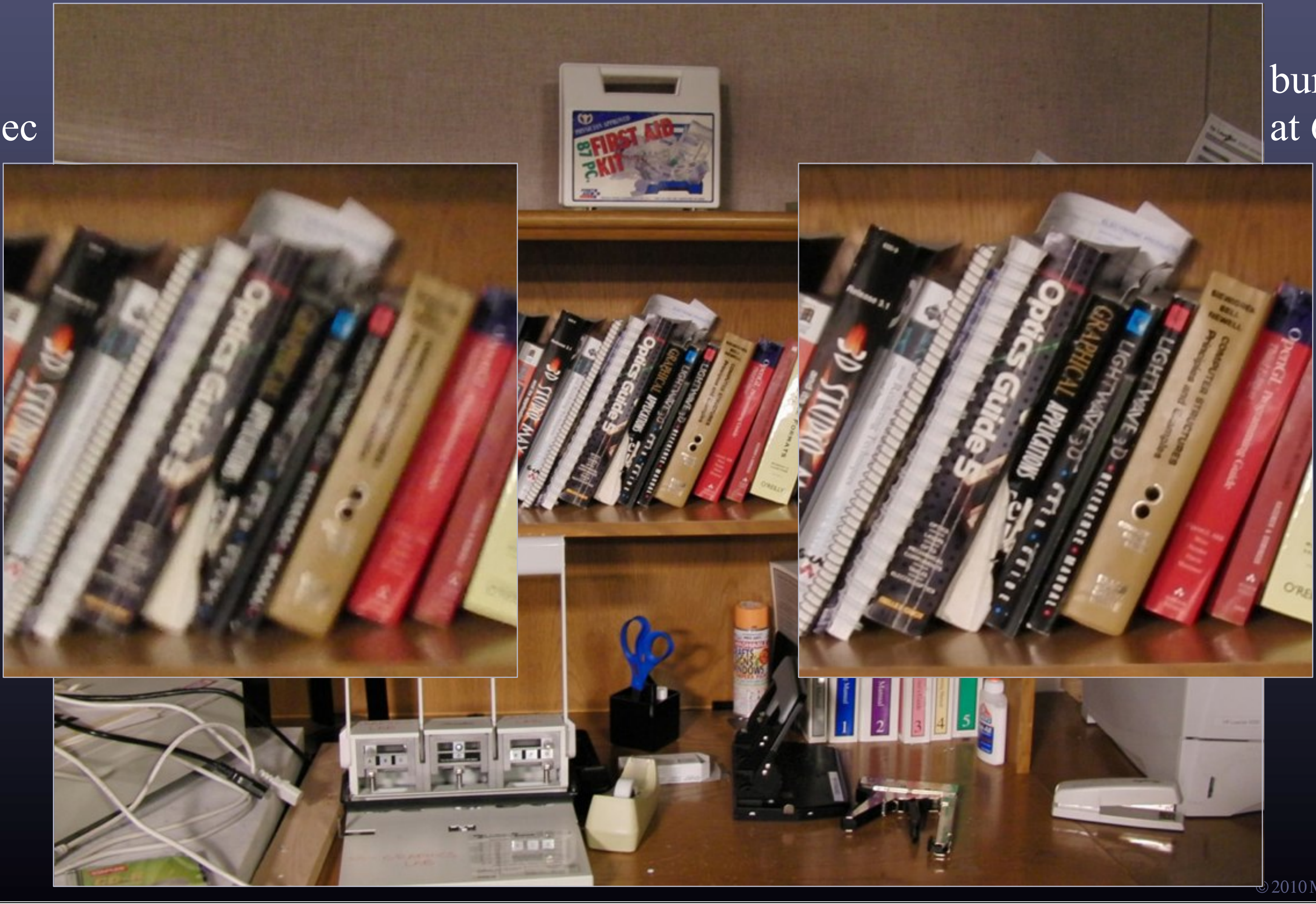


Aligning a burst of short-exposure, high-ISO shots using the Casio EX-F1

burst
at 60fps

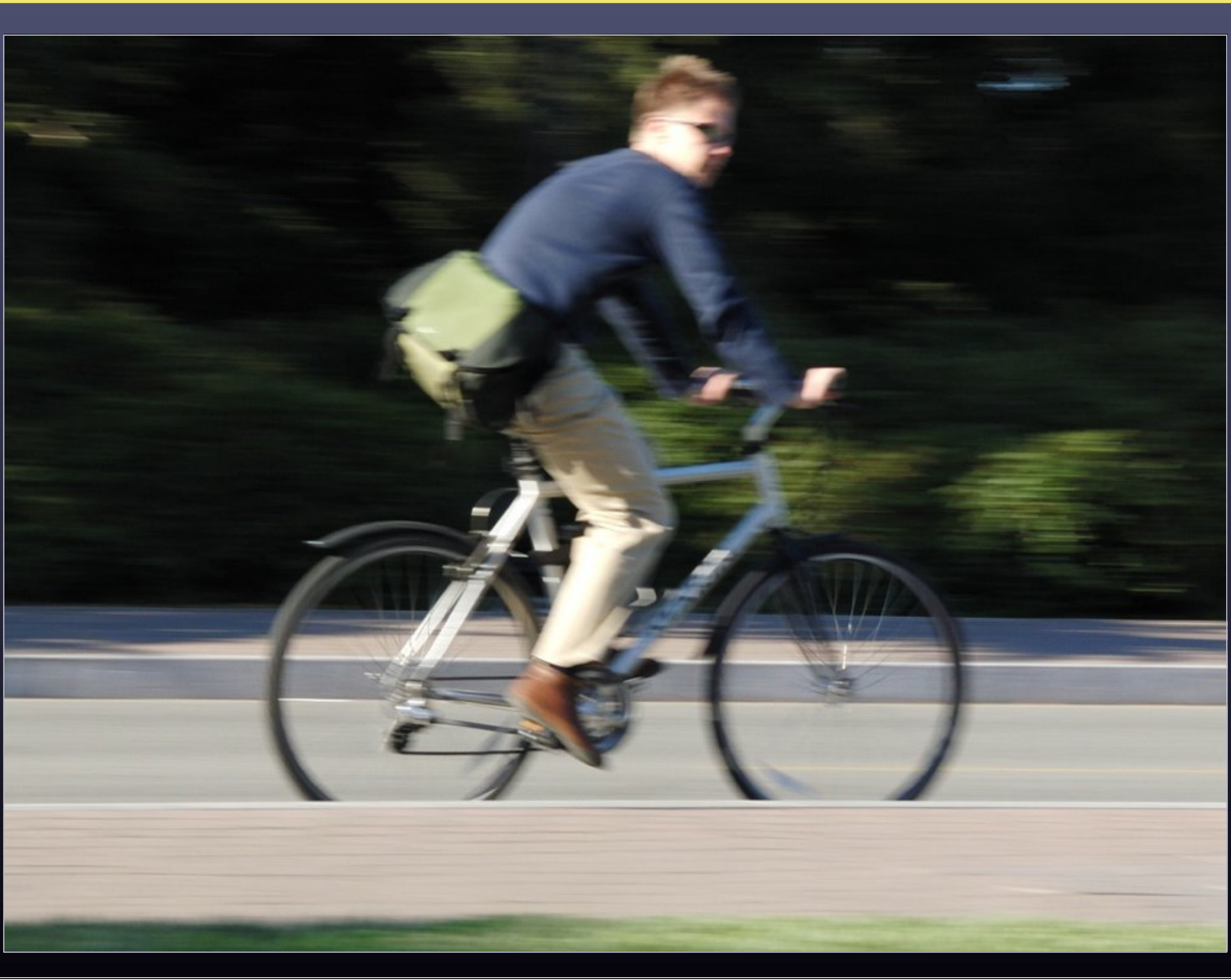
Aligning a burst of short-exposure, high-ISO shots using the Casio EX-F1

1/3 sec

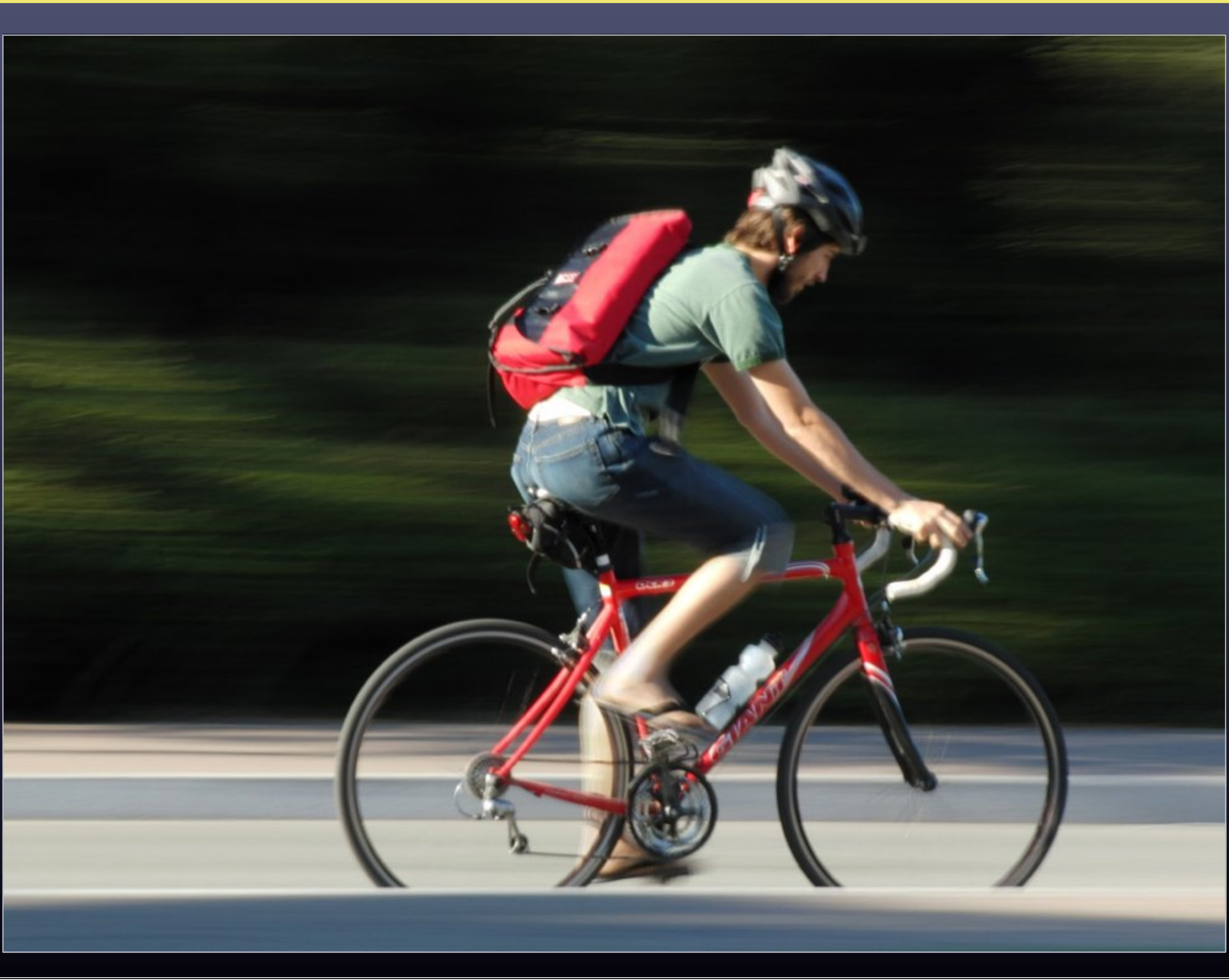


burst
at 60fps

Aligning on a foreground object using the Casio EX-F1



Aligning on a foreground object using the Casio EX-F1



All-focus algorithms
[Agarwala 2004]

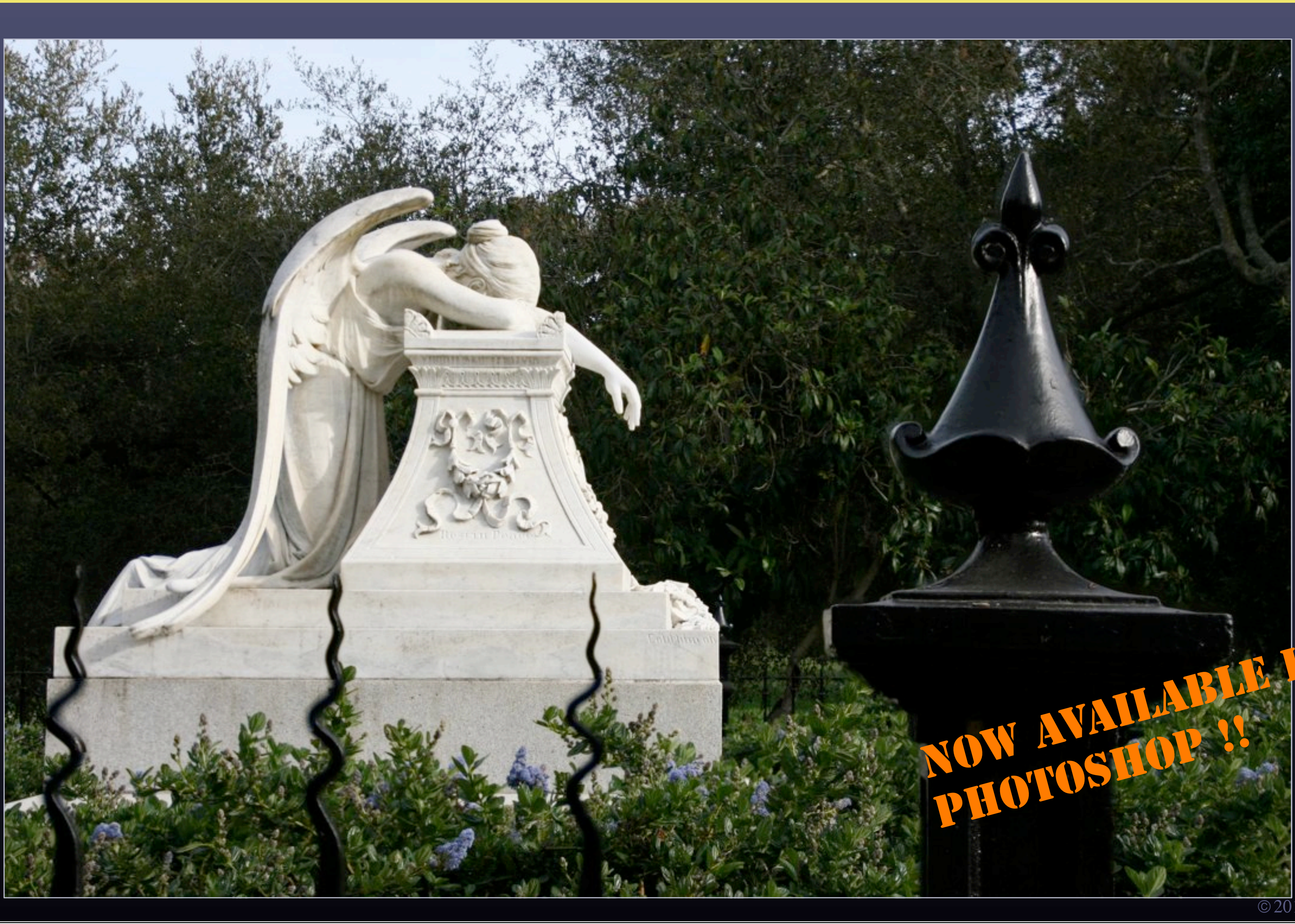
All-focus algorithms
[Agarwala 2004]

All-focus algorithms
[Agarwala 2004]

All-focus algorithms
[Agarwala 2004]

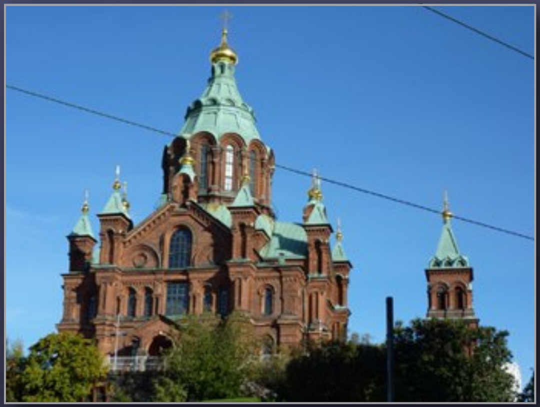
All-focus algorithms
[Agarwala 2004]

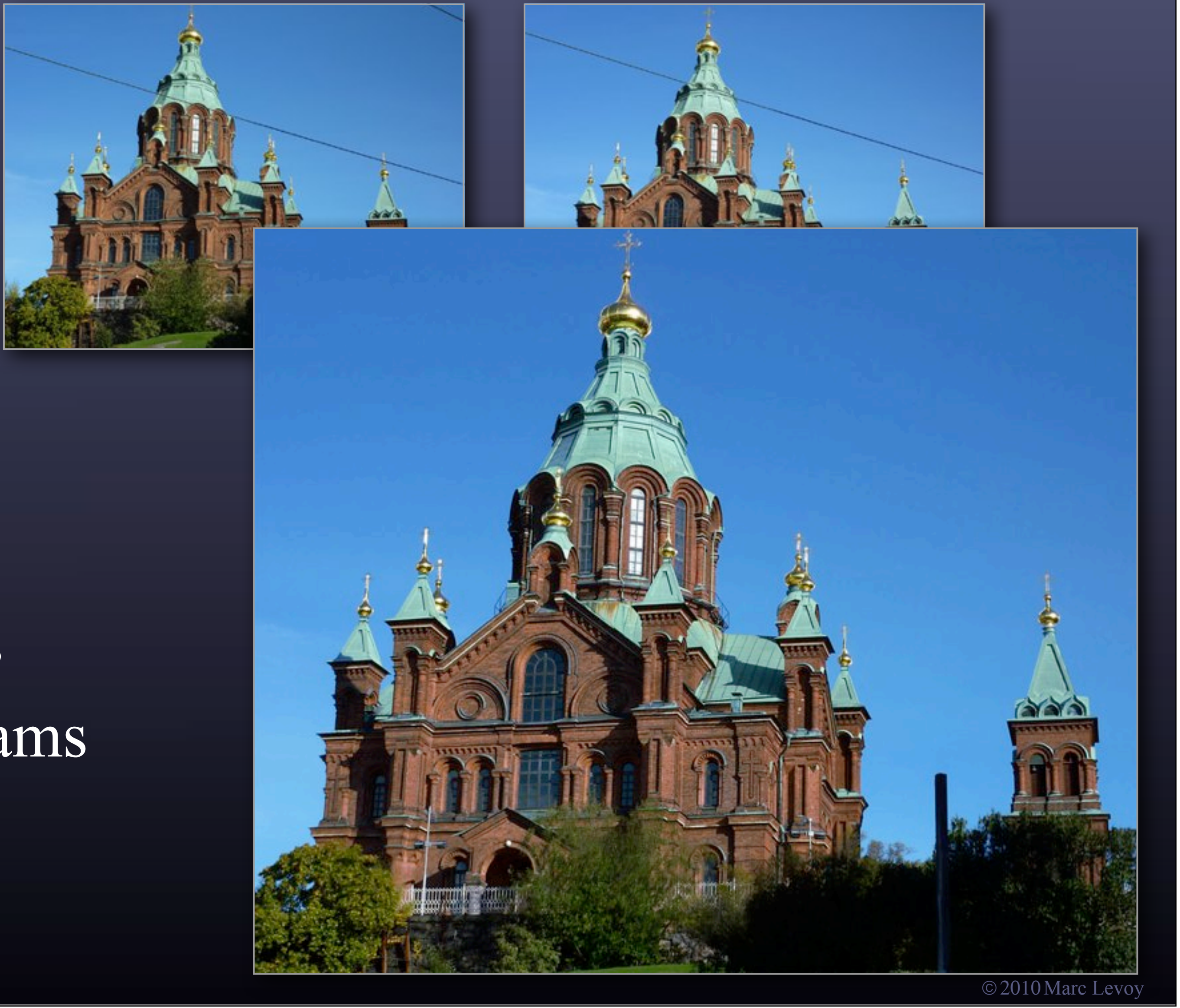
all-focus



NOW AVAILABLE IN PHOTOSHOP !!

