Microsoft Research

tackling the battery problem

a scenario based approach

Victor Bahl

Oct. 5, 2014

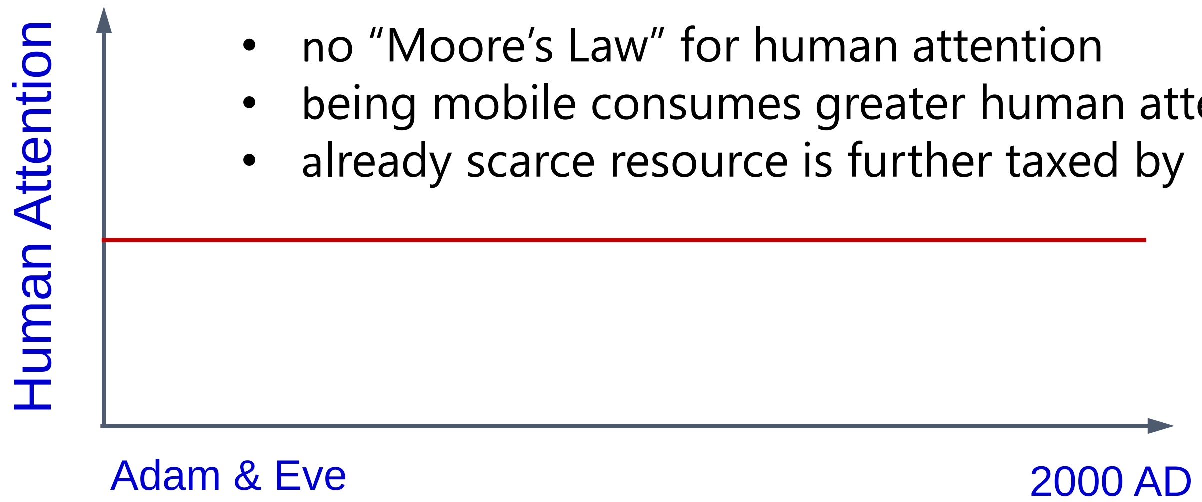
HotPower 2014

my amazing collaborators

- chen, yu-han (MIT)
- chandra, ranveer
- han, seungyeop (UW)
- liKamWa, robert (Rice)
- priyantha, bodhi
- philipose, Matthai
- wolman, alec
- zhong, lin (Rice)

resource poverty hurts

- no “Moore’s Law” for human attention
- being mobile consumes greater human attention
- already scarce resource is further taxed by resource poverty



technology should reduce the demand on human attention

clever exploitation of {*context awareness, computer vision, machine learning, augmented reality*} needed to deliver vastly superior mobile user experience

Microsoft

continuous mobile vision

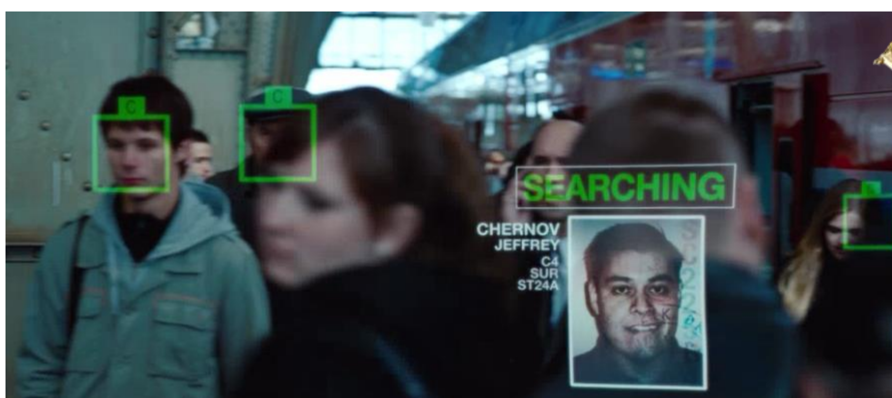
reality vs. movies

Steve Mann (early 90s)

COBOT, CMU (2013)

iRobot (2004)

Mission Impossible 4 (2011)



C-3PO (1977)

Victor Bahl, MSR

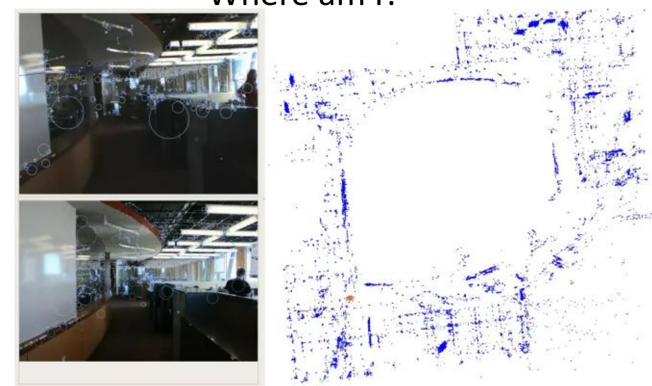
Microsoft

perennial challenges

MSR's SenseCam for memory assistance

Augmented Reality

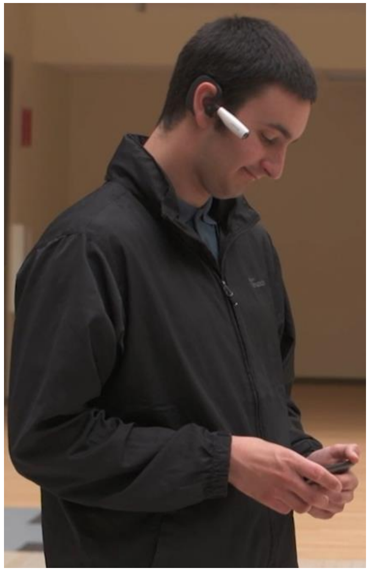
Where am I?

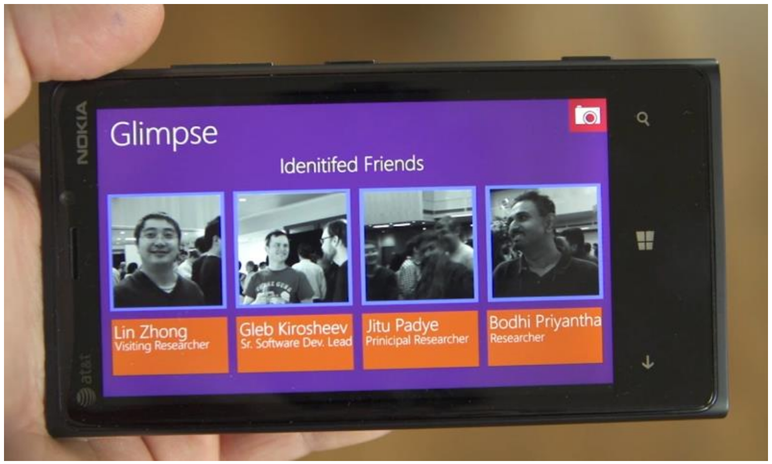


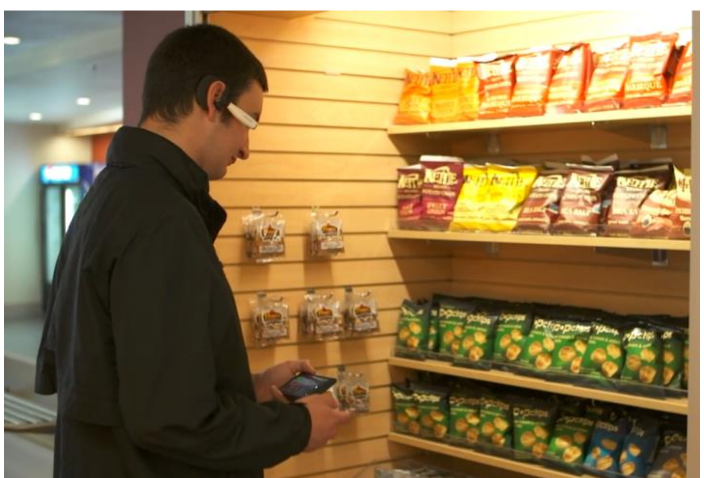
- computation } cloudlets
- connectivity & bandwidth } white space networks, small cell networks, mm-wave networks
- battery