Removing foreground objects by translating the camera



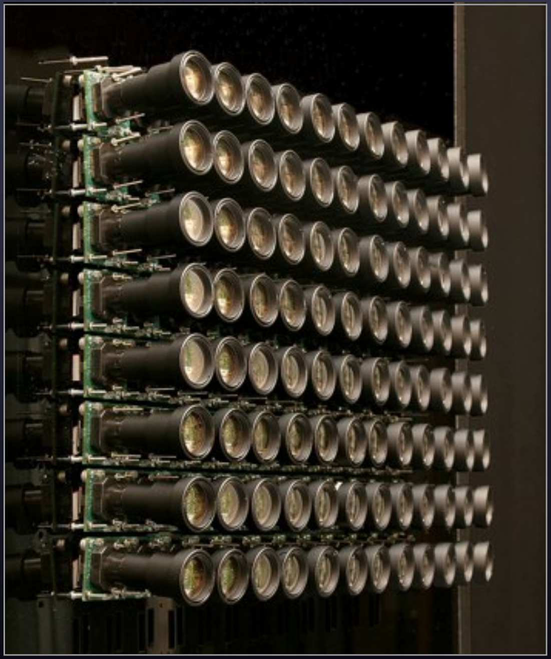


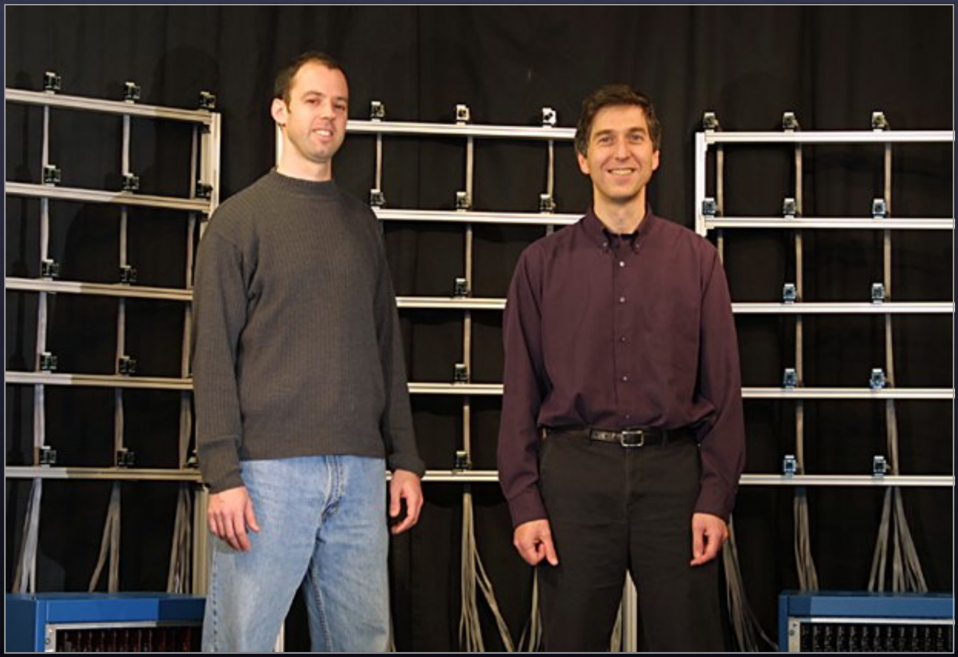
- align the shots
- match histograms
- median filter

Stanford Multi-Camera Array
[Wilburn SIGGRAPH 2005]

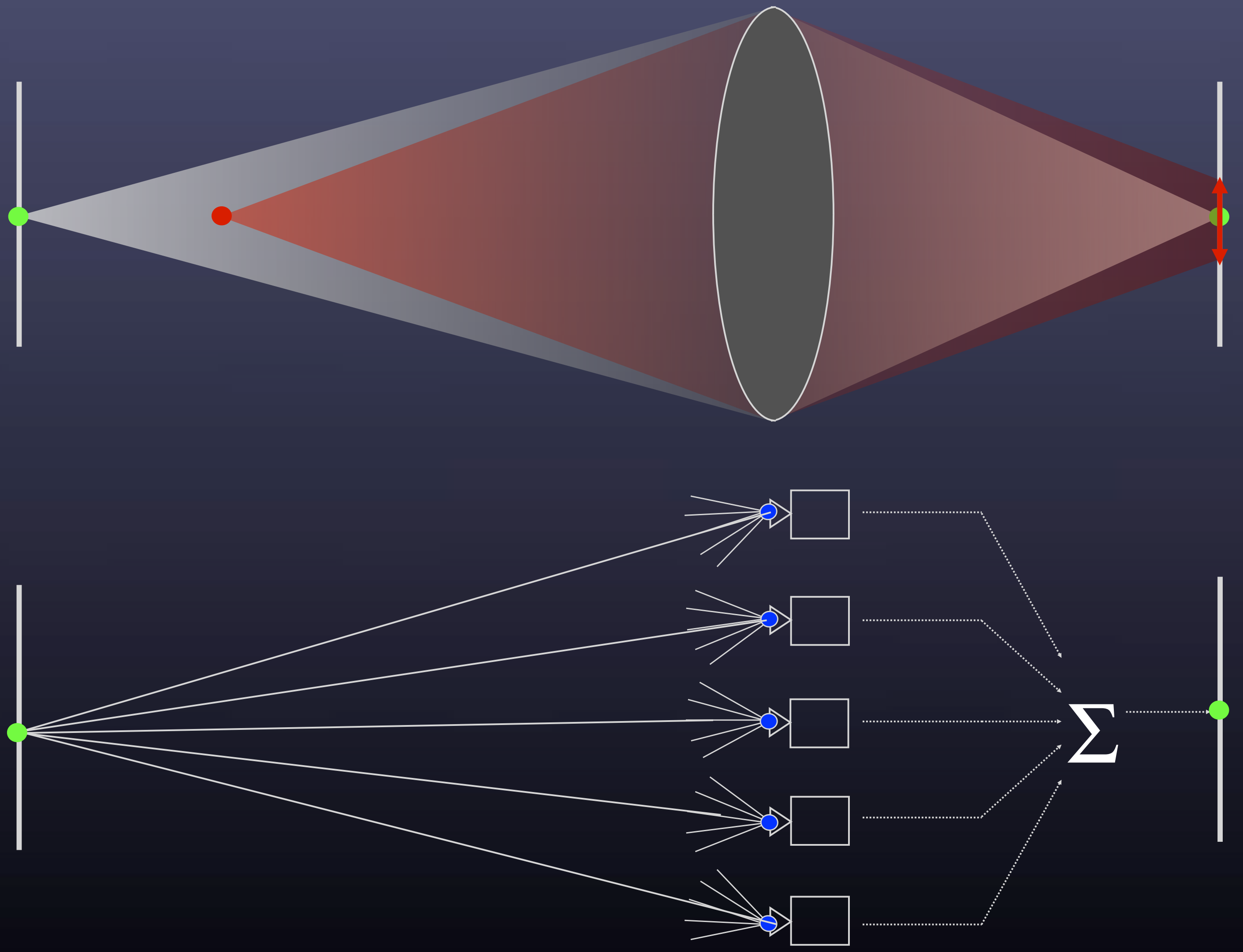
- 640 × 480 pixels × 30 fps × 128 cameras

- synchronized timing
- continuous streaming
- flexible arrangement

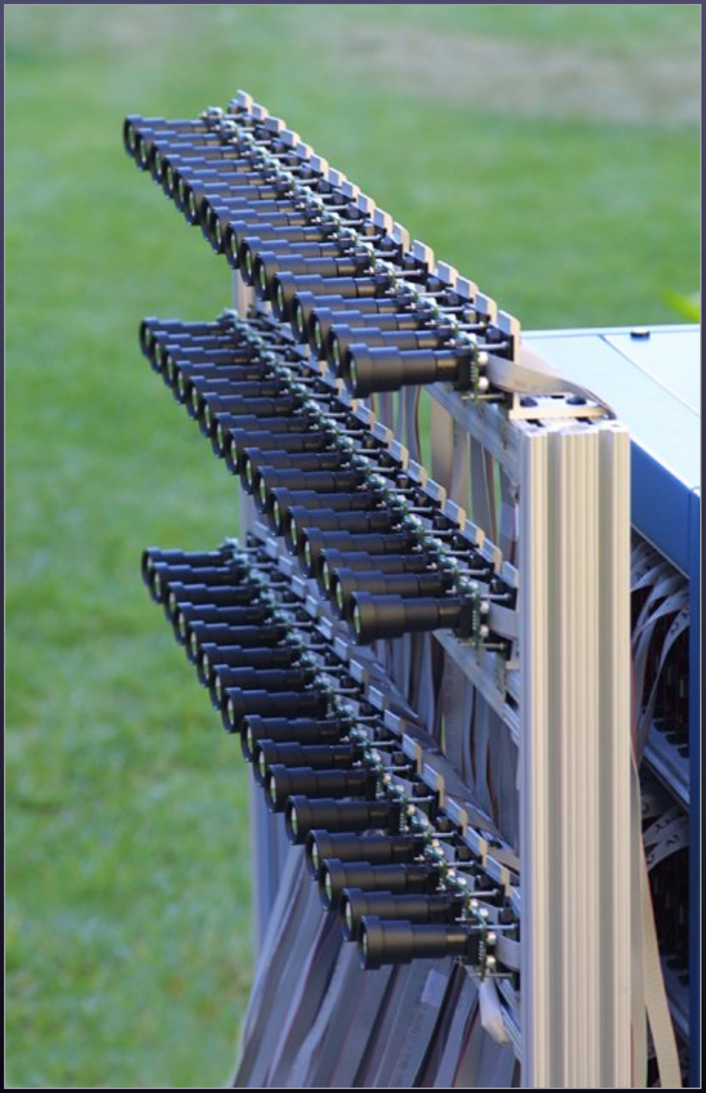


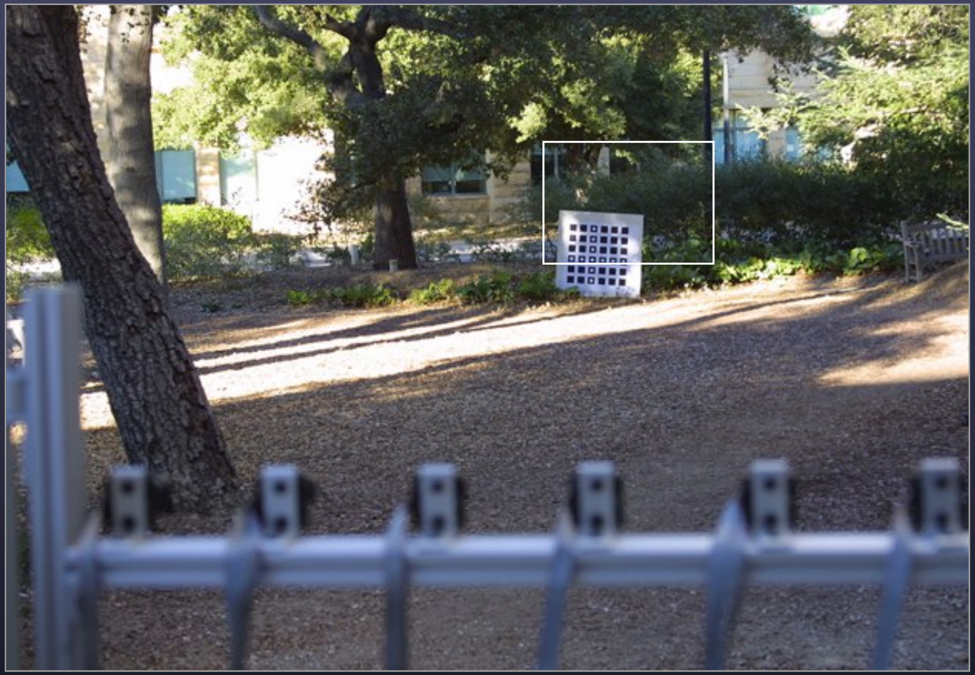


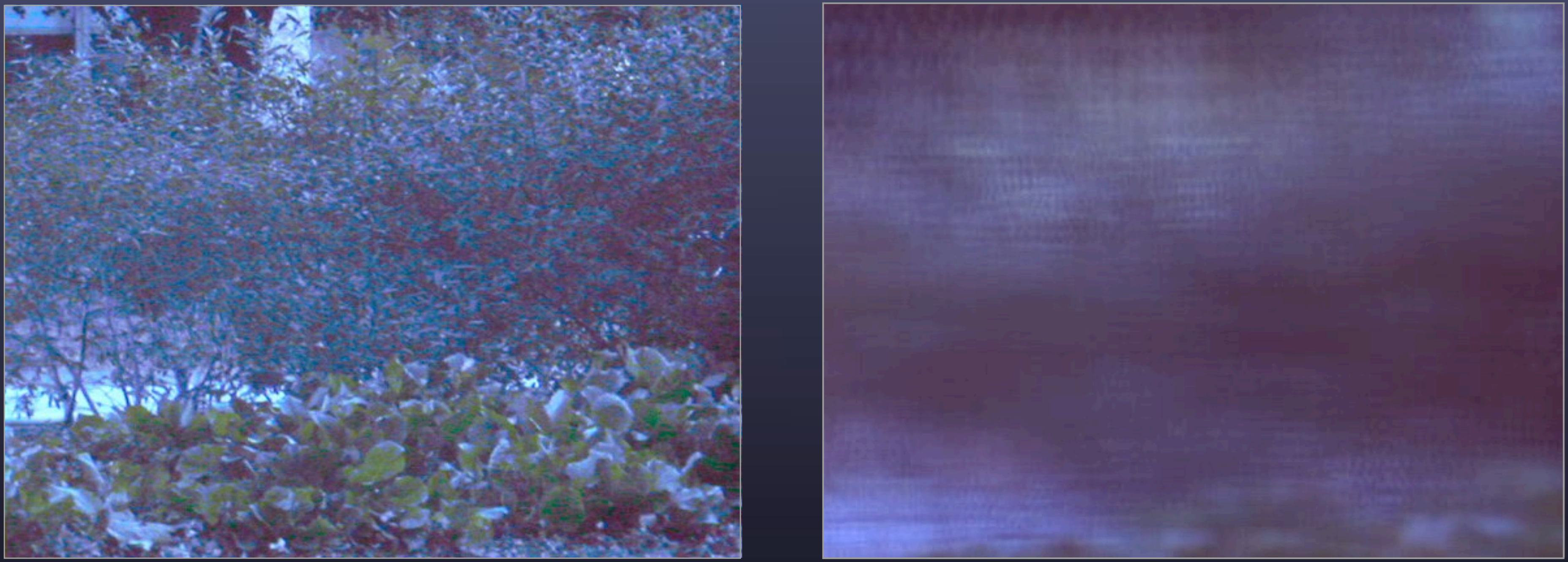
Synthetic aperture photography



Example using 45 cameras
[Vaish CVPR 2004]



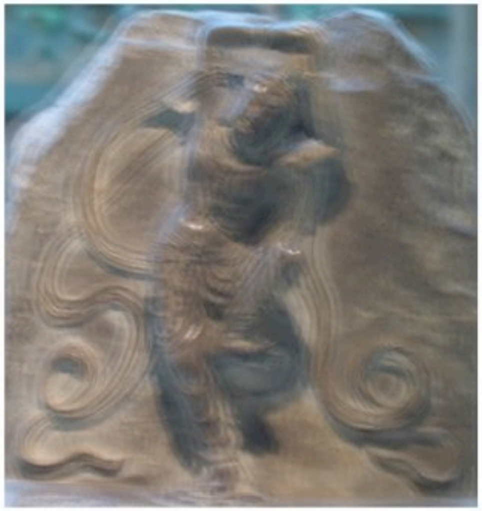




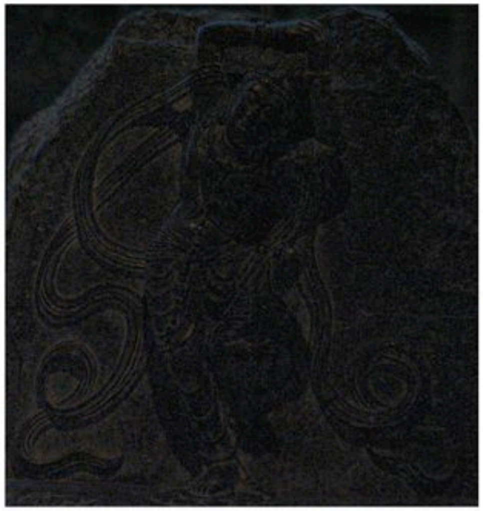
(movie is available at http://graphics.stanford.edu/projects/array)

Removing camera shake (again)

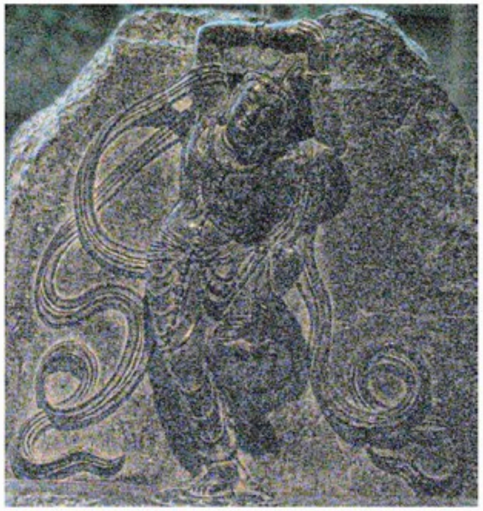
- deconvolve long-exposure (blurred) image, using short-exposure (noisy) image as prior
  [Yuan SIGGRAPH 2007]



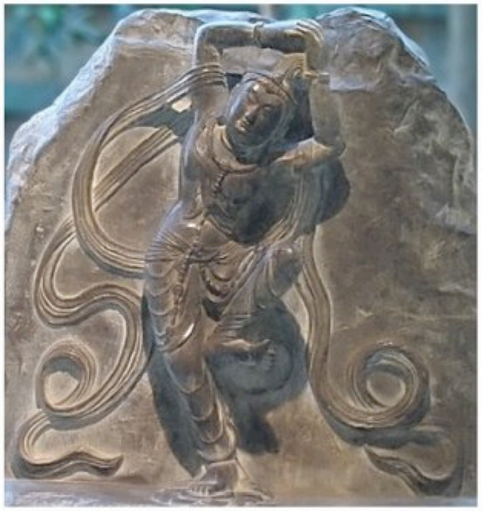
long exposure
(blurry)



short exposure
(dark)



same, scaled up
(noisy)

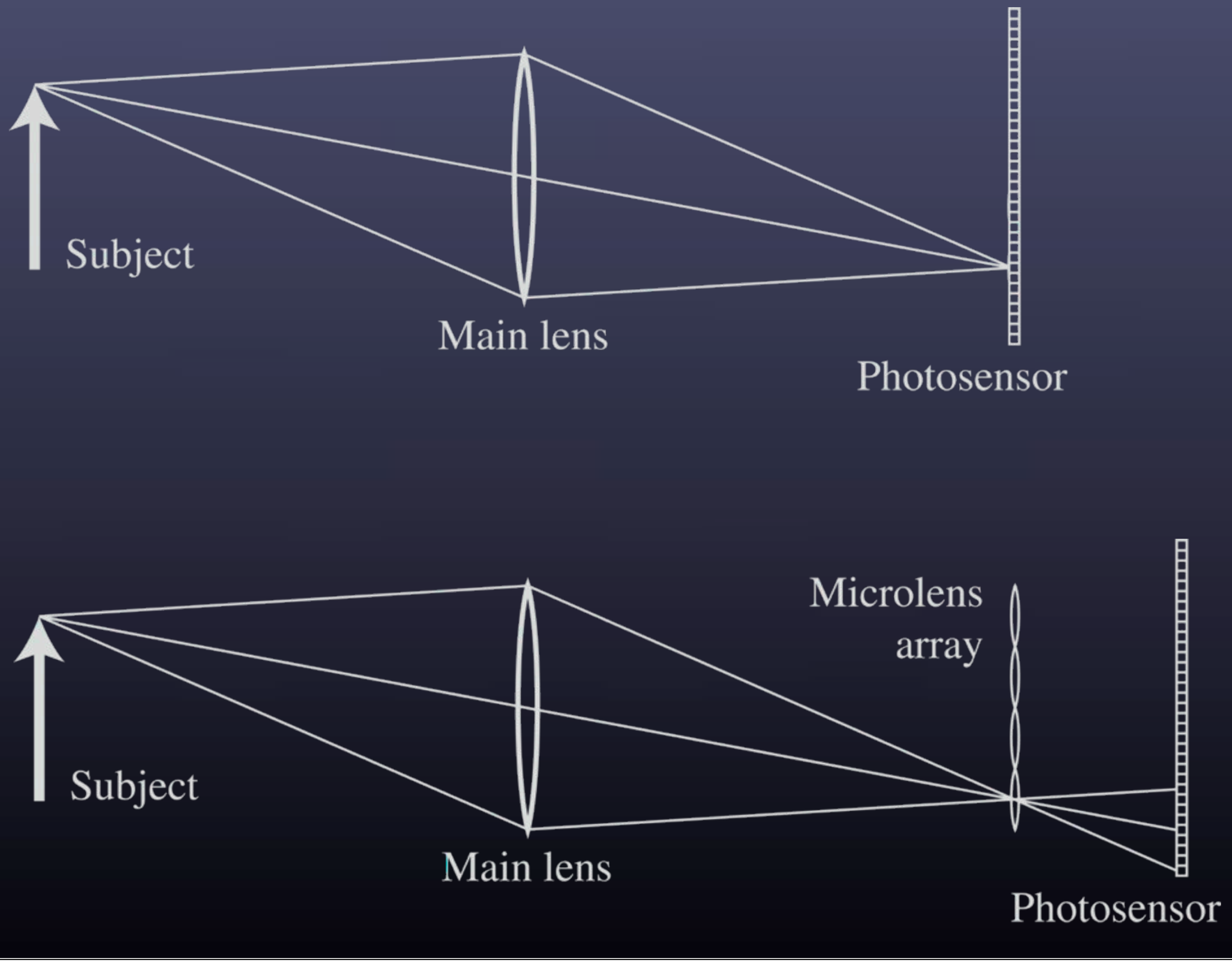


joint deconvolution

Film-like Photography with bits	Computational Photography			
	Computational Camera			Smart Light
Digital Photography	Computational Processing	Computational Imaging/Optics	Computational Sensor	Computational Illumination
Image processing applied to captured images to produce better images.	Processing of a set of captured images to create new images.	Capture of optically coded images and computational decoding to produce new images.	Detectors that combine sensing and processing to create smart pixels.	Adapting and Controlling Illumination to Create revealing image
Examples: Interpolation, Filtering, Enhancement, Dynamic Range Compression, Color Management, Morphing, Hole Filling, Artistic Image Effects, Image Compression, Watermarking.	Examples: Mosaicing, Matting, Super-Resolution, Multi-Exposure HDR, Light Field from Mutiple View, Structure from Motion, Shape from X.	Examples: Coded Aperture, Optical Tomography, Diaphanography, SA Microscopy, Integral Imaging, Assorted Pixels, Catadioptric Imaging, Holographic Imaging.	Examples: Artificial Retina, Retinex Sensors, Adaptive Dynamic Range Sensors, Edge Detect Chips, Focus of Expansion Chips, Motion Sensors.	Examples: Flash/no flash, Lighting domes, Multi-flash for depth edges, Dual Photos, Polynomial texture Maps, 4D light source

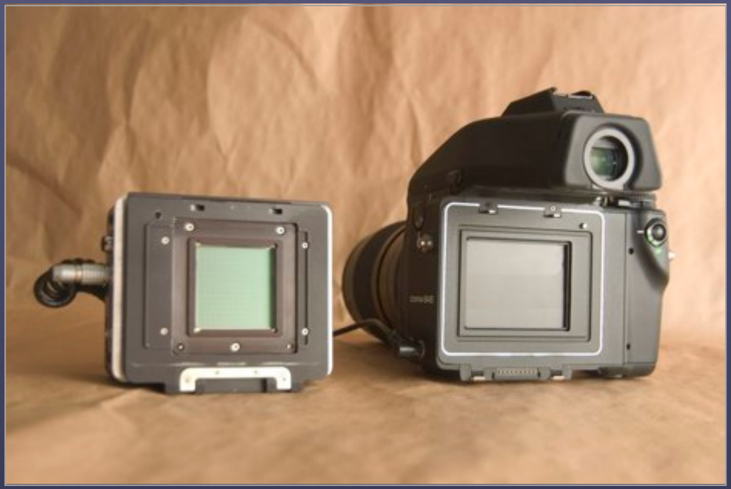
[Nayar, Tumblin]

Light field photography
[Ng SIGGRAPH 2005]

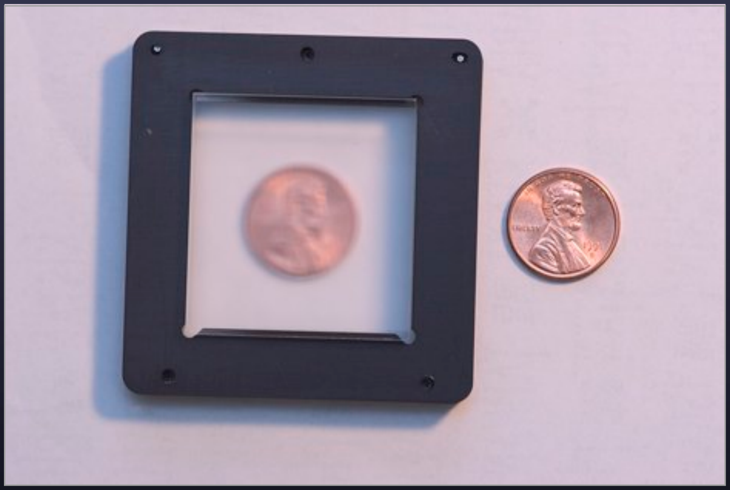


Prototype camera

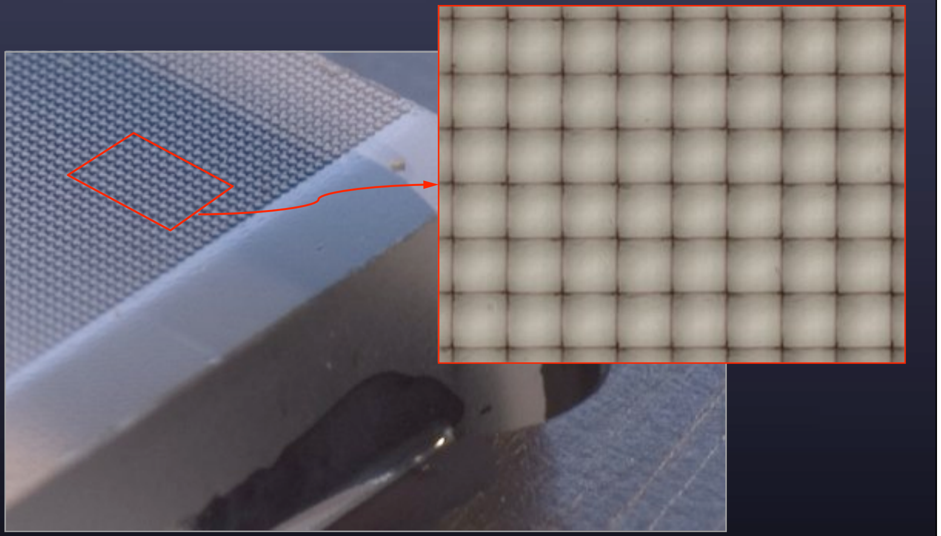
Contax medium format camera



Kodak 16-megapixel sensor

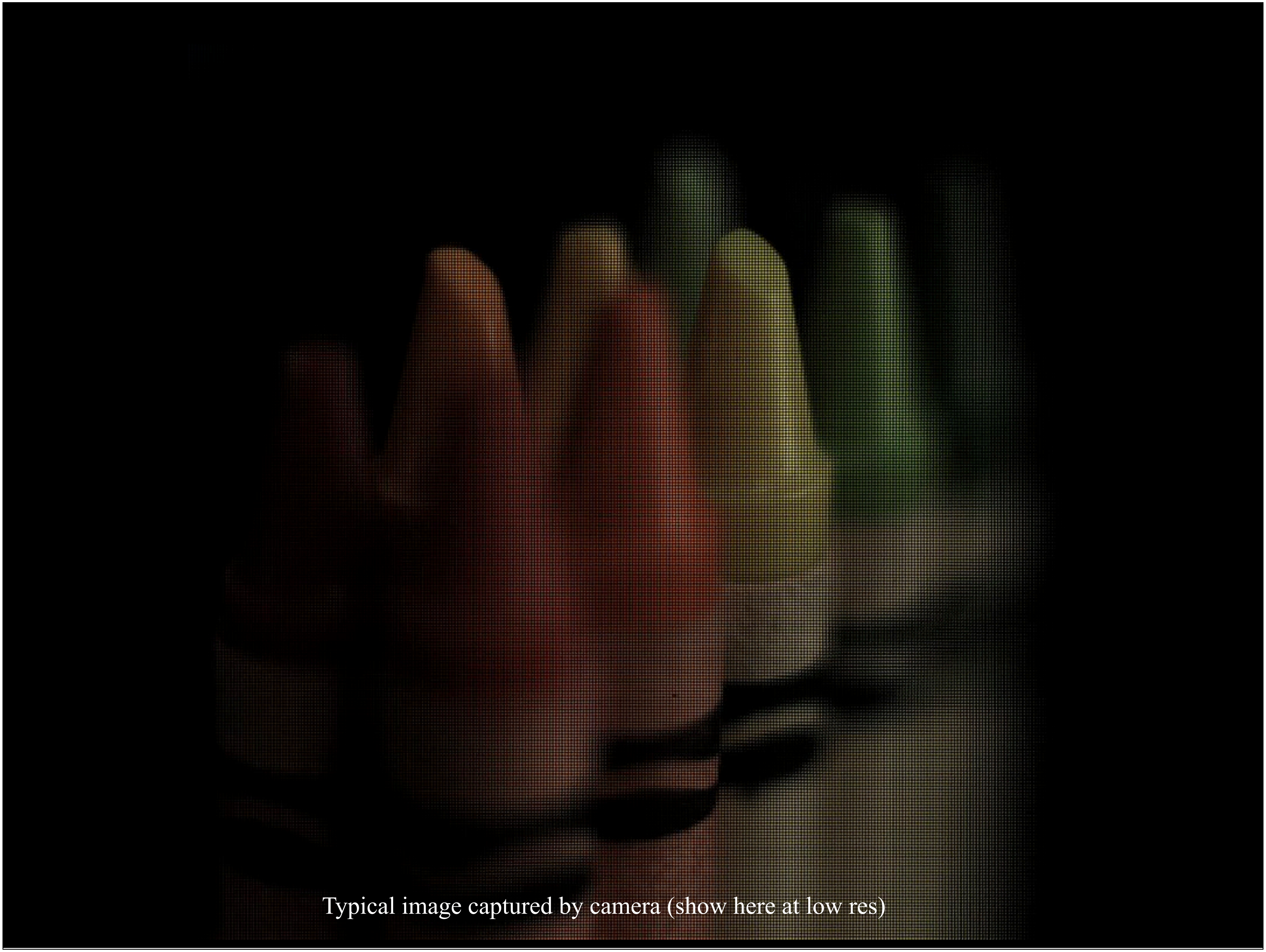


Adaptive Optics microlens array

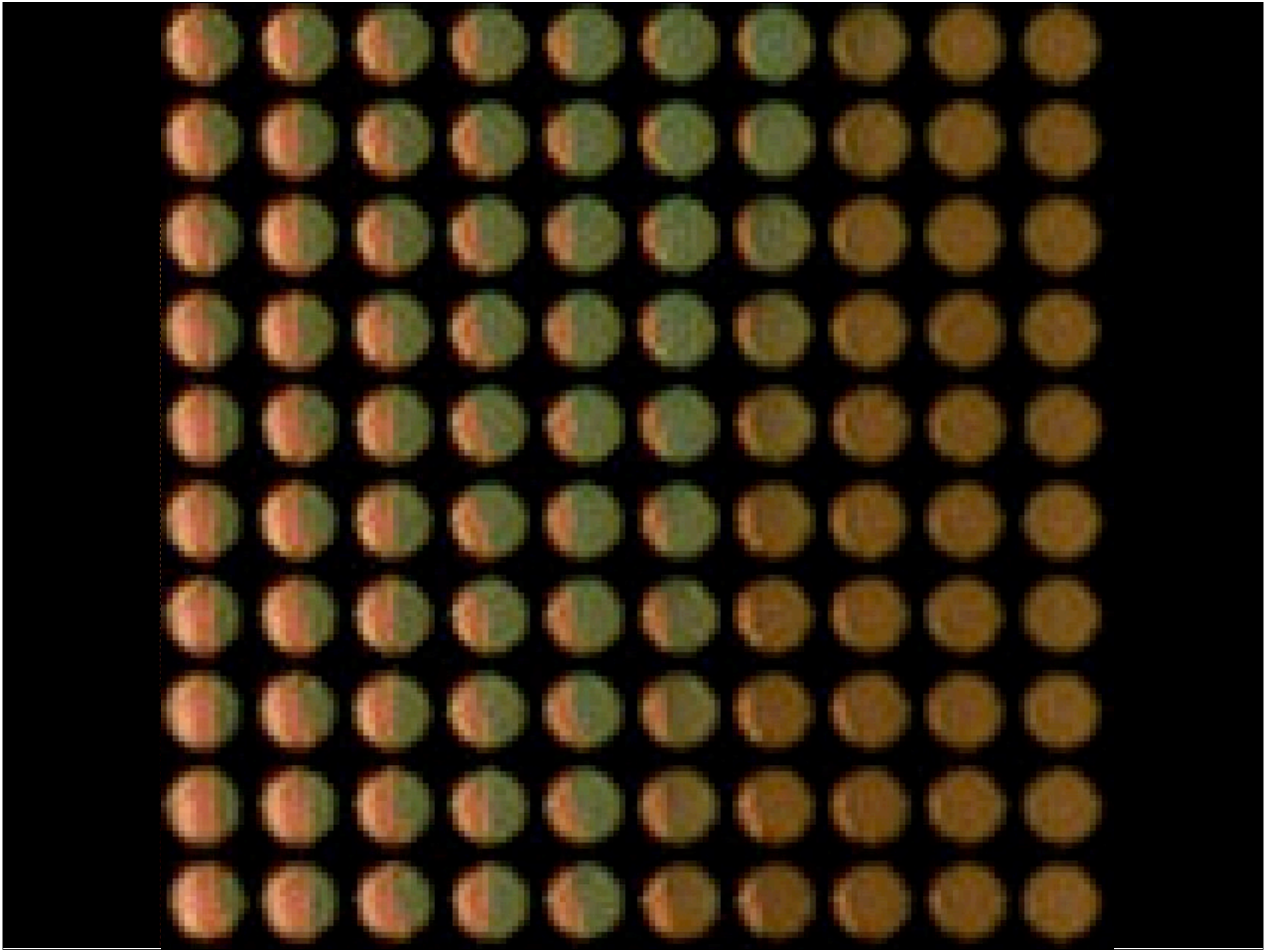


125µ square-sided microlenses

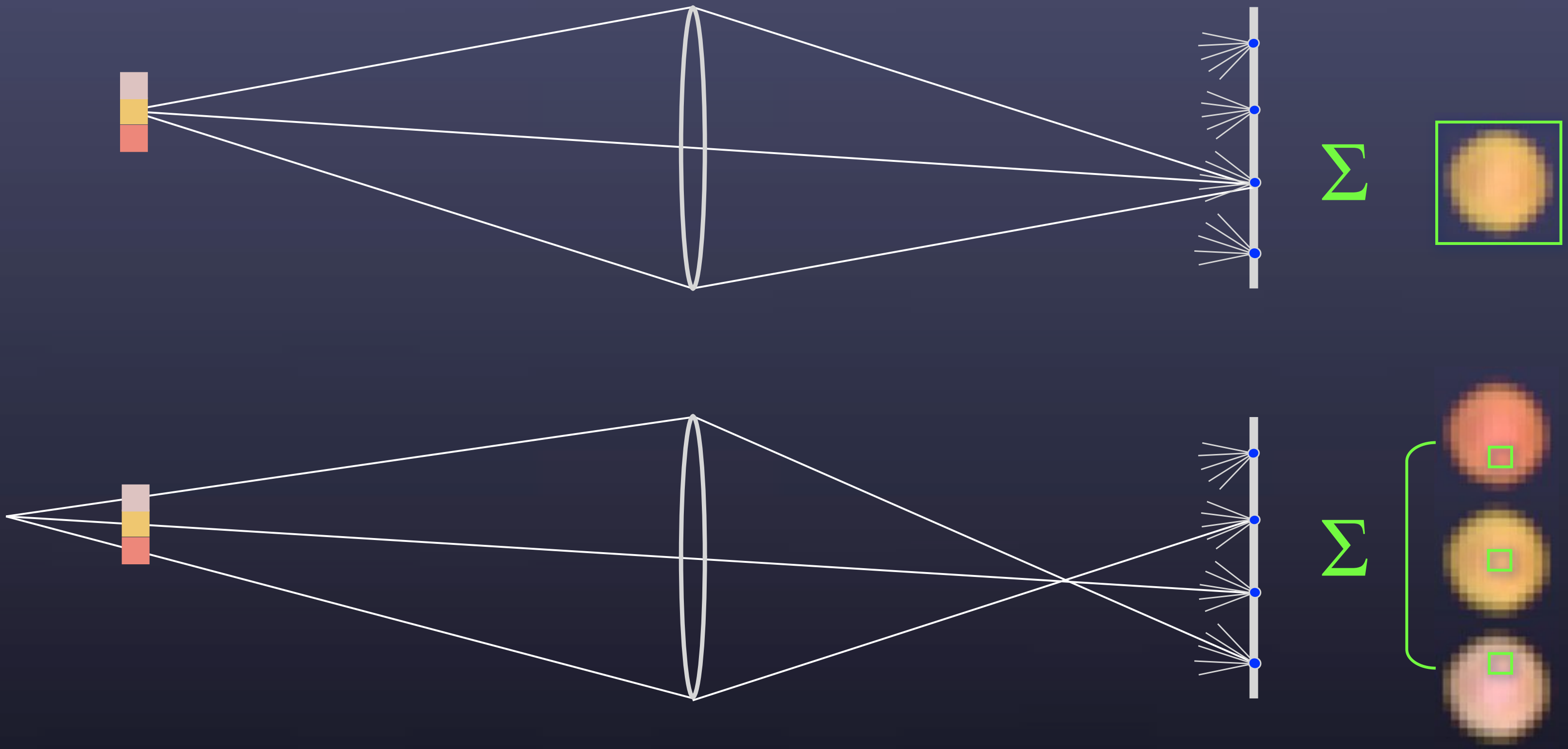
4000 × 4000 pixels ÷ 292 × 292 lenses = 14 × 14 pixels per lens