Resource constraints prevent today's mobile apps from reaching their full potential

MSR’s Glimpse project







challenges in vision-based applications

resources

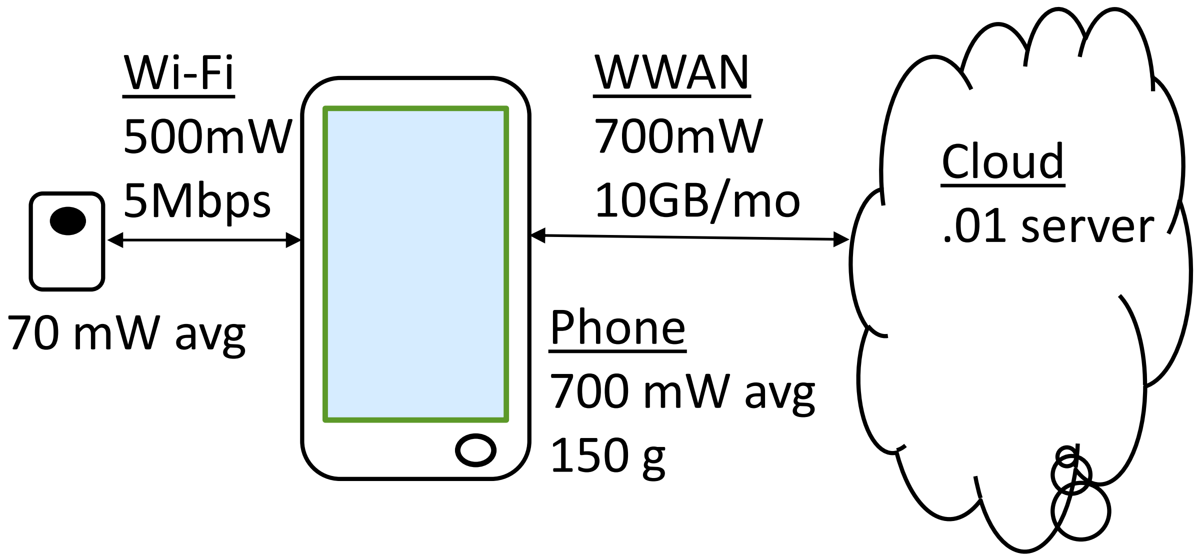
- cpu, bandwidth, power are limited

vision algorithms

privacy and security

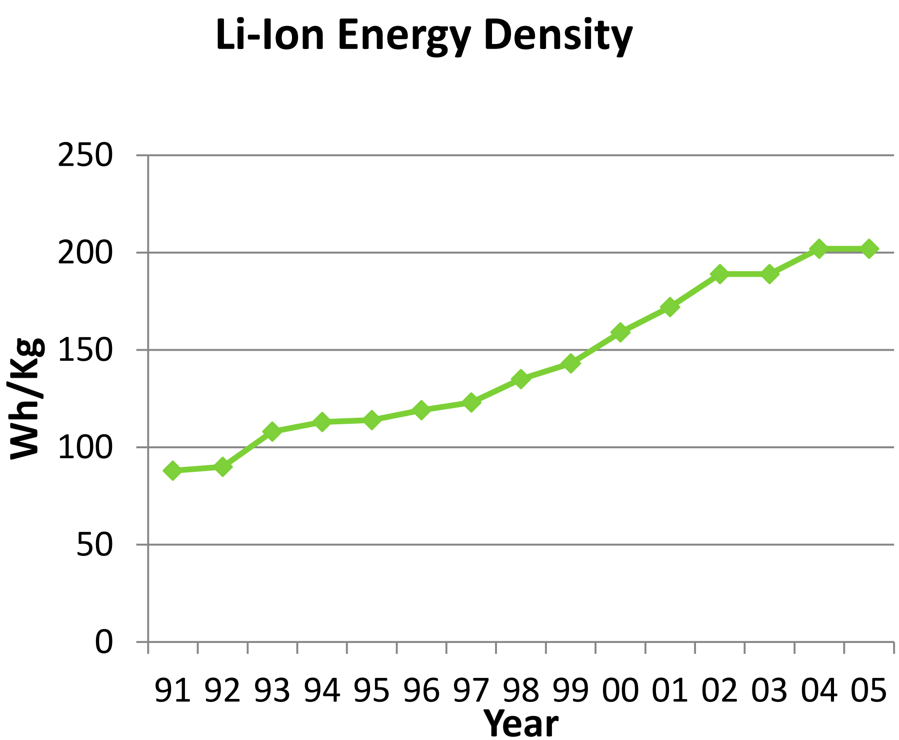
user interaction with applications

break it down into a systems issue…



need *cost-sensitive* detection !

battery improvement trends look bad



- lagged behind
  - Higher voltage batteries (4.35 V vs. 4.2V) – 8% improvement
  - Silicon anode adoption (vs. graphite) – 30% improvement
- trade-offs
  - Fast charging = lower capacity
  - Slow charging = higher capacity

- CPU performance improvement during same period: 246x
- A silver bullet seems unlikely

so where is the energy going?

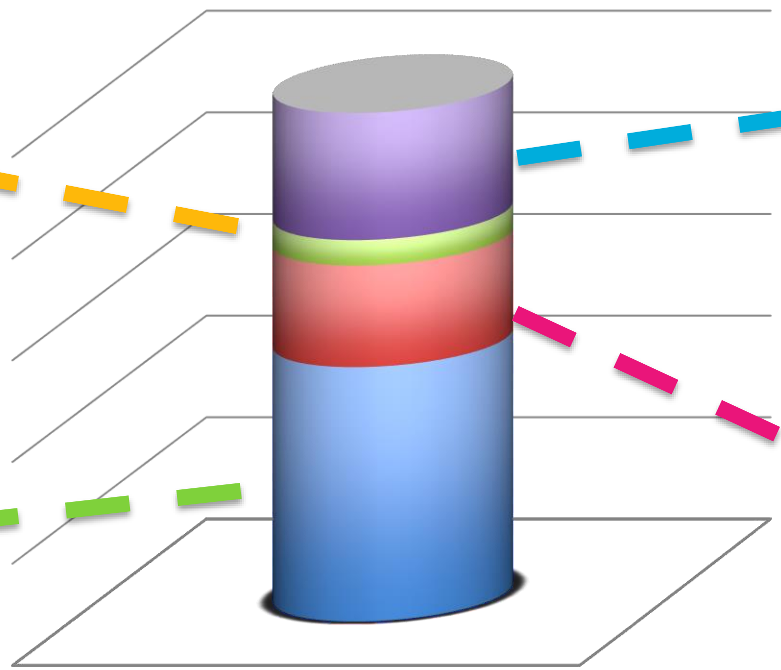
assuming a typical SmartPhone battery of 1500 mAh (~5.5 W)

Sensors + Memory + Disk

~ 15 mW

Display

~500 mW



Single Core Processor

CPU + GPU

~150 mW

Network Stack

(5 min. of usage / hour)

~100 mW

battery lifetime ~7.25 hours

Microsoft

power consumption of a typical image sensor

Reduce frame rate ←

Reduce resolution ↓

5 MP, 5 fps 345 mW		
1 MP, 5 fps 250 mW	1 MP, 15 fps 295 mW	
0.3 MP, 5 fps 232 mW	0.3 MP, 15 fps 245 mW	0.3 MP, 30 fps 268 mW

low resolution, low frame rate image sensing for vision related tasks can reduce battery life by > 25%