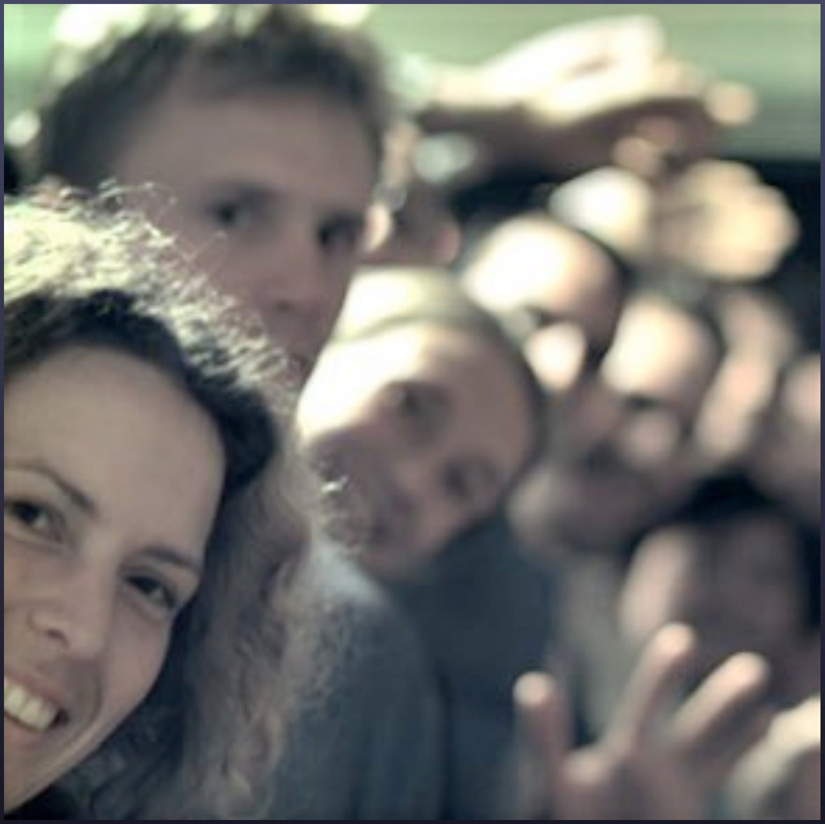
Typical image captured by camera (show here at low res)



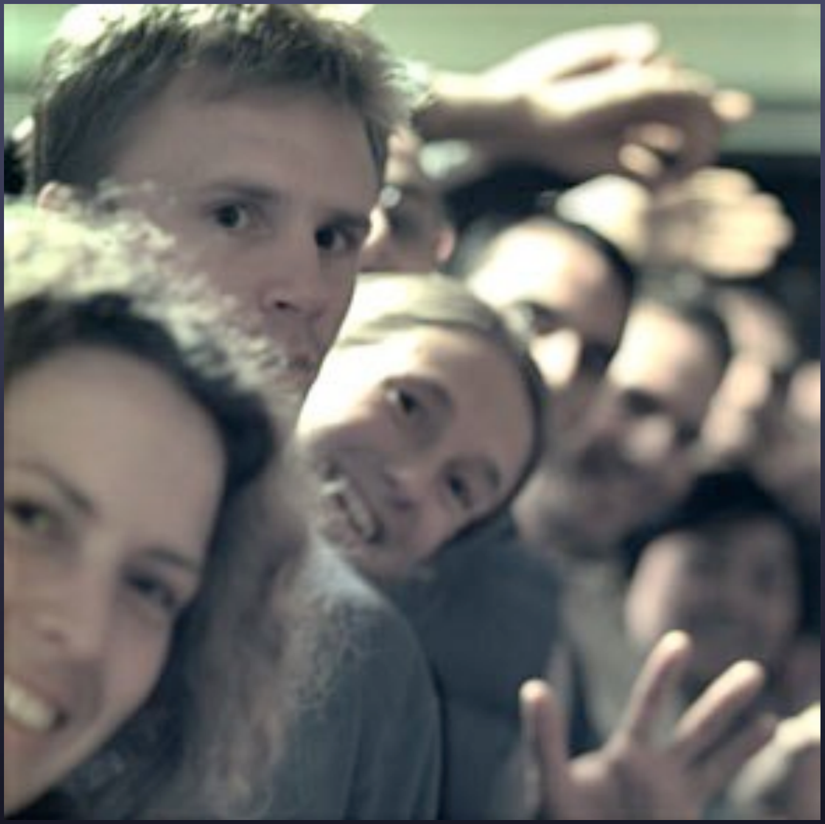
Digital refocusing



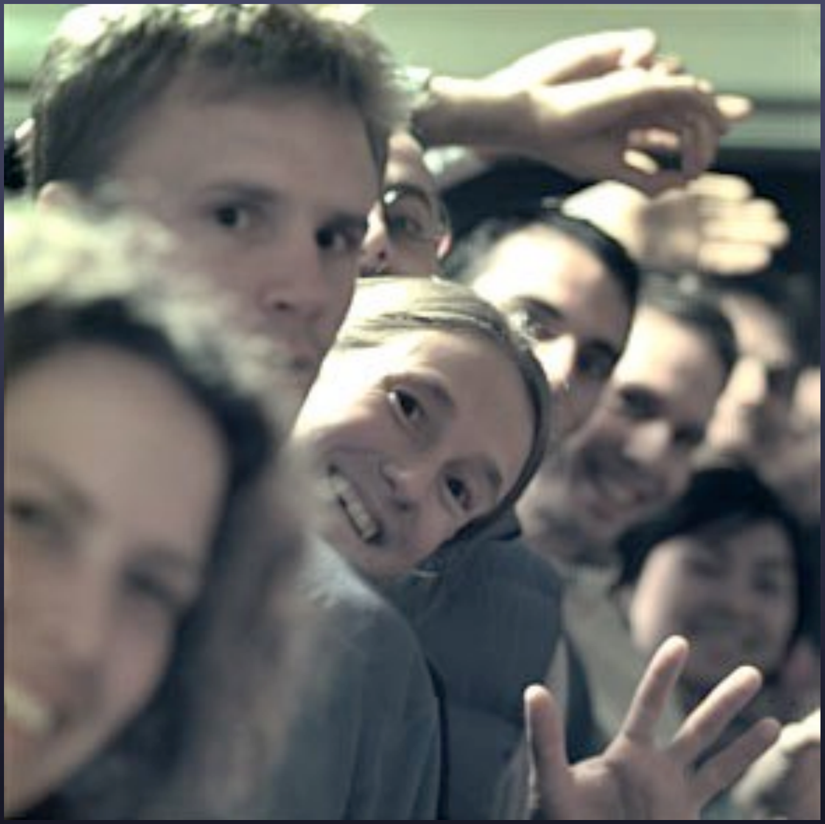
Example of digital refocusing



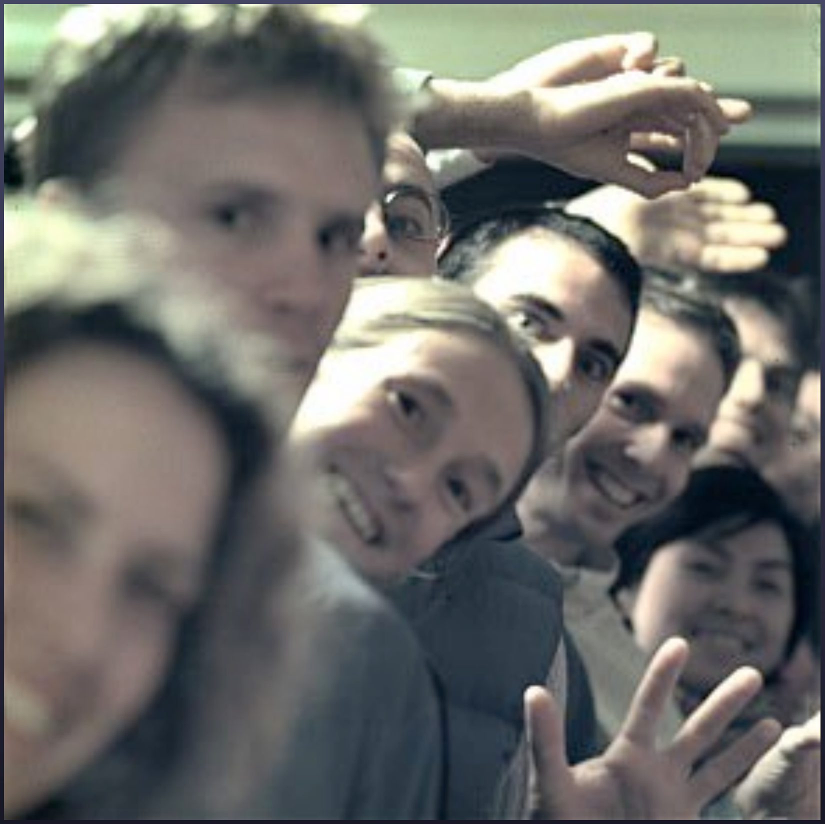
Example of digital refocusing



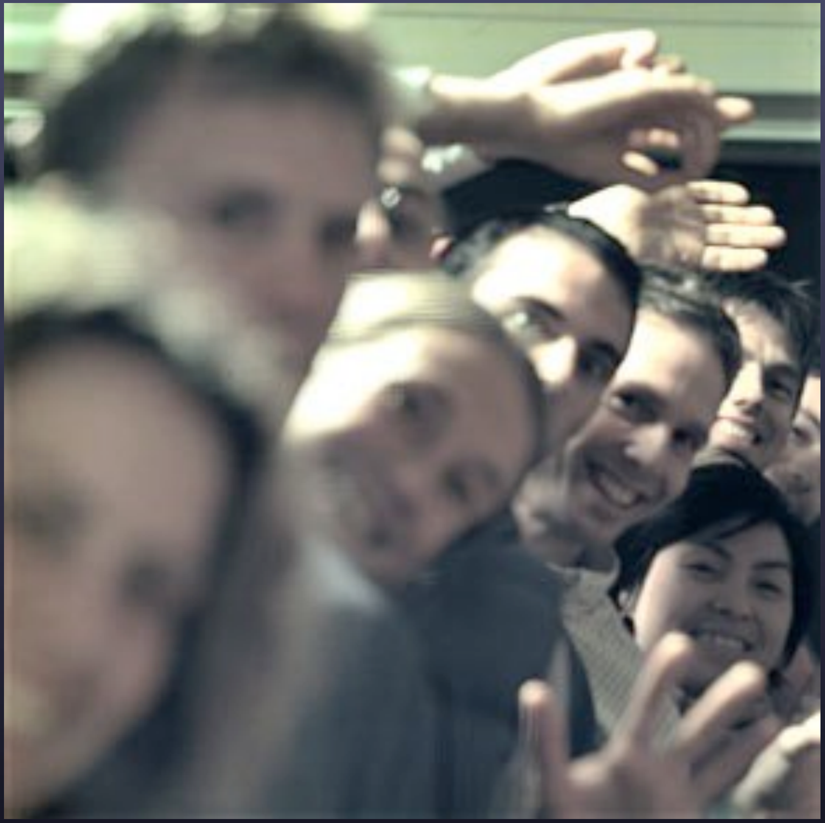
Example of digital refocusing



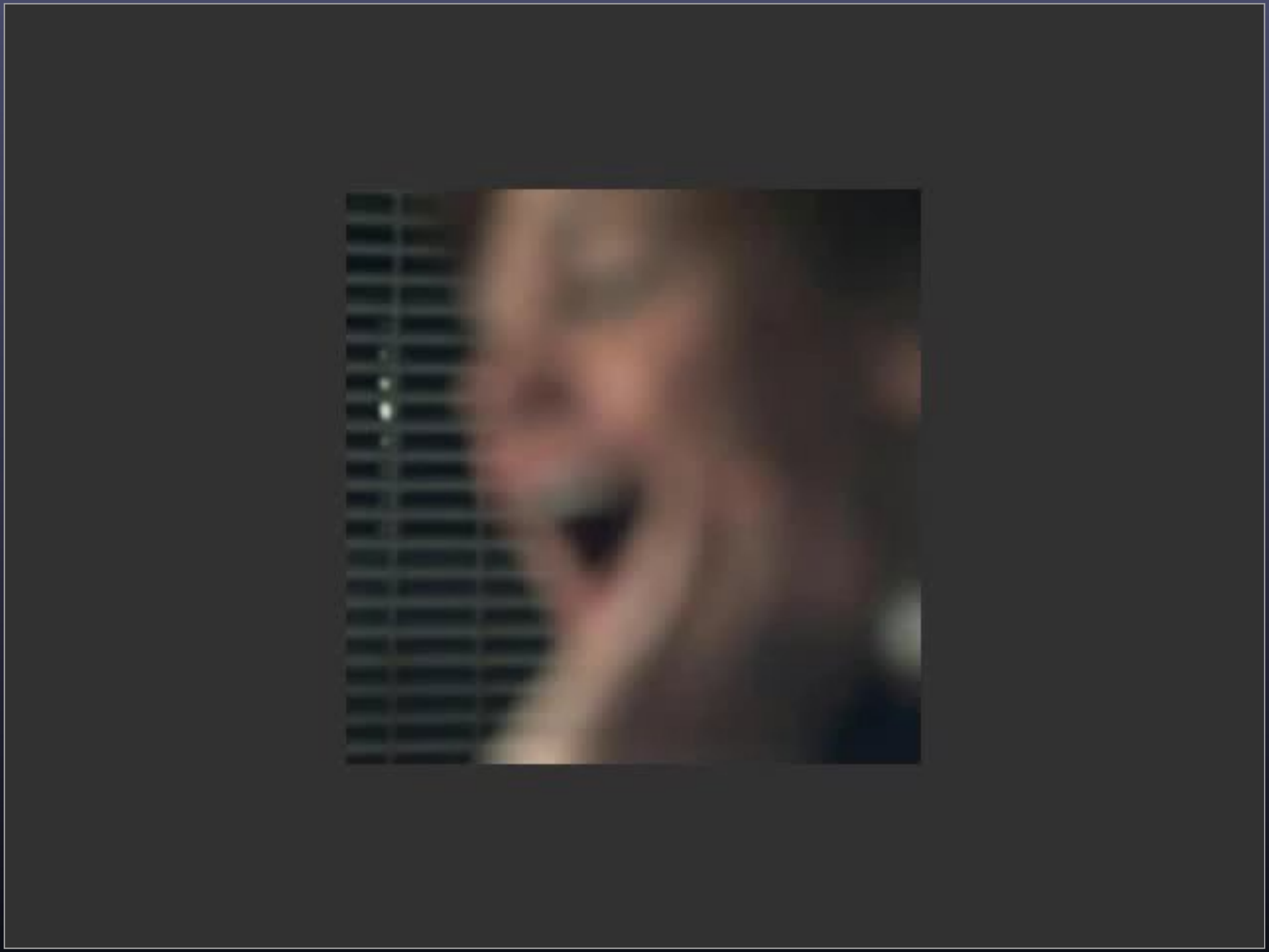
Example of digital refocusing



Example of digital refocusing



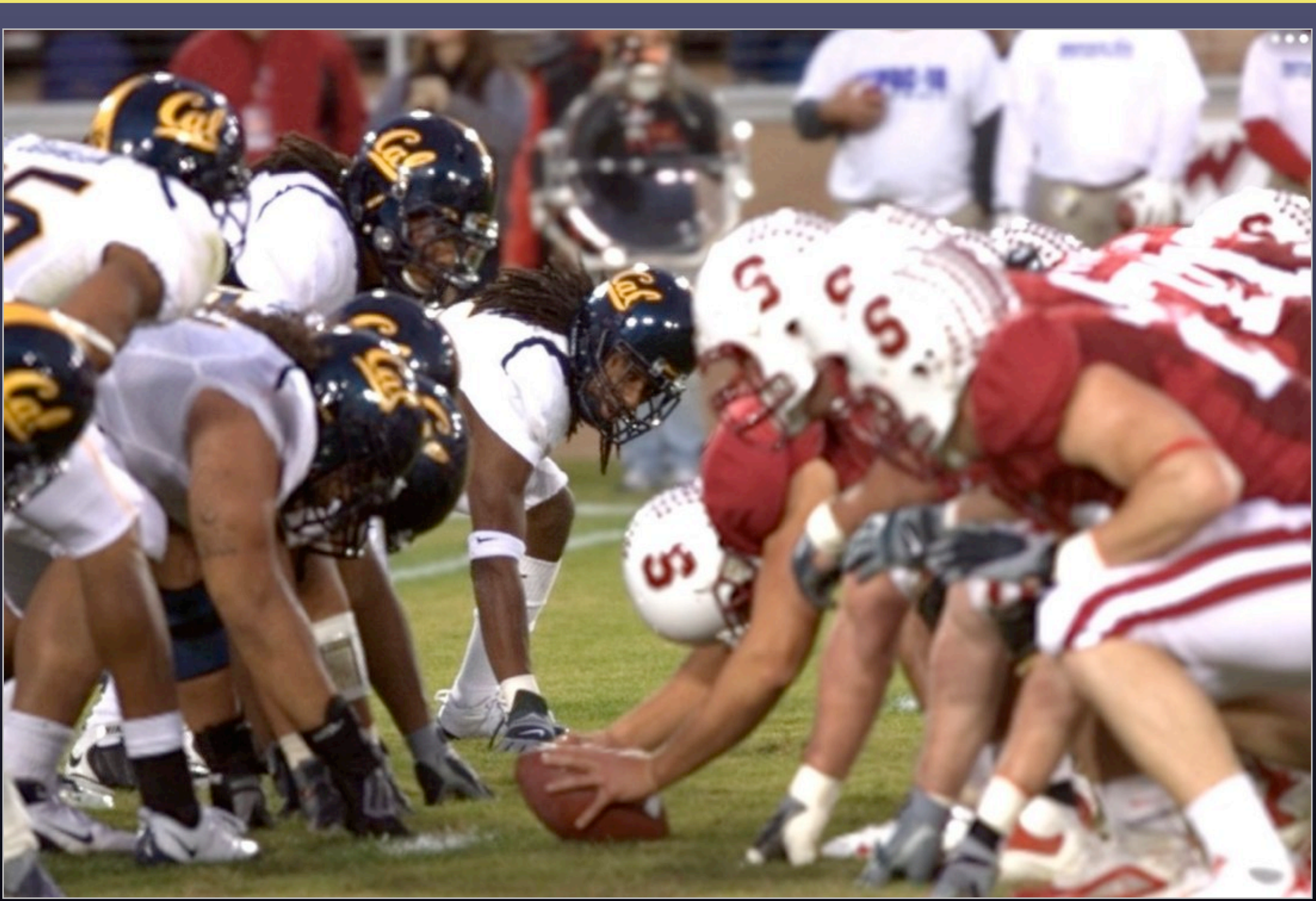
Refocusing portraits



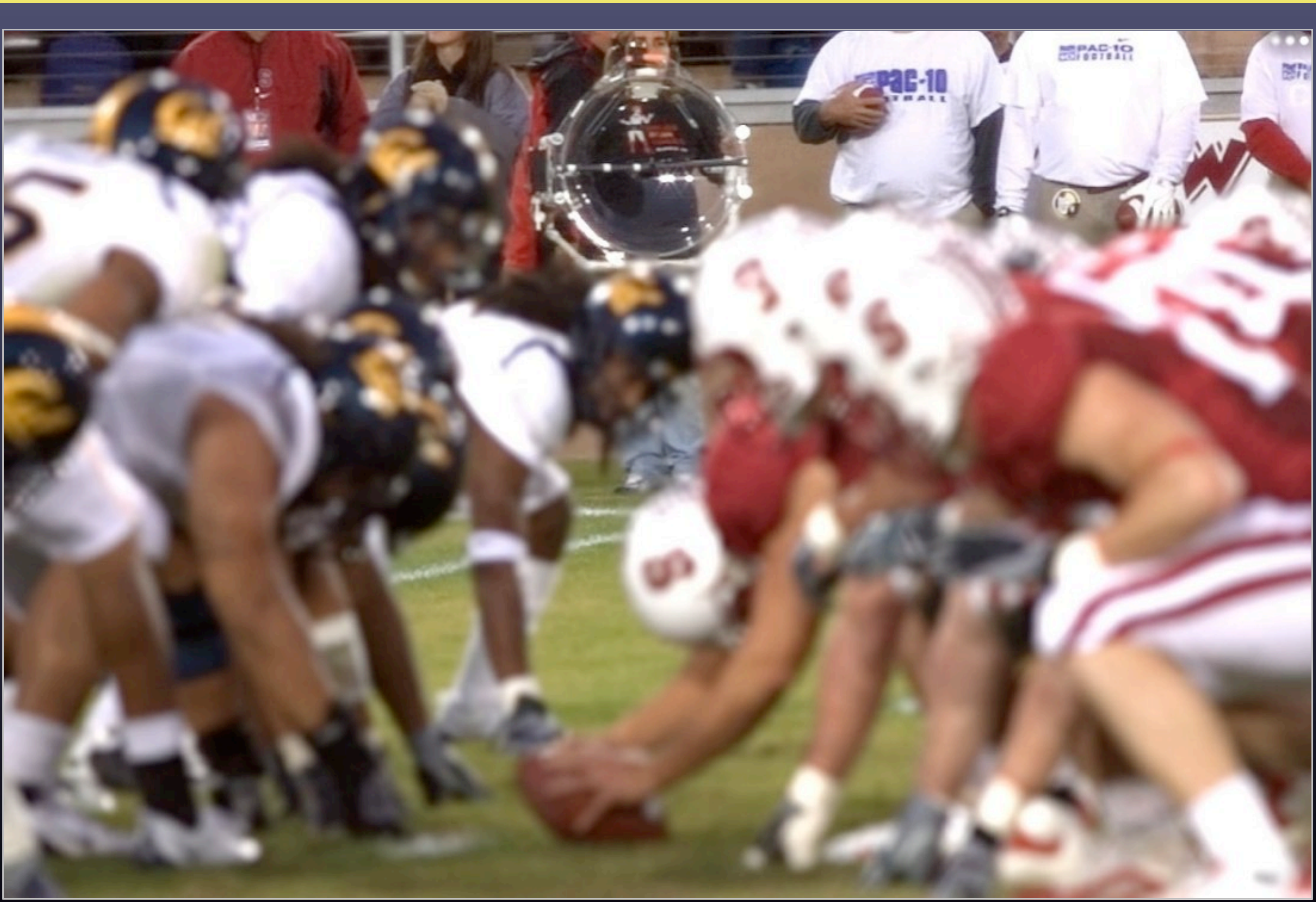
(movie is available at http://refocusimaging.com)

Application to sports photography

Application to sports photography



Application to sports photography

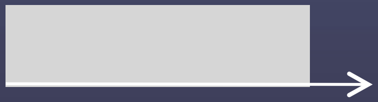


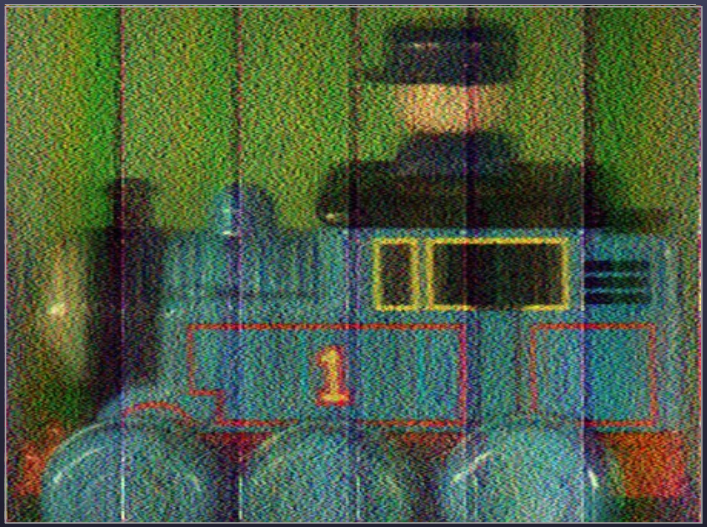
Film-like Photography with bits	Computational Photography			
	Computational Camera			Smart Light
Digital Photography	Computational Processing	Computational Imaging/Optics	Computational Sensor	Computational Illumination
Image processing applied to captured images to produce better images.	Processing of a set of captured images to create new images.	Capture of optically coded images and computational decoding to produce new images.	Detectors that combine sensing and processing to create smart pixels.	Adapting and Controlling Illumination to Create revealing image
Examples: Interpolation, Filtering, Enhancement, Dynamic Range Compression, Color Management, Morphing, Hole Filling, Artistic Image Effects, Image Compression, Watermarking.	Examples: Mosaicing, Matting, Super-Resolution, Multi-Exposure HDR, Light Field from Mutiple View, Structure from Motion, Shape from X.	Examples: Coded Aperture, Optical Tomography, Diaphanography, SA Microscopy, Integral Imaging, Assorted Pixels, Catadioptric Imaging, Holographic Imaging.	Examples: Artificial Retina, Retinex Sensors, Adaptive Dynamic Range Sensors, Edge Detect Chips, Focus of Expansion Chips, Motion Sensors.	Examples: Flash/no flash, Lighting domes, Multi-flash for depth edges, Dual Photos, Polynomial texture Maps, 4D light source

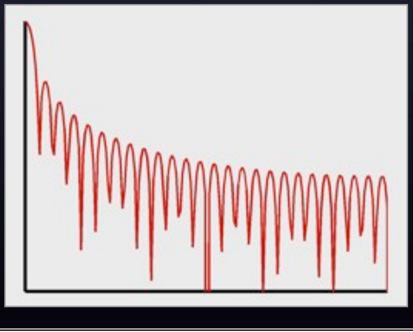
[Nayar, Tumblin]

Coded-exposure photography
[Raskar SIGGRAPH 2006]

continuous shutter



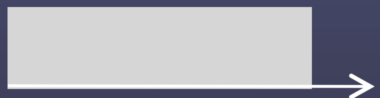


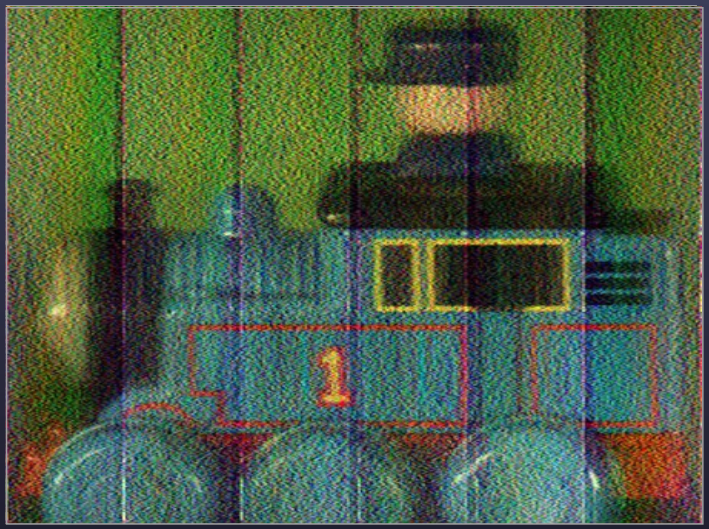


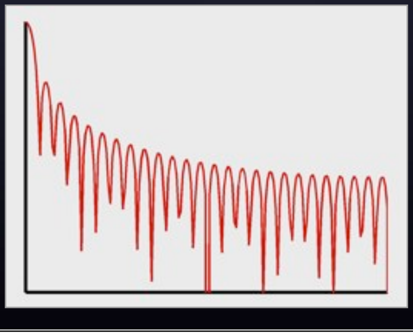
Coded-exposure photography

[Raskar SIGGRAPH 2006]

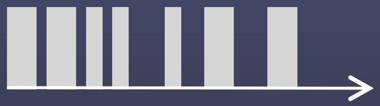
continuous shutter

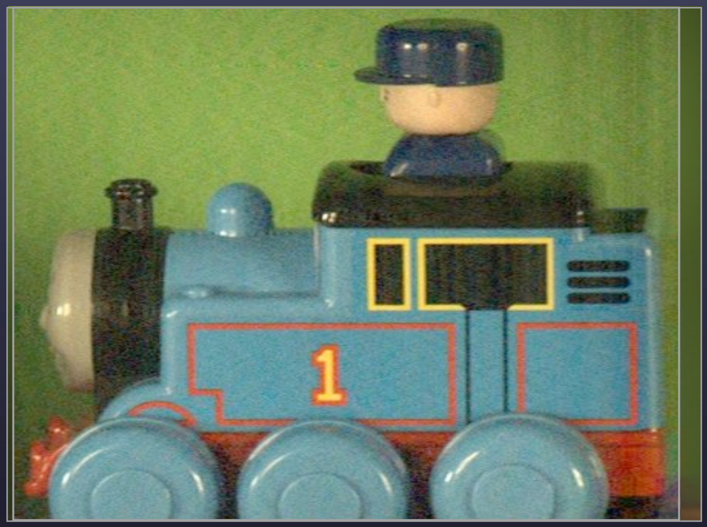


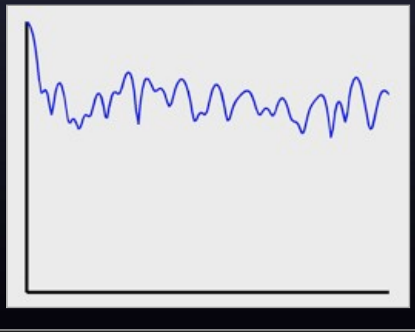




fluttered shutter







Computational Photography

Film-like Photography with bits

Digital Photography

Image processing applied to captured images to produce better images.

Examples:
Interpolation, Filtering, Enhancement, Dynamic Range Compression, Color Management, Morphing, Hole Filling, Artistic Image Effects, Image Compression, Watermarking.

Computational Camera

Computational Processing

Processing of a set of captured images to create new images.

Examples:
Mosaicing, Matting, Super-Resolution, Multi-Exposure HDR, Light Field from Mutiple View, Structure from Motion, Shape from X.

Computational Imaging/Optics

Capture of optically coded images and computational decoding to produce new images.

Examples:
Coded Aperture, Optical Tomography, Diaphanography, SA Microscopy, Integral Imaging, Assorted Pixels, Catadioptric Imaging, Holographic Imaging.

Computational Sensor

Detectors that combine sensing and processing to create smart pixels.

Examples:
Artificial Retina, Retinex Sensors, Adaptive Dynamic Range Sensors, Edge Detect Chips, Focus of Expansion Chips, Motion Sensors.

Smart Light

Computational Illumination

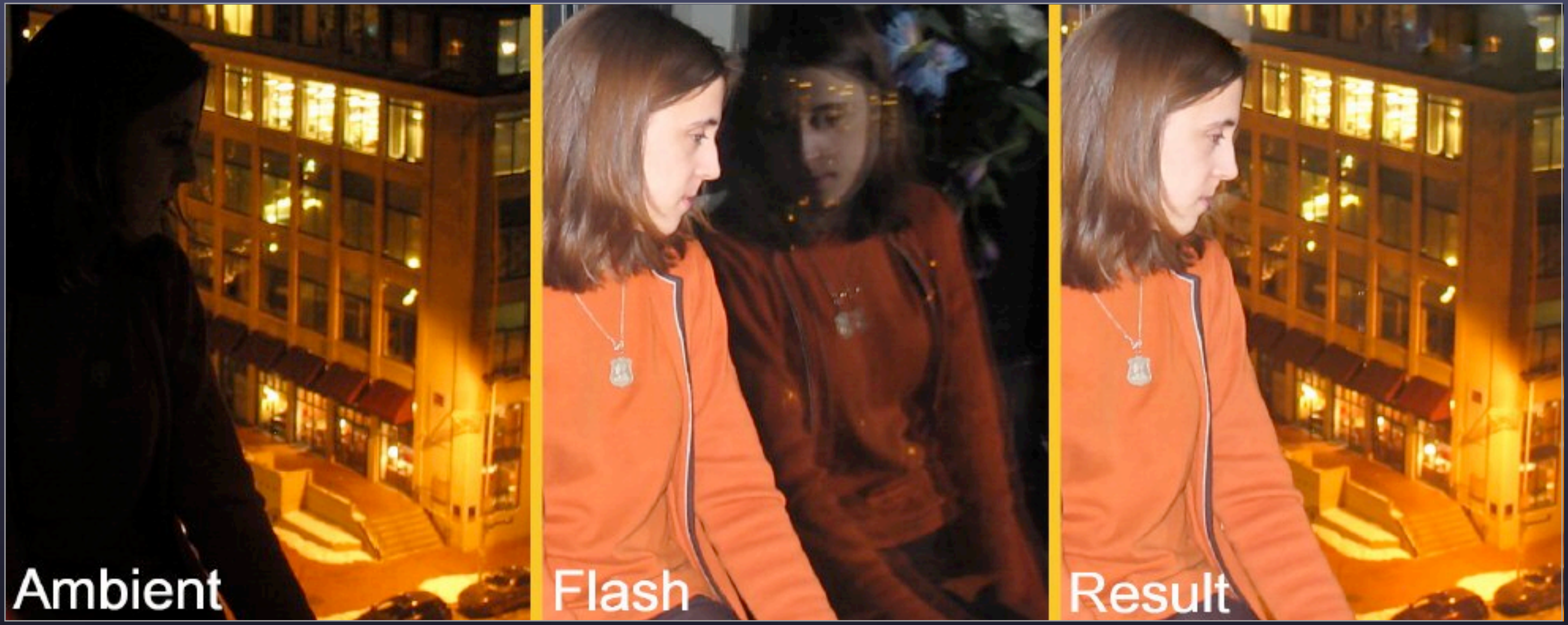
Adapting and Controlling Illumination to Create revealing image

Examples:
Flash/no flash, Lighting domes, Multi-flash for depth edges, Dual Photos, Polynomial texture Maps, 4D light source

[Nayar Tumblin]

Flash-noflash photography

[Agrawal SIGGRAPH 2005]



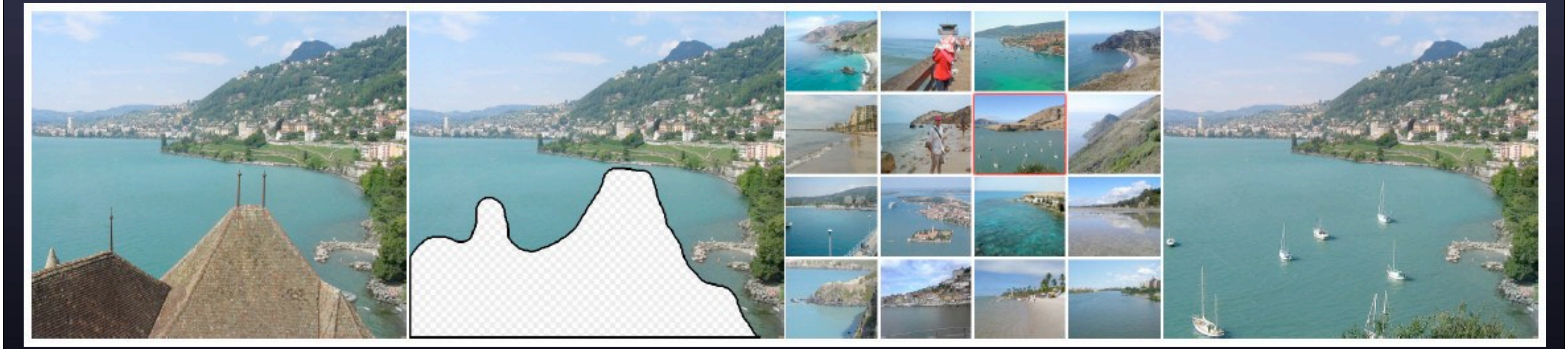
- compute ambient + flash – features in sum that don’t appear in ambient alone (as determined from image gradients) (except where ambient image is nearly black)