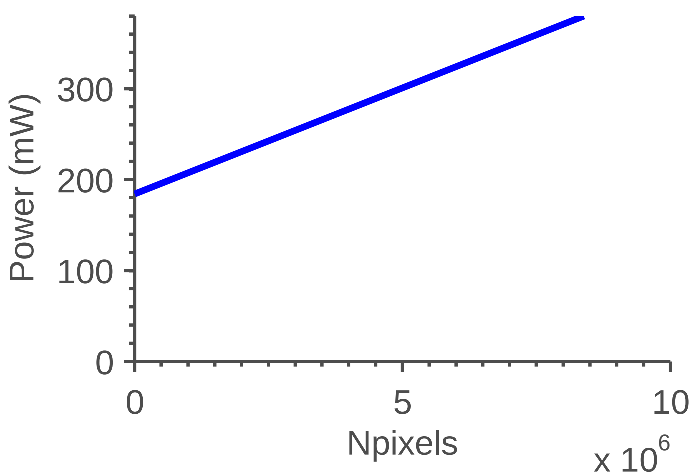
Microsoft

state of art

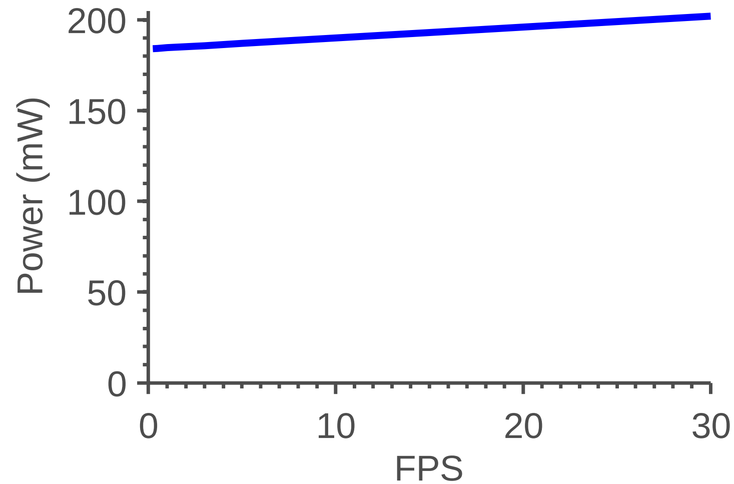
energy / pixel is **inversely proportional** to the frame rate & image resolution

Profiled 5 image sensors from 2 manufacturers

power vs. resolution
Video at 30 fps



power vs. frame rate
Video at 0.1 MP

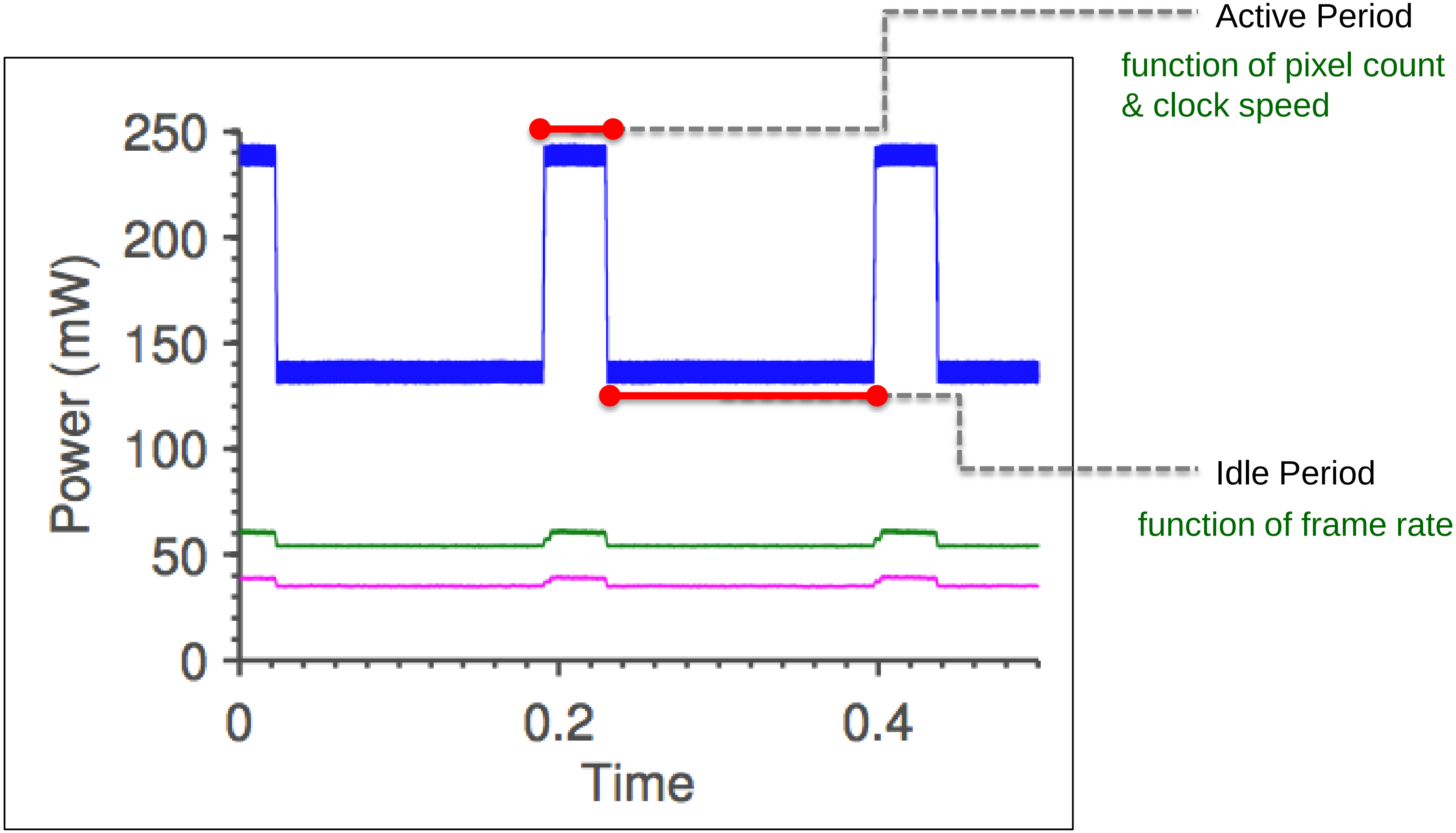


regardless of image resolution & frame rate,
image sensors consume about the same power

Victor Bahl, MSR

Microsoft

digging deeper (1 MP, 5 fps)



$$E_{frame} = P_{active}T_{active} + P_{idle}T_{idle}$$

Microsoft

reduce power by reducing pixel readout time

one pixel is read out per clock period

$$E_{frame} = P_{active} T_{active} + P_{idle} T_{idle}$$

reduce this

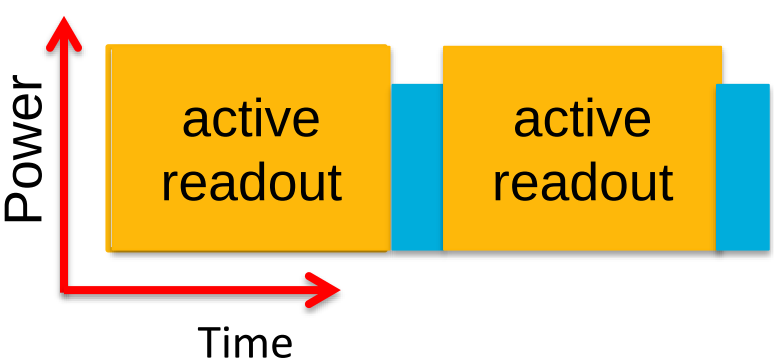
$$T_{active} = \frac{N}{f}$$

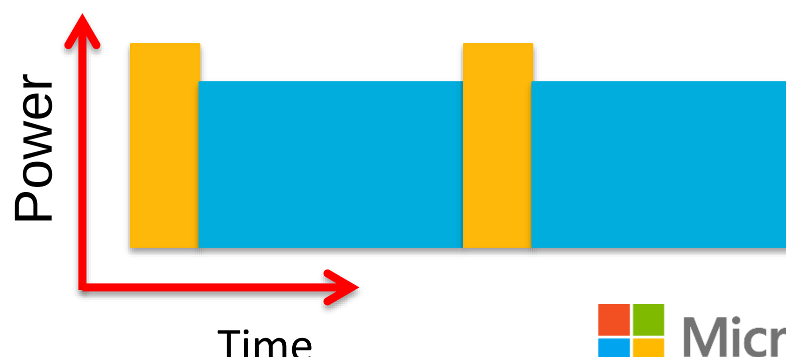
Number of Pixels
divided by
Clock Frequency

reducing pixel count (N)

region-of-interest (windowing)

scaled resolution (pixel skipping)



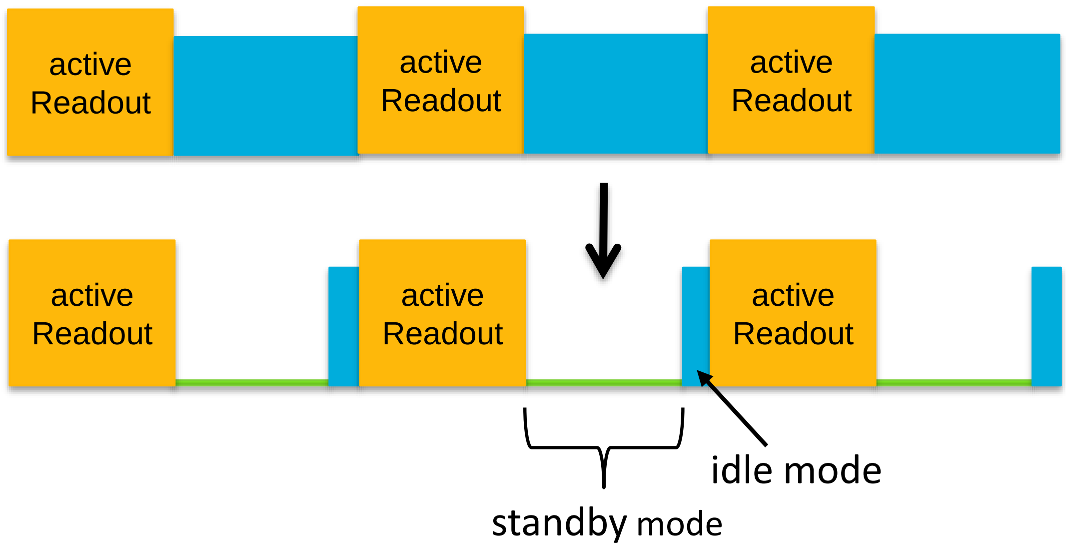


Microsoft

reduce power by aggressive use of standby

turn off sensor during idle period

idle mode necessary to allow exposure before readout



best when frame rate and resolution are sufficiently low

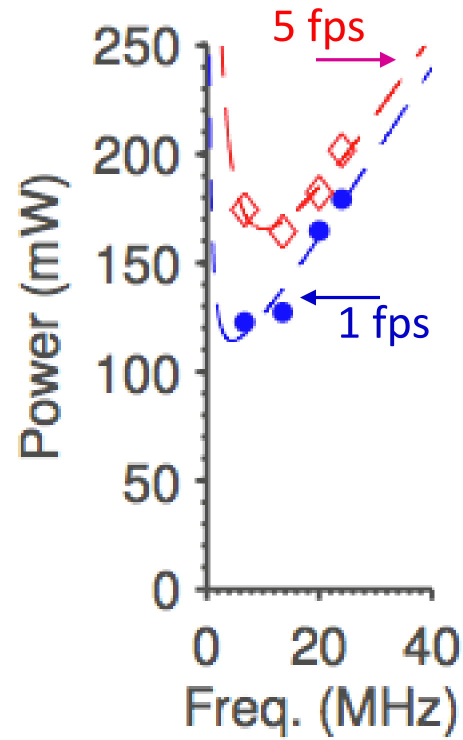
Microsoft

reduce power by adjusting clock frequency

Adjust clock frequency to minimize power

$$T_{active} = \frac{N}{f}$$

Adjust this



Tradeoff

frequency ↑ Power ↑

frequency ↑ T_{active} ↓

At low frame rates, run the clock as slow as possible

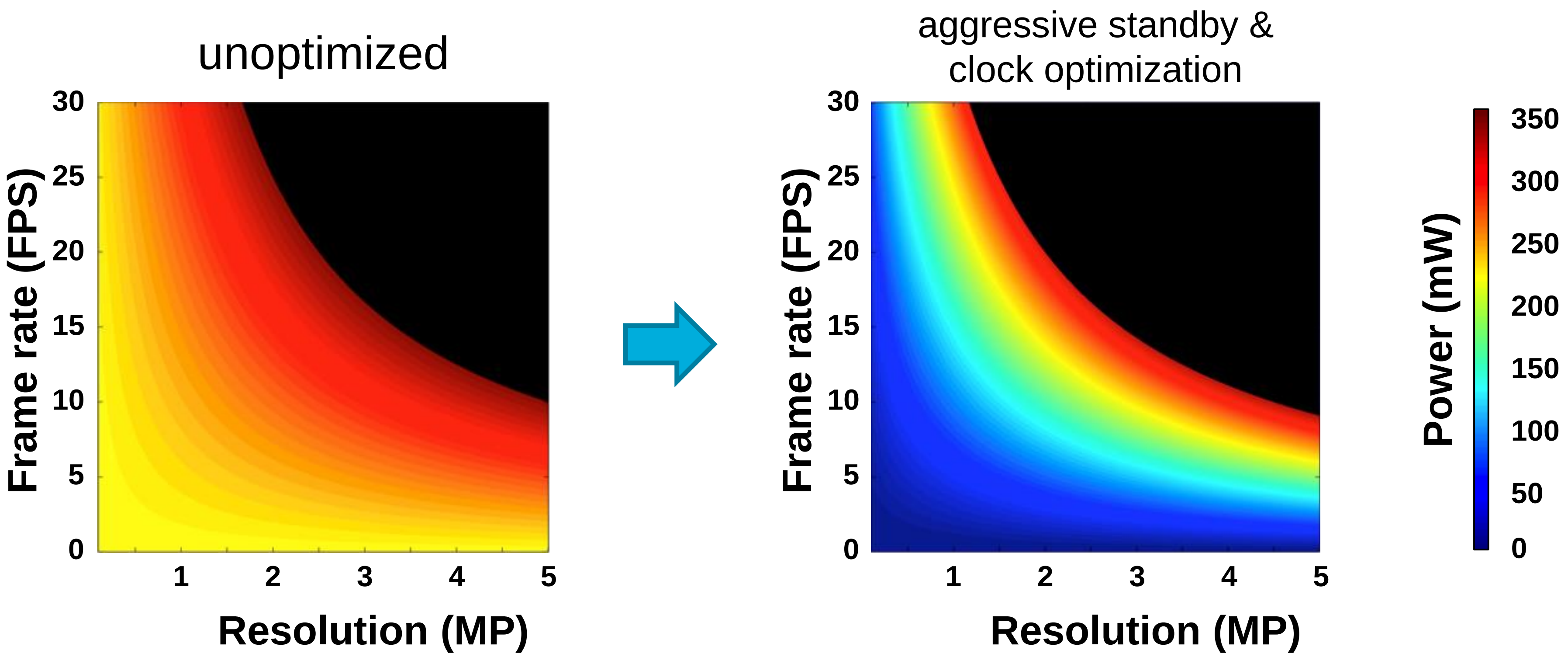
Microsoft

summarizing power reduction techniques for image sensors

- reduce T_{active} & increase T_{idle}
  - ✓ decrease frame rate
  - ✓ reduce total pixel readout time (by reducing N)
  - ✓ adapt clock frequency
- instead of idle-ing put sensor in *standby state*
- reduce P_{active} (not covered in this talk, see paper)