Large online photo collections

- Facebook
  - –3 billion photos
- Flickr
  - –9 billion photos
- Google Library Project
  - –50 million books × 300 pages = 15 billion images
- Google Earth
- Google StreetView Project
  - –formerly the Stanford CityBlock Project

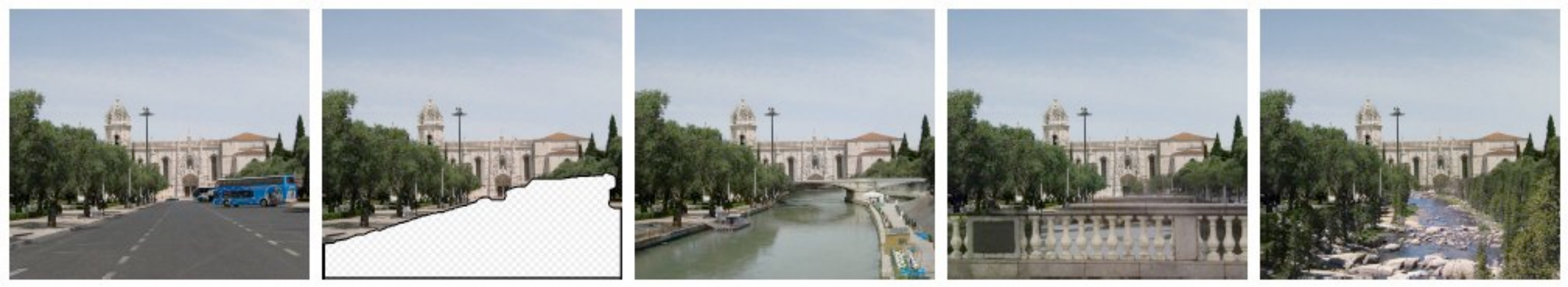
Scene completion using millions of photographs
[Hays & Efros SIGGRAPH 2007]

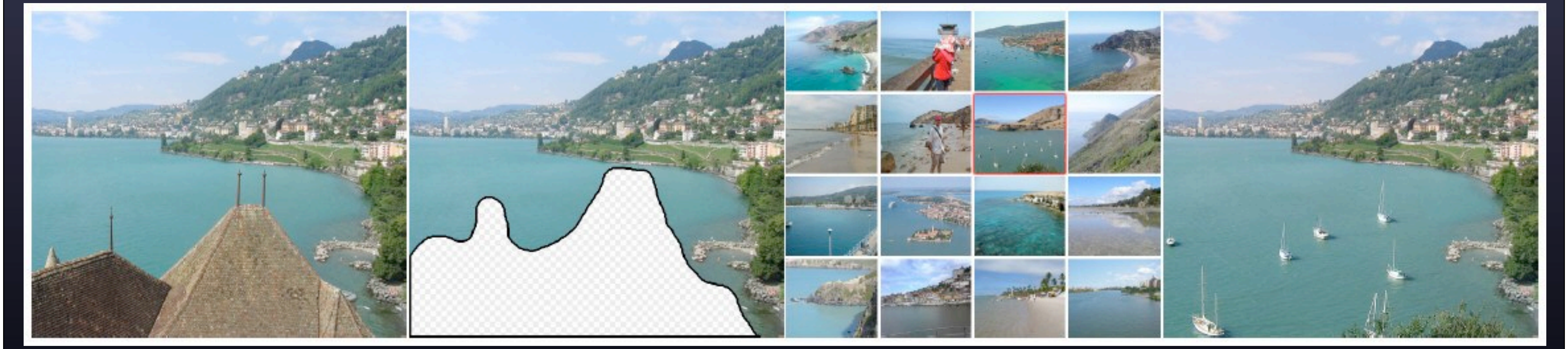
- search for matches from a large database
- Find least visible seams using graph-cut algorithm
- blend gradients & integrate to create image



Scene completion using millions of photographs

[Hays & Efros SIGGRAPH 2007]





What’s wrong with this picture?

- many of these techniques require modifying the camera
  - coded-exposure

- some of these techniques could use help from the camera
  - metering for HDR

- none of these ideas are finding their way into consumer cameras...

☹

Why are traditional camera makers not embracing computational photography?

(soapbox mode ON)

- the camera industry is secretive
  - no flow of workers between companies and universities
  - few publications or open source standards
- camera companies sell hardware, not software
  - many are not comfortable with Internet ecosystems
- some computational techniques are still not robust
  - partly because researchers can’t test them in the field

(soapbox mode OFF)

Stanford Camera 2.0 Project

The Walt Disney Company

NOKIA Connecting People

Adobe

Kodak

hp invent

Programmable cameras

Canon 5D Mark II

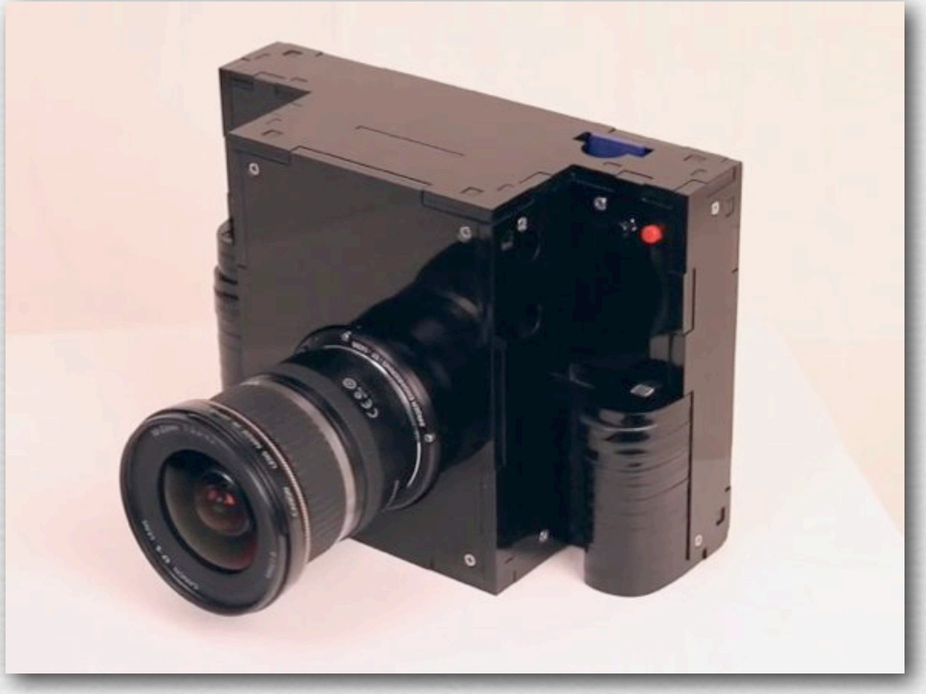
- SLR camera SDKs
  - treats camera as black box
- scriptable cameras
  - Kodak DC2XX
  - HP PhotoSmart CXX
  - Canon Hack Development Kit (CHDK) & Magic Lantern
  - still treats metering, focusing, post-processing as black boxes
- Elphel
  - runs Linux
  - limited power & extensibility
- machine vision cameras
  - like Elphel, these are not a complete photographic camera

Unretouched pictures from Nokia N95

(5 megapixels, Zeiss lens, auto-focus)

The Stanford Frankencamera(s)

[Adams SIGGRAPH 2010]



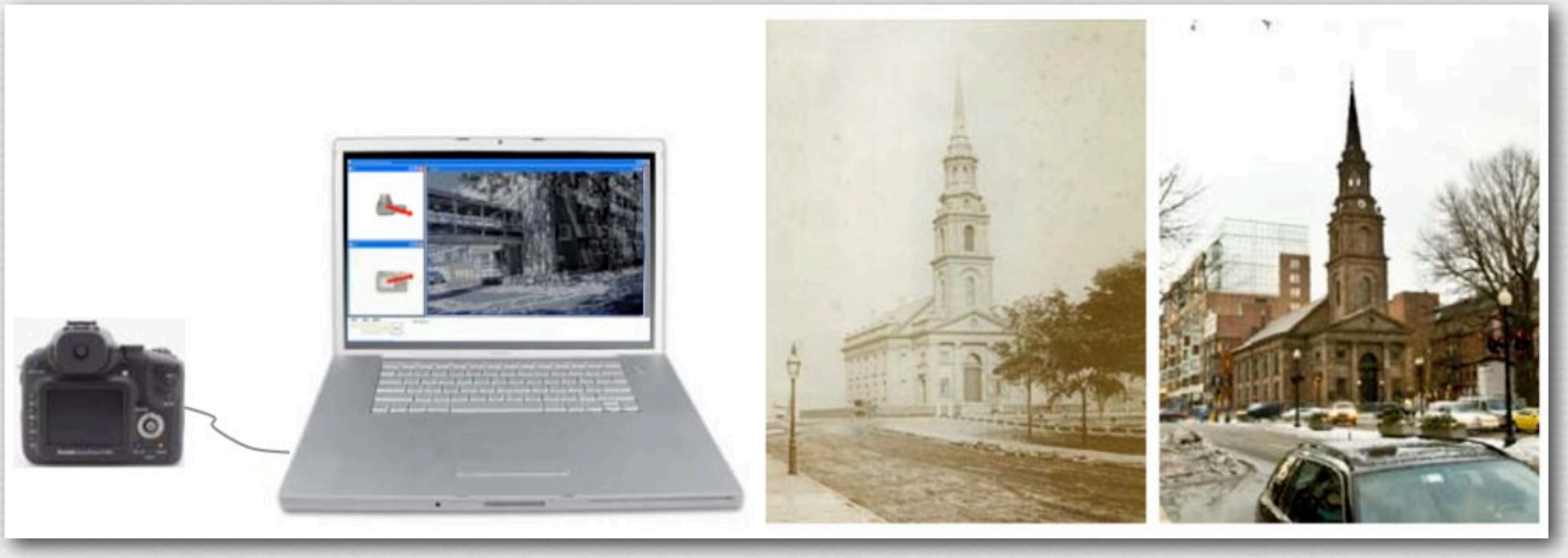
Frankencamera F2

Nokia N900 “F”

- facilitate research in experimental computational photography
- for students in computational photography courses worldwide
- proving ground for plugins and apps for future cameras

What should an open-source camera do?

- handheld and self-powered
  - not tethered to a laptop in a backpack



Example app: re-photography (courtesy of Fredo Durand)

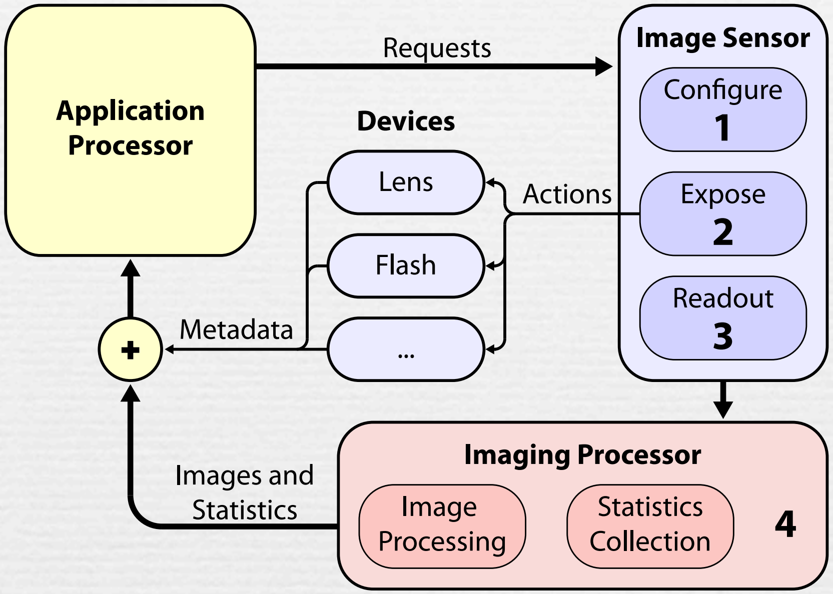
What should an open-source camera do?

- handheld and self-powered
  - not tethered to a laptop in a backpack
- a photographer's camera
  - SLR-quality sensor and lenses
  - LCD viewfinder with multi-touch screen
- ability to manipulate sensor, lens, and camera settings
  - with synchronization, so we know settings for each frame
  - no interruption of video stream, even if settings changes
- fully programmable
  - register/instruction-level access to all hardware at µsec granularity
  - access to raw pixels (before demosaicing or compression)
  - easy to program

(continued)

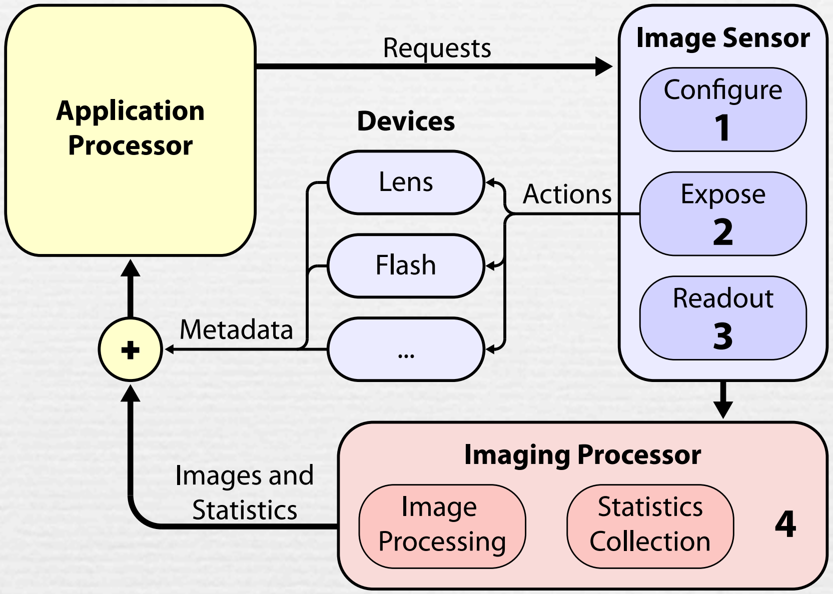
- enough memory to store a burst in the camera
- enough computing power to process a burst quickly using algorithms from the computational photography literature
- connectivity
  - wired to desktop
  - wireless to Internet
  - peer-to-peer (to other cameras)
- physical extensibility
  - multiple flash units, GPS
  - filters, masks, microlens arrays, at aperture or field planes
  - additional user interface widgets
- roadmap that includes existing or feasible commercial products

Frankencamera architecture



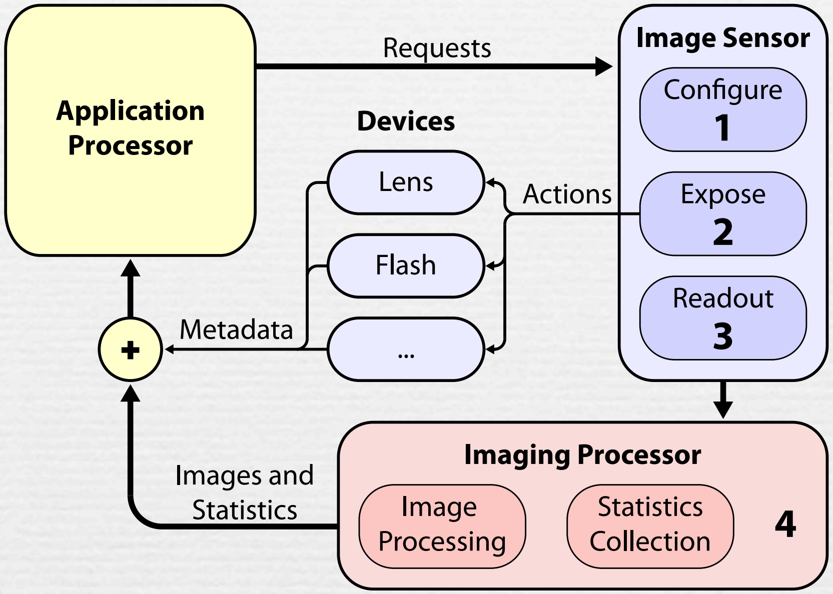
- image sensor is stateless and inaccessible
- instead, a *pipeline* converts requests into frames
- a *request* includes all settings (exposure, ISO, zoom, focus, white balance, resolution, region of interest, flash) for one frame
- a returned *frame* contains an image, some statistics (histogram, sharpness map), and the settings used to capture that frame

Imaging processor



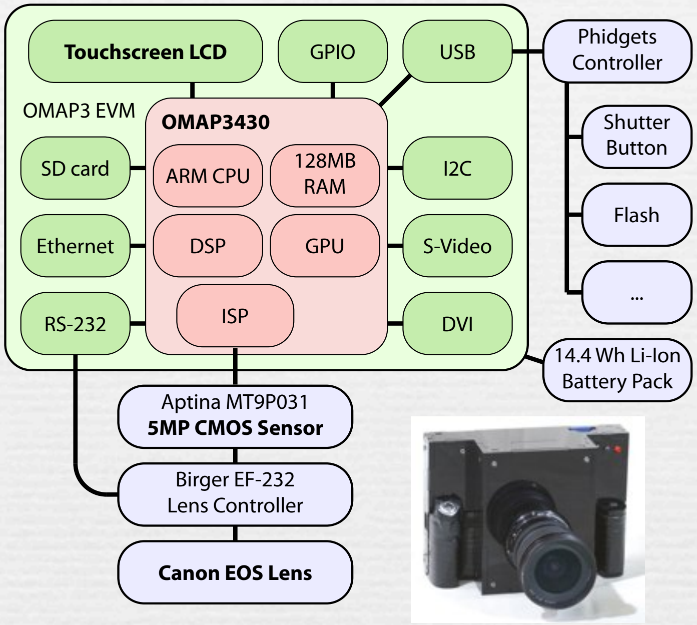
- ✦ demosaicking, white balancing, tone mapping, denoising, sharpening, resizing, gamma correction, compression, etc.
- ✦ may or may not be implemented using fixed-function hardware
- ✦ must be capable of producing
  - a raw sensor image
  - an image suitable for display on a live viewfinder

Application processor



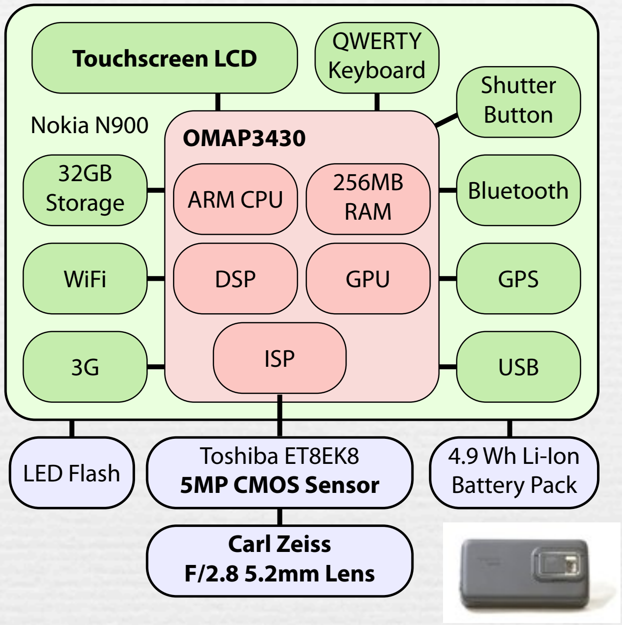
- general-purpose CPU
- auto-focusing, auto-exposure, white balance determination
- synchronization of flash and other devices
- computational photography applications

Two reference implementations



Touchscreen LCD
GPIO
USB
Phidgets Controller
OMAP3 EVM
OMAP3430
SD card
ARM CPU
128MB RAM
I2C
Shutter Button
Ethernet
DSP
GPU
S-Video
Flash
RS-232
ISP
DVI
...
14.4 Wh Li-Ion Battery Pack
Aptina MT9P031
5MP CMOS Sensor
Birger EF-232
Lens Controller
Canon EOS Lens

Frankencamera F2



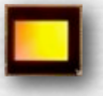
Touchscreen LCD
QWERTY Keyboard
Shutter Button
Nokia N900
OMAP3430
32GB Storage
ARM CPU
256MB RAM
Bluetooth
WiFi
DSP
GPU
GPS
3G
ISP
USB
LED Flash
Toshiba ET8EK8
5MP CMOS Sensor
4.9 Wh Li-Ion Battery Pack
Carl Zeiss
F/2.8 5.2mm Lens

Nokia N900 “F”

- off-the shelf parts
- open-source Linux platform
- interchangeable SLR-quality lenses
- currently building SLR-sized sensor

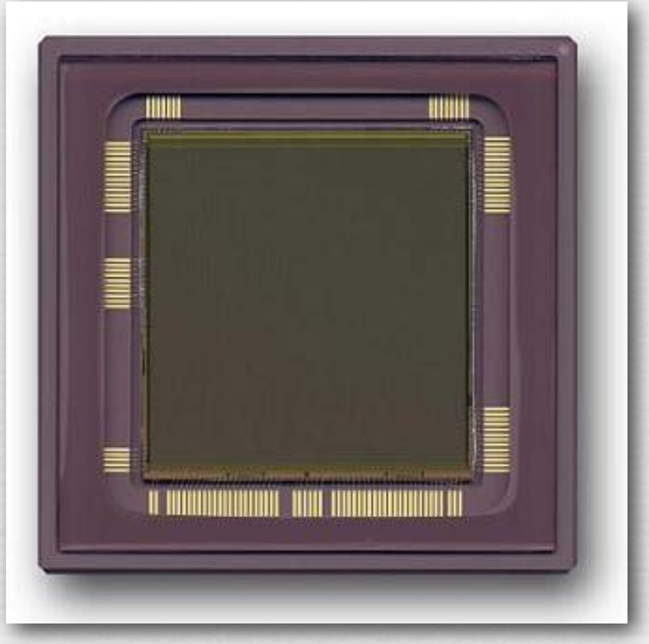
- also runs Linux (partly open source)
- retail hardware + our software stack
- more I/O devices (GPS, radio, etc.)
- runs the same applications

Sensors for the F2



Micron MT9001

- 5 megapixel
- cell phone quality
- $150



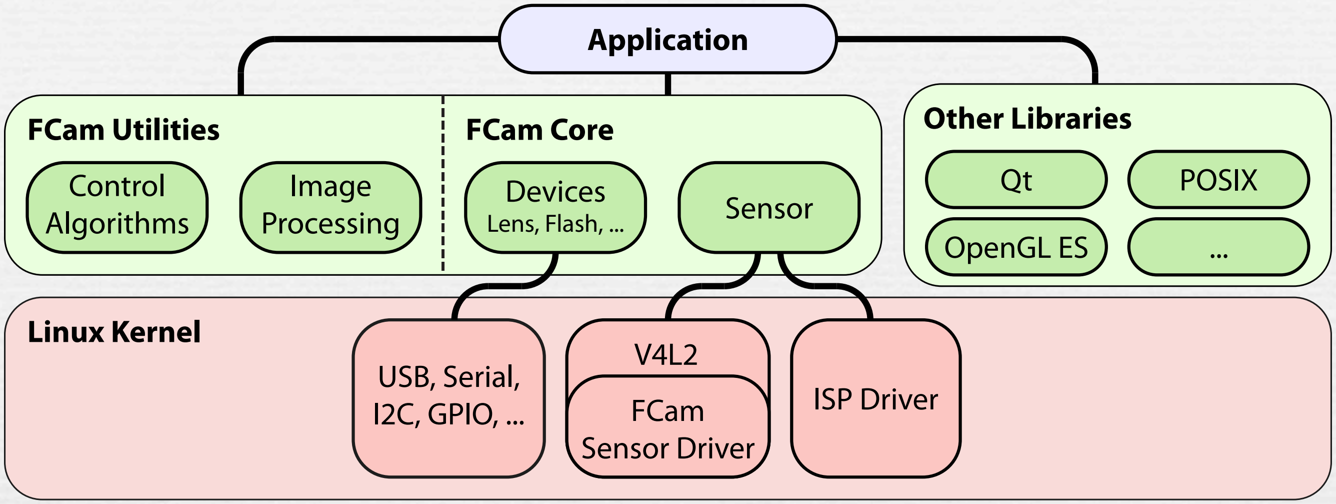
Cypress LUPA 4000

- $1500
- DSLR quality
- arbitrary ROIs and non-destructive readout

Why do we need a new camera API?

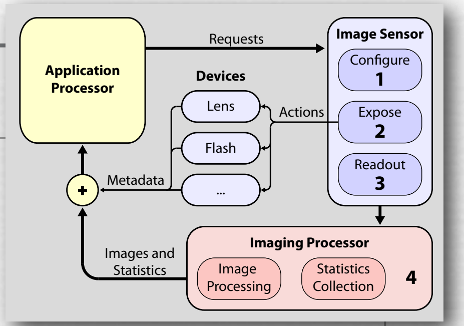
- cell phone makers implement only full-auto camera apps
- so existing APIs are either:
  - *Minimal*: no API control over any settings (iPhone)
  - *Incomplete*: API allows control over many settings, but no implementation actually supports more than the minimum (Symbian CCamera)
  - *Poorly Abstracted* - the camera is presented as either a:
    - *still camera*: only one image capture request can be active at once; frame rate limited to 1/(processing time)
    - *video camera*: full frame rate, but no way to know when a parameter change takes effect, and no way to make sure a parameter set gets applied to the right number of frames

Frankencamera software stack



- what is the right API?
  - choosing the right level of abstraction of the hardware
  - compiling to a heterogeneous computing platform
  - facilitating hardware experiments (LEGO camera)
- standard C++, cross-compiled for device,
  loaded using ssh, debugged using gdb, etc.
- principle of least surprise

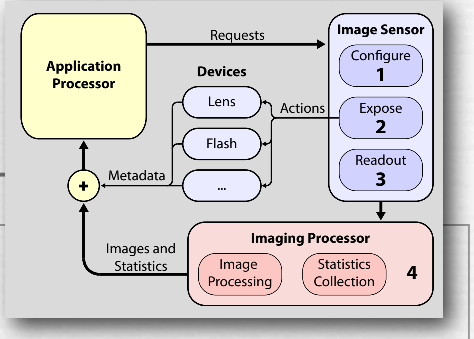
Example #1: capture an HDR stack



- `Shot` describes all parameters used to capture a `Frame`
- `Sensor` transforms shots into frames
- multiple captures in progress at once to maximize frame rate (but programmer must keep the pipeline full)

```
Sensor sensor;
Shot low,med,high;

low.exposure = 1/80.;
med.exposure = 1/20.;
high.exposure = 1/5.;

sensor.capture(low);
sensor.capture(med);
sensor.capture(high);

Frame frames[3];
frames[0] = sensor.getFrame();
frames[1] = sensor.getFrame();
frames[2] = sensor.getFrame();

fused = mergeHDR(frames);
```

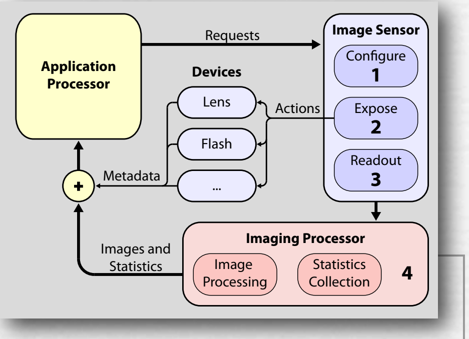
Ex #2: strobing flash-noflash



- shots can be grouped into bursts, which makes their acquisition atomic, and as fast as possible
- a device `Action` can be slaved to a shot, then scheduled at a given time during the exposure
- shots and bursts can be streamed continuously

```
Sensor sensor;
Flash flash;
vector<Shot> burst(2);

burst[0].exposure = 1/200.;
burst[1].exposure = 1/30.;

Flash::FireAction fire(&flash);
fire.time = burst[0].exposure/2;
burst[0].actions.insert(fire);

sensor.stream(burst);

while (1) {
  Frame flashFrame =
    sensor.getFrame();
  Frame noflashFrame =
    sensor.getFrame();
}
```

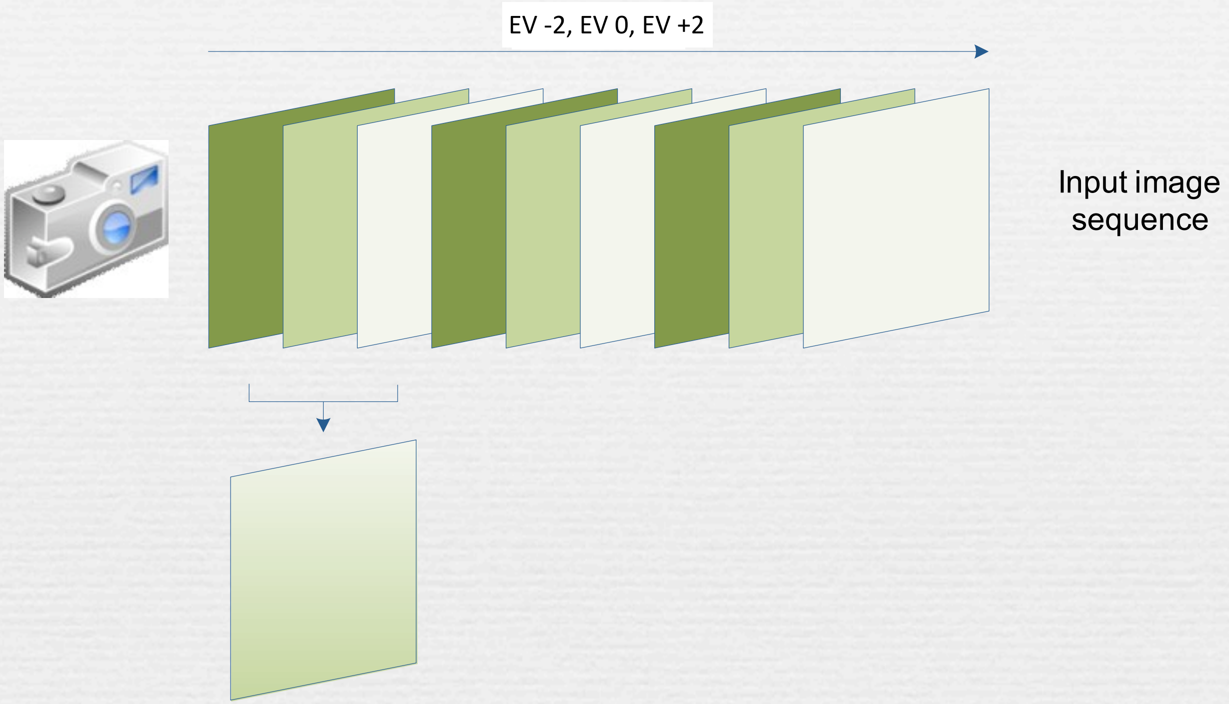
Ex #3: basic auto-exposure



- uses imaging processor to generate histograms
- updates shot exposure based on histogram and exposure data for latest received frame
- frame always tagged with exposure, etc, settings used for its capture

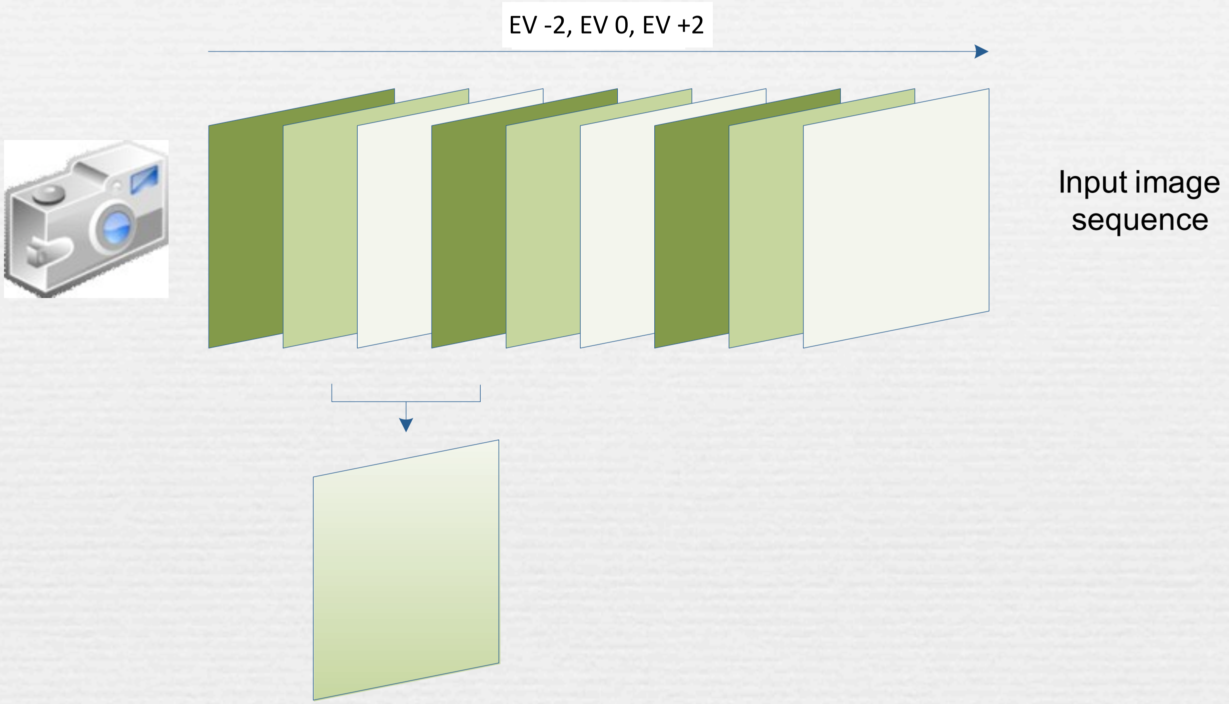
```
Sensor sensor;
Shot viewfinder;

viewfinder.exposure = 1/200.;
viewfinder.histogram.regions = 1;
viewfinder.histogram.region[0] =
  Rect(0,0,640,480)
sensor.stream(viewfinder);

while (1) {
  Frame f = sensor.getFrame();
  if (f.histogram.valid) {
    viewfinder.exposure =
      autoExp(f.exposure,
              f.histogram);
    sensor.stream(viewfinder);
  }
}
```

Application #1: real-time HDR viewfinder



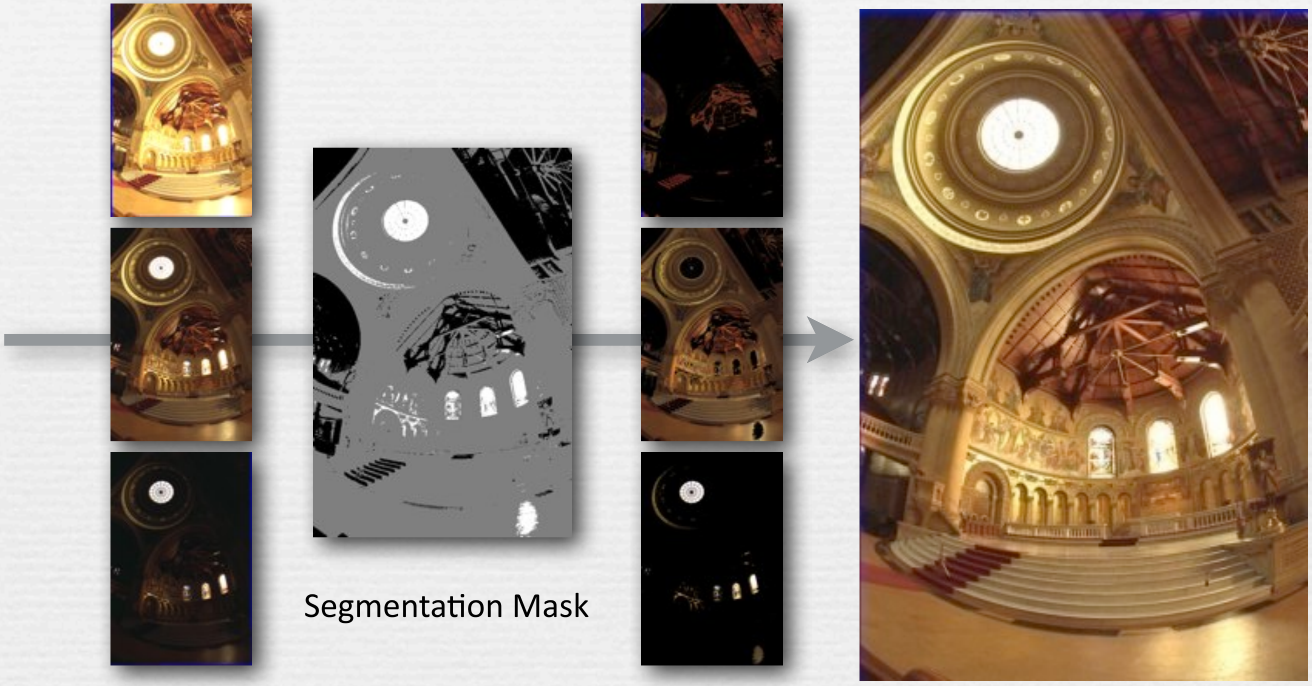
- cycles through three different exposure times at 40fps
- moving 3-frame window merged to HDR and tone mapped
- similar to [Kang 2003] and Stanford CityBlock Project