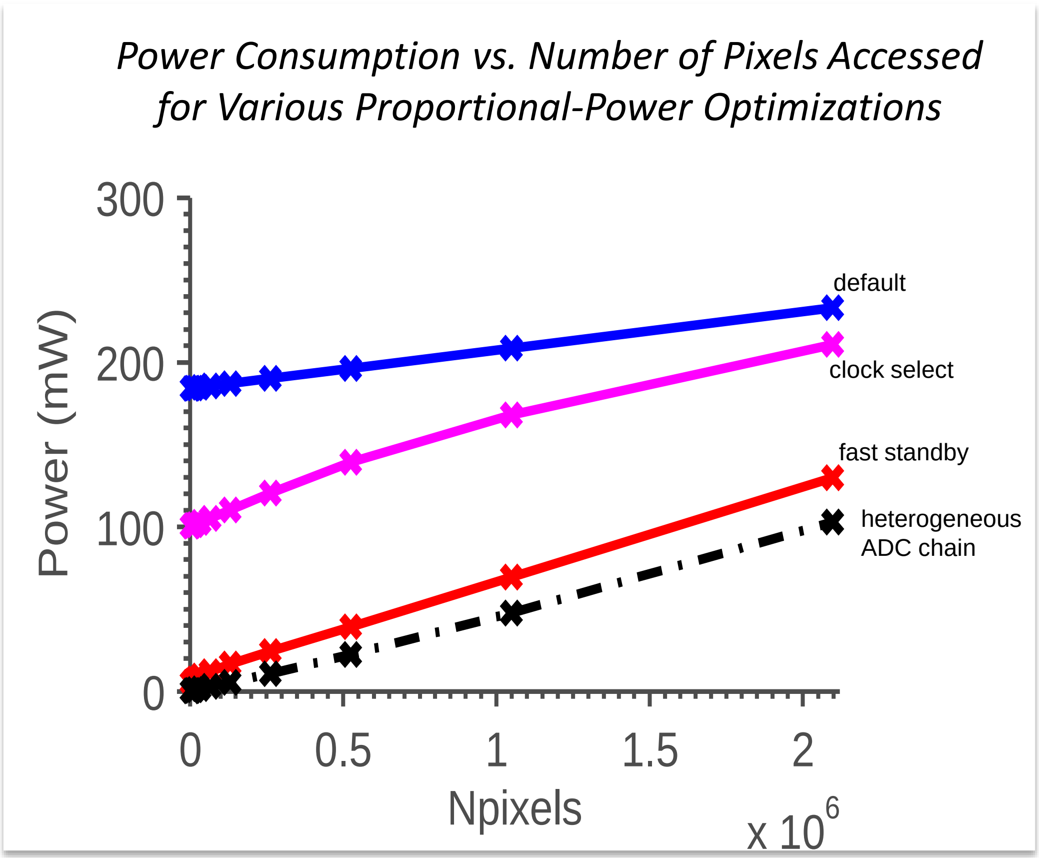
Microsoft

applying these techniques



stated another way

actual numbers



impact on vision algorithms

image registration

person detection

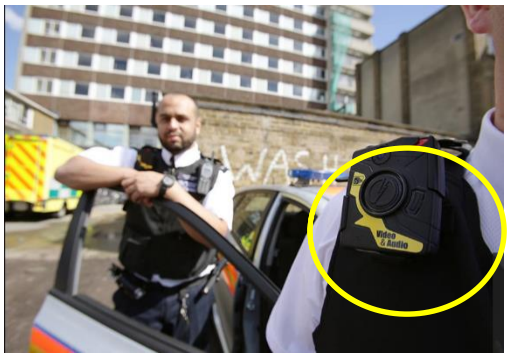
480 x 270

	Image Registration Success	Person Detection Success	Actual Power Reduction with software assist	Estimated Power Reduction with hardware assist
Full Resolution (129600 pixels)	99.9%	94.4%	51%	84%
Frame Rate- 3 FPS	**95.7%**	83.3%	**95%**	98%
30% Window (63504 pixels)	96.5%	77.8%	63%	91%
Subsampled by 2 (32400 pixels)	91.8%	72.2%	71%	94%

that's great, but what else can we do?

Step 1: collect some image data

first, collect some real-world data…

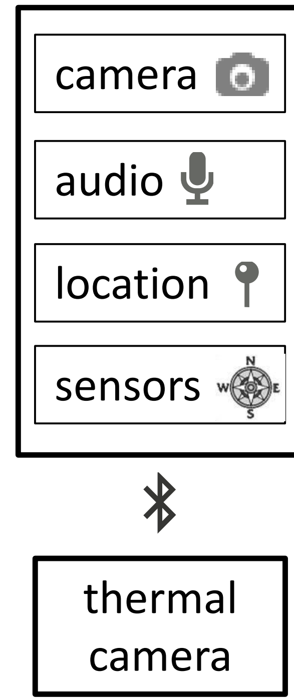


[camera](#) integrated into the officer's uniform (500 London police officers are carrying this around).

Seungyeop Han's version

data gathering application

- ~5 video frame per second
- sync with timestamps
- collect all possible sensors



- xustom-built
- 16x4 temp array
- 40x15° FOV

analyze the frames in the video data …

data was collected while walking around,

total 116 minutes over 7 days

~1M sensor readings

>30k RGB frames

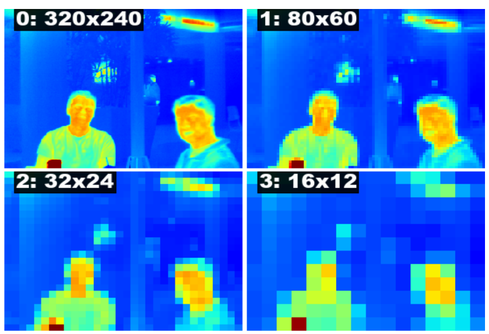
~100k thermal camera frames

less than 5% frames contained faces, another way to look at this, 99% of the windows (smaller than a frame) did not contain a face

Microsoft

analyze the thermal camera data

200mW



~10mW using Melexis 90620 imager

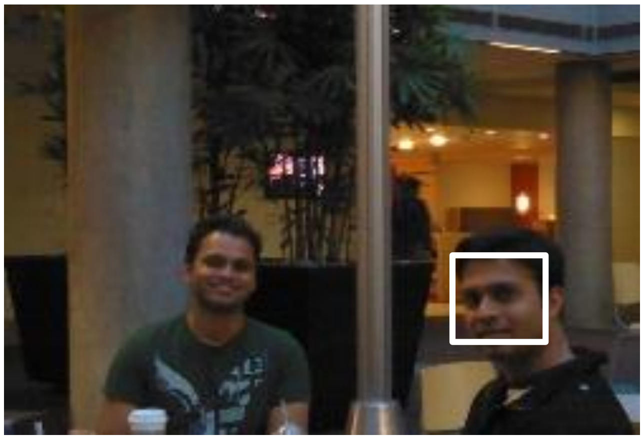
thermal camera (gating imager):

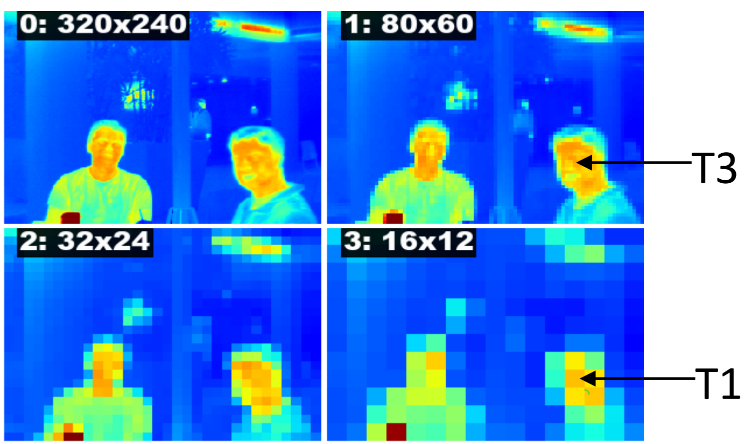
- low-resolution, low-power
- can *detect*, but not recognize, entities, e.g., body parts, planar surfaces, text

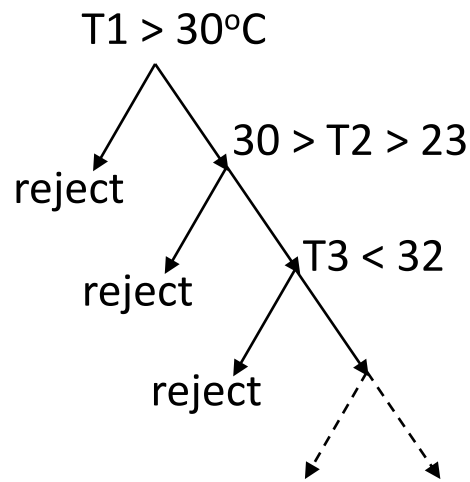
gating avoids need to read most (hi-res) pixels