Application #1: real-time HDR viewfinder



- cycles through three different exposure times at 40fps
- moving 3-frame window merged to HDR and tone mapped
- similar to [Kang 2003] and Stanford CityBlock Project

Application #1: real-time HDR viewfinder



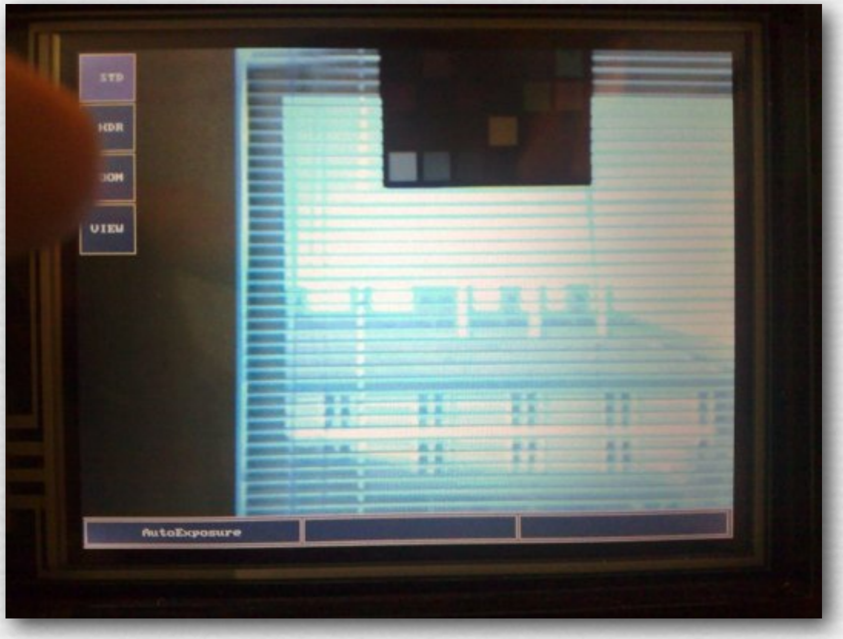
Segmentation Mask

LDR Images

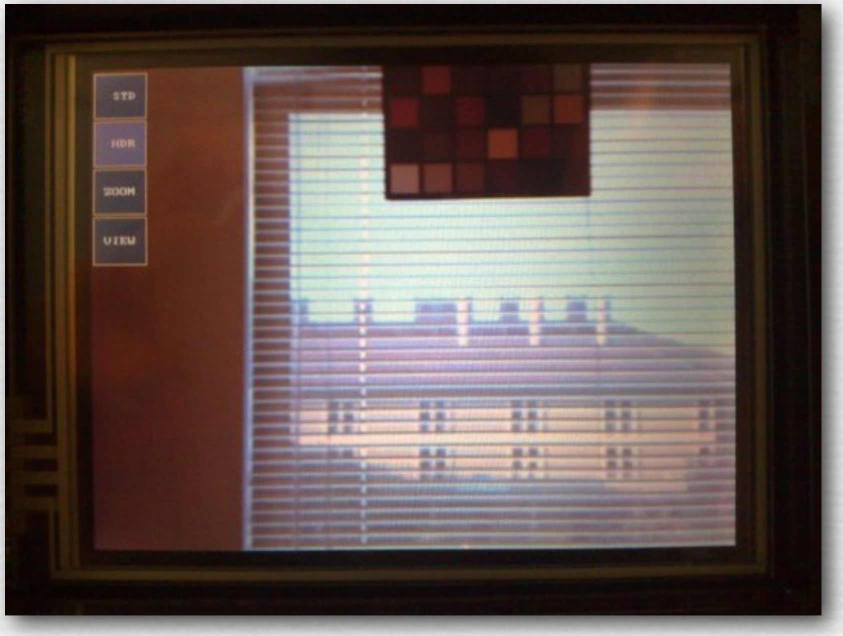
HDR Image

- segmentation by looking for too-bright or too-dark pixels
- simple, global tone mapping curve
- all done with lookup tables

Application #1: real-time HDR viewfinder



single exposure

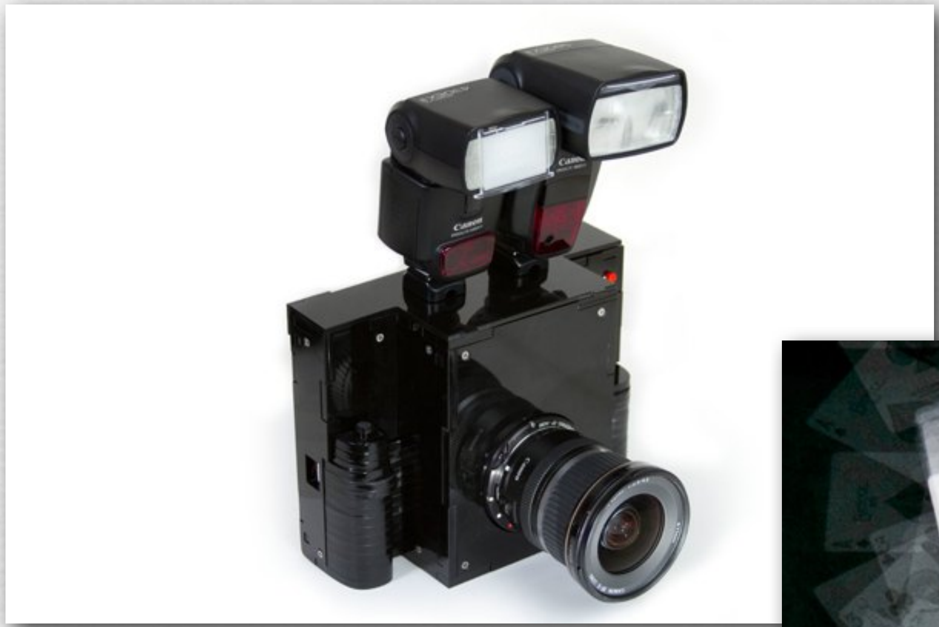


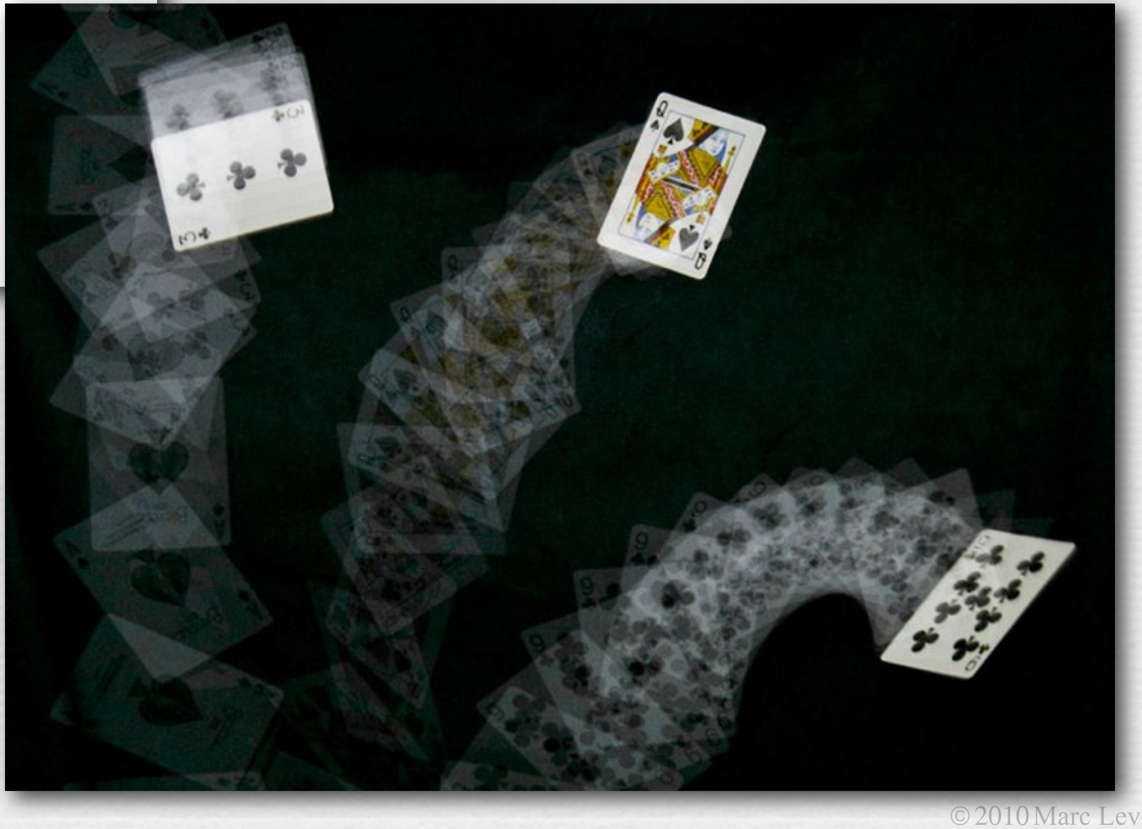
3-exposure HDR

- cycles through three different exposure times at 40fps
- moving 3-frame window merged to HDR and tone mapped
- runs on both Frankencamera F2 and Nokia N900 F

Application #2: dual flash units

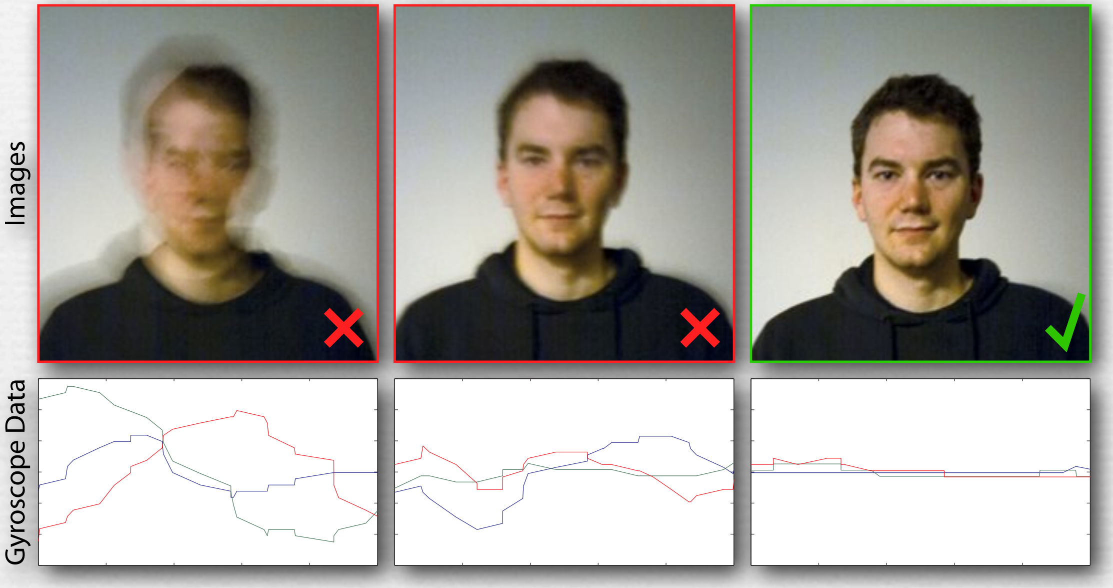
- Canon 430EX (smaller flash) strobed continuously
- Canon 580EX (larger flash) fired once at end of exposure





Application #3: lucky imaging

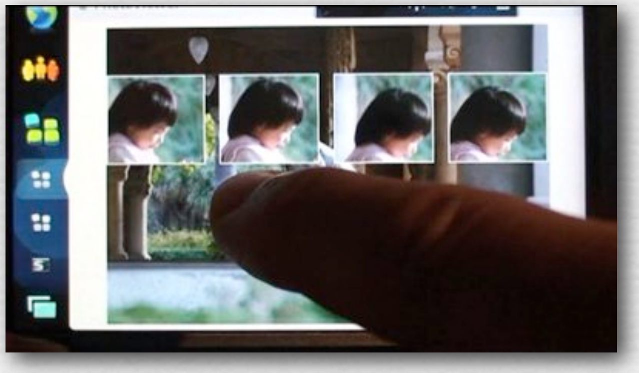
- 3-axis gyroscope on N900
- burst of 1/2-sec exposures
- save image if little motion

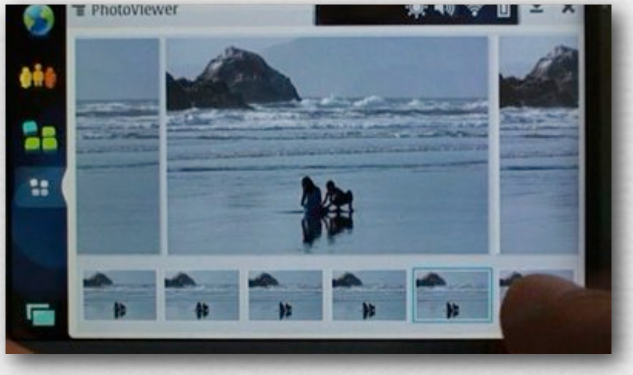


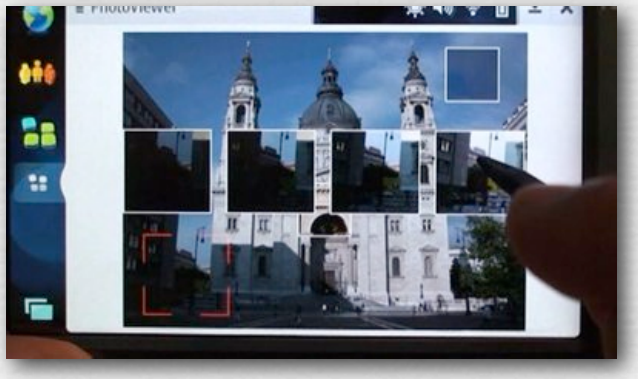
- future: deconvolve using IMU trace as initial guess of kernel
- also: deconvolve from multiple lucky images

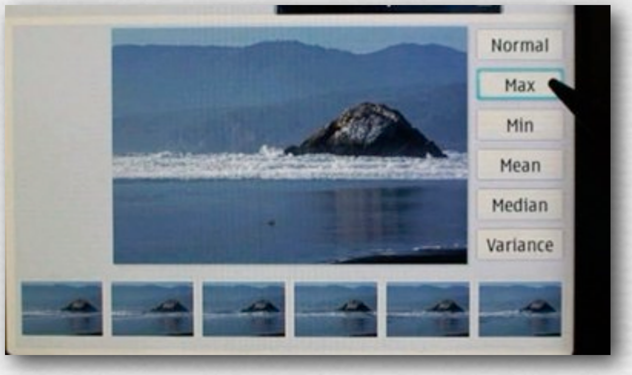
Rethinking the user interface

- controlling the camera while shooting
- Did I capture enough information?
- editing and sharing









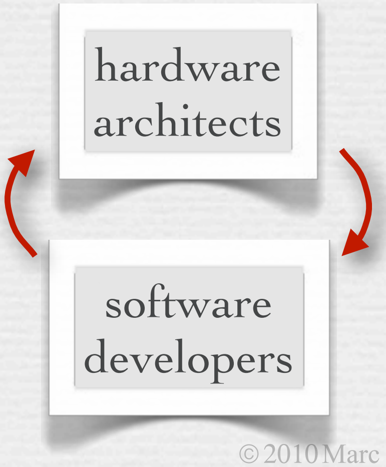
Long-term roadmap

- distribution to researchers and students
  - courseware + Frankencameras/N900s
  - bootstrap open-source community

- distribution to hobbyists, 3rd party developers
  - probably only N900s or equiv.
  - plugins and apps

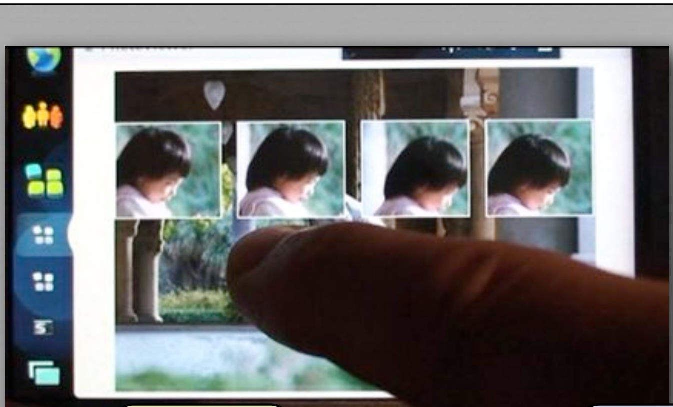
openSourceCamera.org

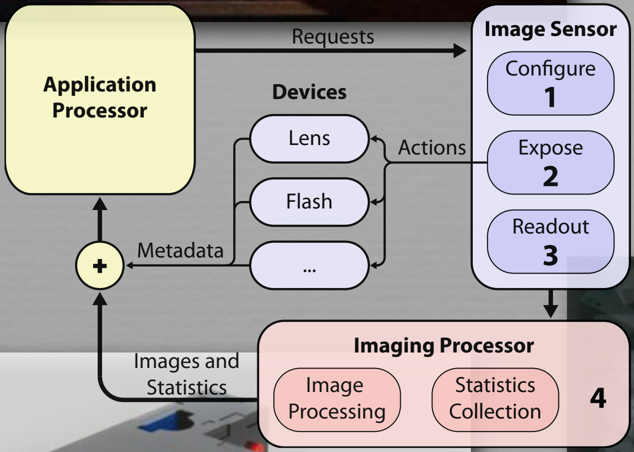
- wish list for makers of camera hardware
  - per-frame resolution switching at video rate
  - fast path into GPU texture memory
  - hardware feature detector

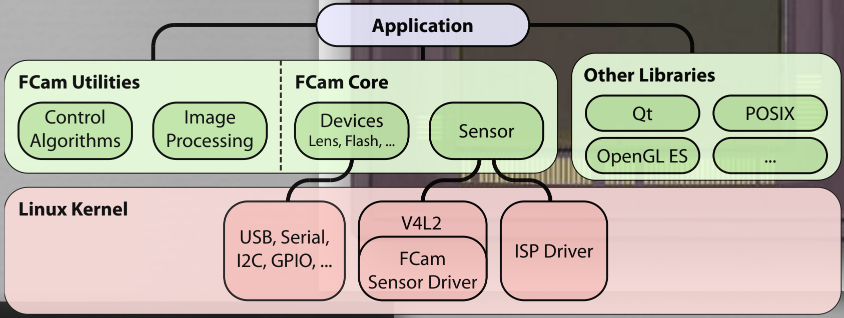


hardware architects

software developers







```
Sensor sensor;
Shot low, med, high;

low.exposure = 1/80.;
med.exposure = 1/20.;
high.exposure = 1/5.;

sensor.capture(low);
sensor.capture(med);
sensor.capture(high);

Frame frames[3];
frames[0] = sensor.getFrame();
frames[1] = sensor.getFrame();
frames[2] = sensor.getFrame();

fused = mergeHDR(frames);
```

http://graphics.stanford.edu/projects/camera-2.0/