Microsoft

detect objects

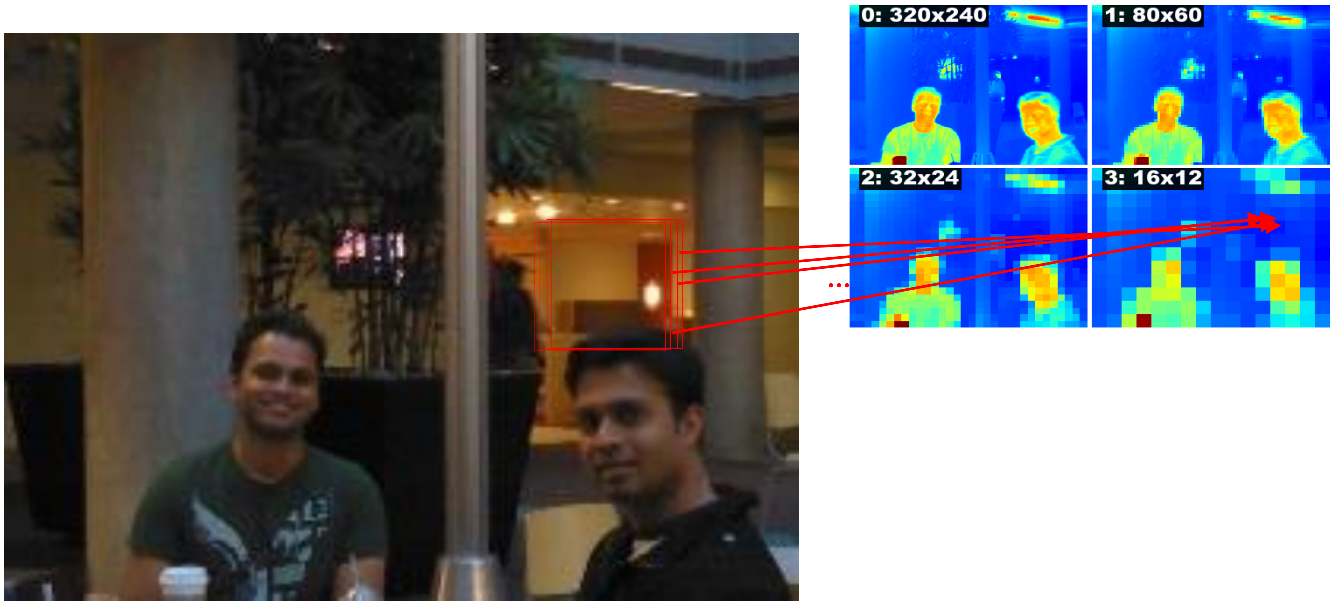




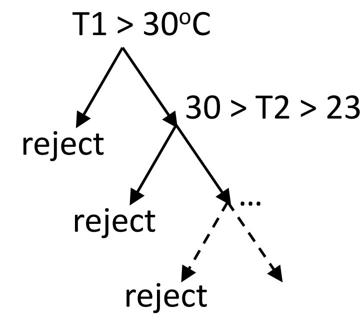


Microsoft

reject windows with no objects

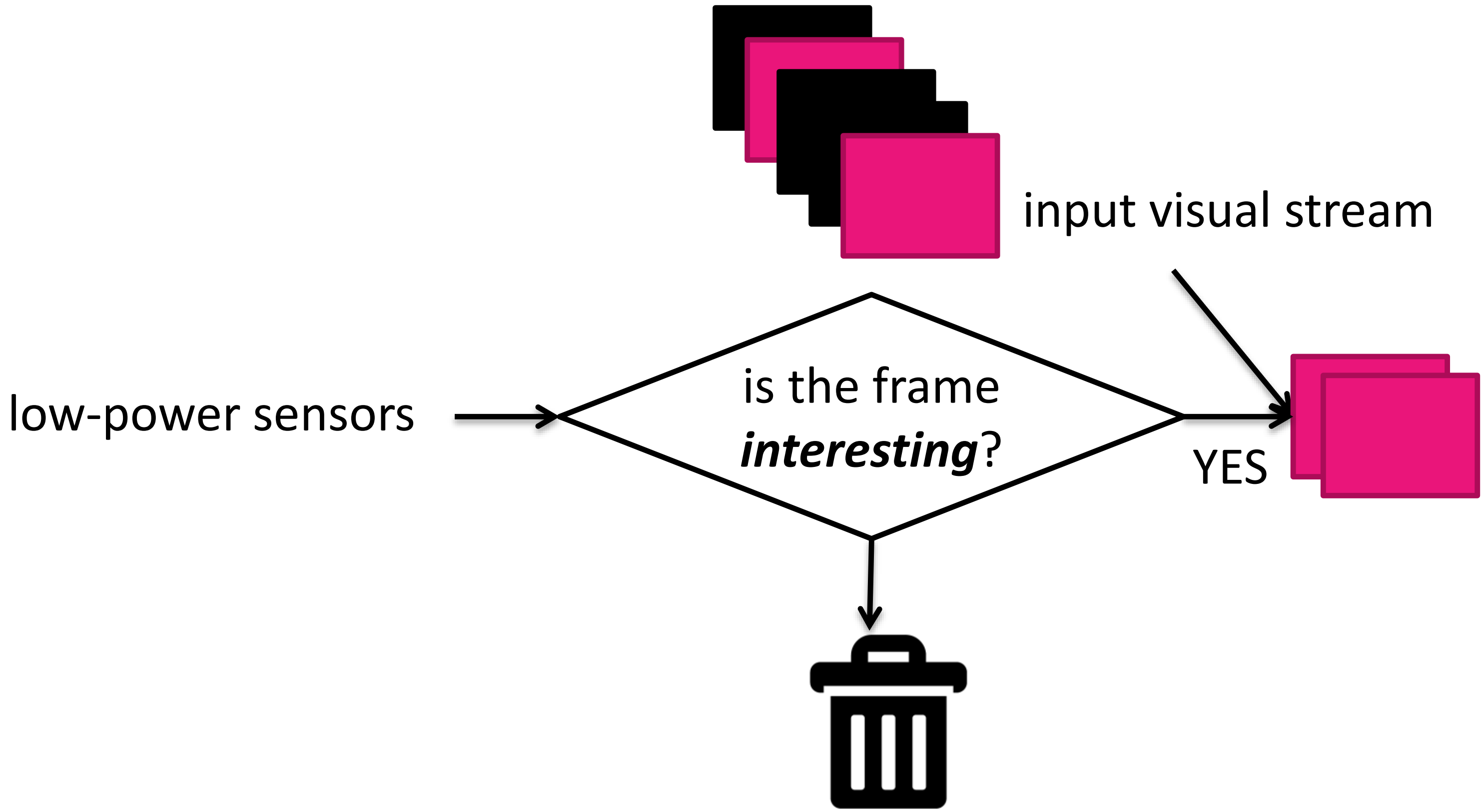


Nearby windows share tree prefixes:

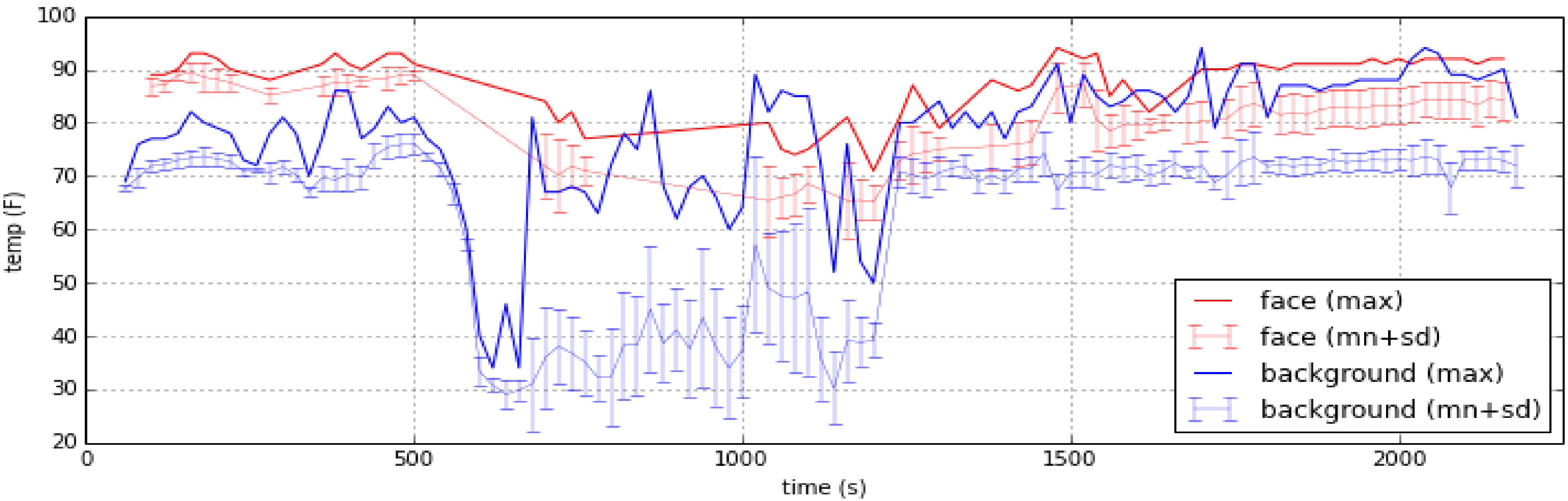


Ensemble structure allows many windows may be rejected by few gating pixels

so can we use lower power sensors to filter out uninteresting frames?



challenge: gating thresholds vary with time

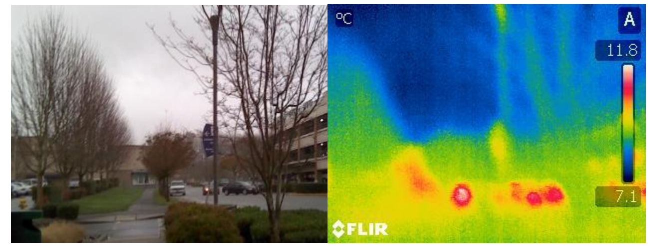


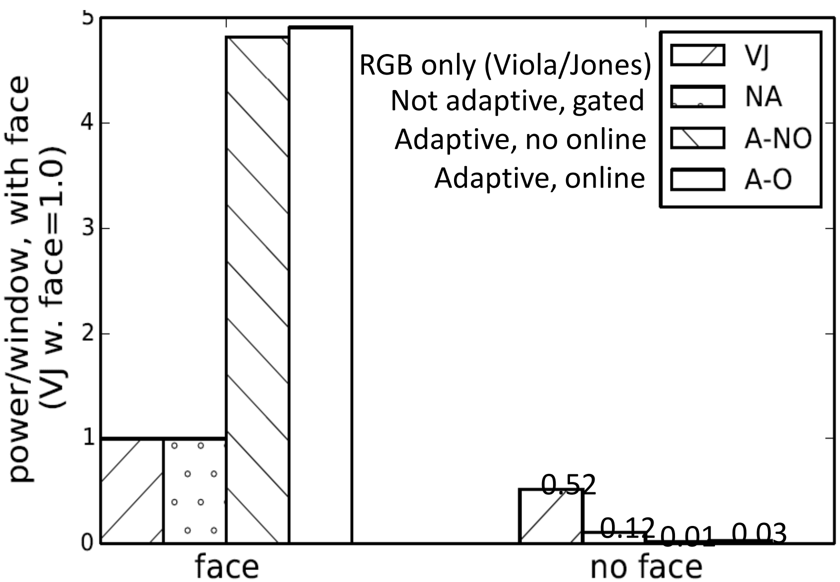
motivates *online re-estimation* of adaptive detectors

AAAI 2014

Microsoft

results





est. power consumed under various schemes

assume 40nJ/read, 5nJ/instruction executed

gating uses ~50x less power to detect windows with no faces

current implementation uses ~5x more power when faces are present (extra checks on gating pixels)

BUT, in real data << 0.01% of windows have faces => Overall efficiency gain of ~ 50x

…. more results in paper

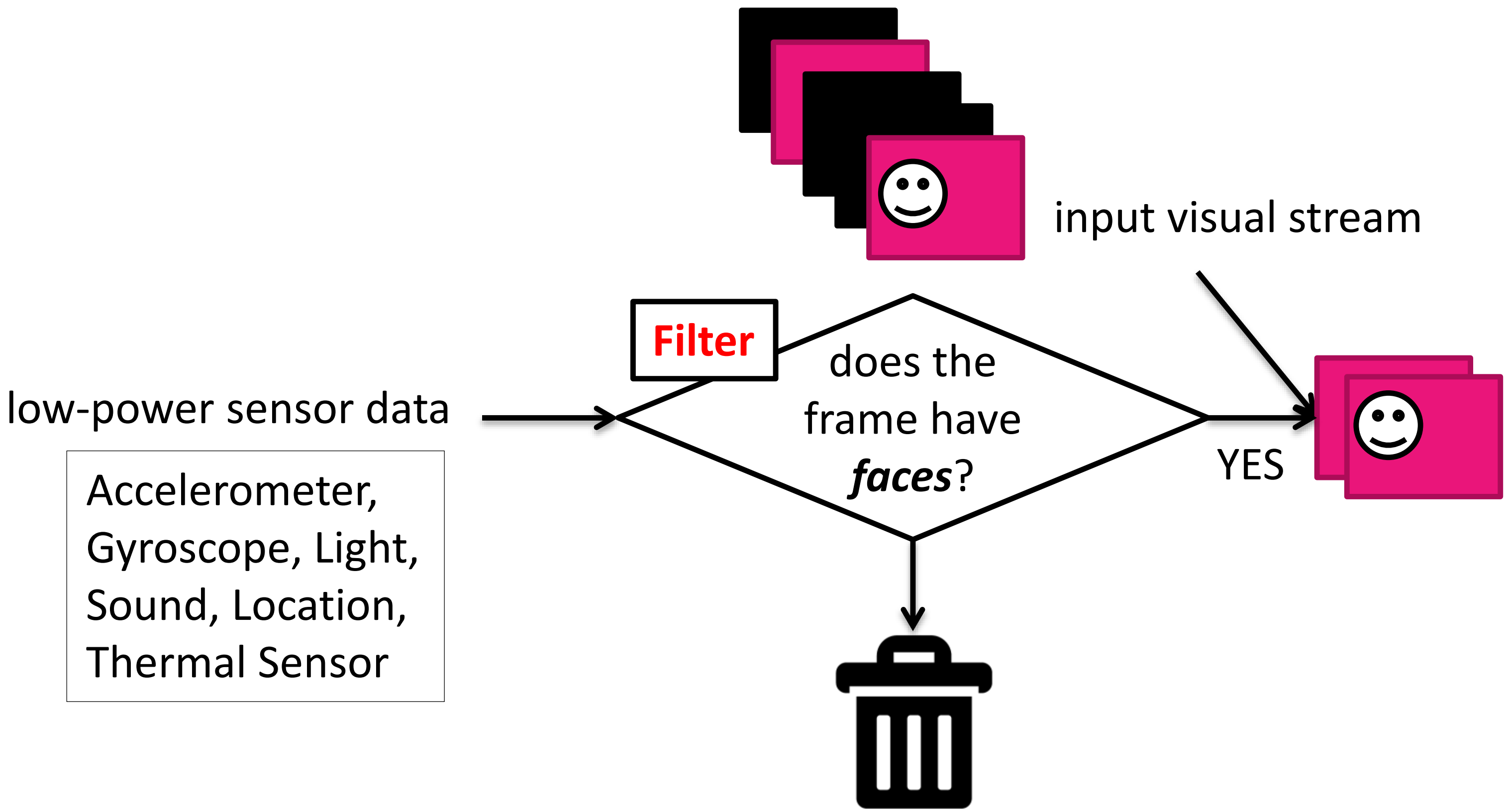
putting it together

- most image frames do not contain objects of interest

- most pixel windows inside a frame do not contain objects of interest

- *gating* imagers, which measure quantitates like temperature or depth can establish the presence or absence of objects with little processing

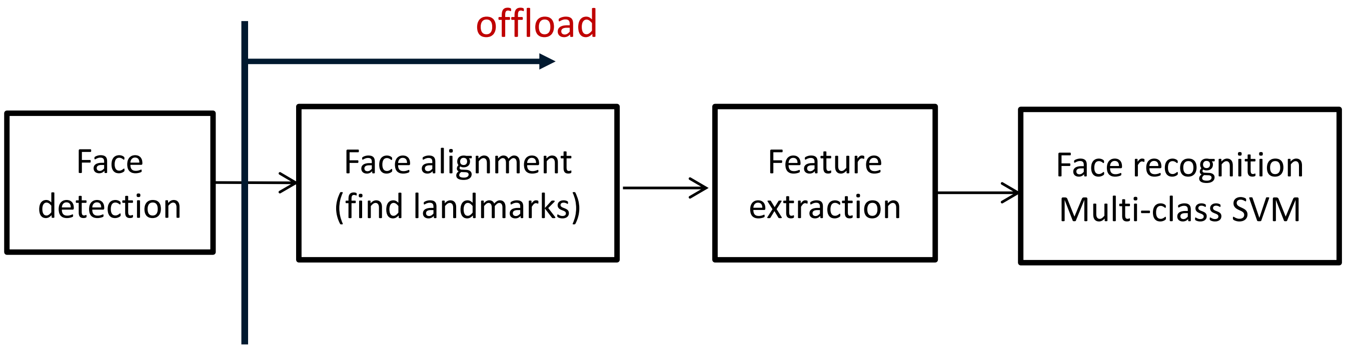
so let's use them!



can we determine if a frame is unlikely to have a face before running face detector?

AAAI 2014

not done yet.....
still need to do object recognition



object recognition

offload to cloud

we (and others) have shown
remote execution reduces energy
consumption and improves performance

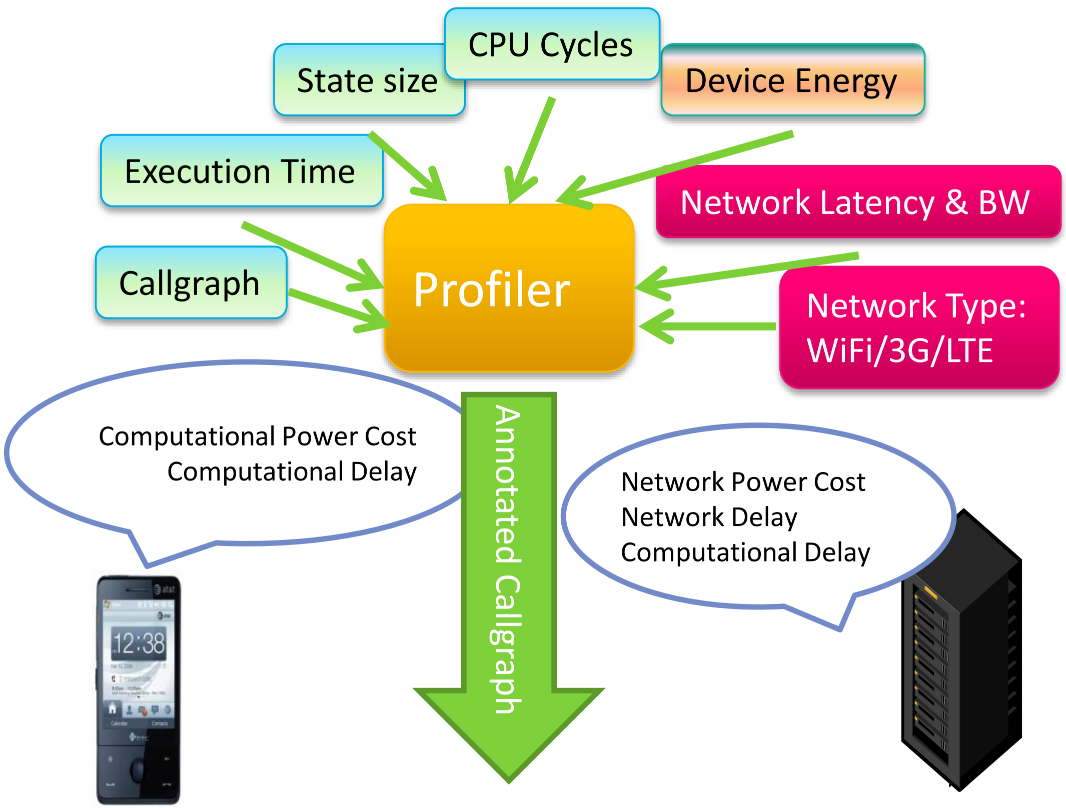
challenges:
- what to offload?
- how to dynamically decide when to offload?

Microsoft

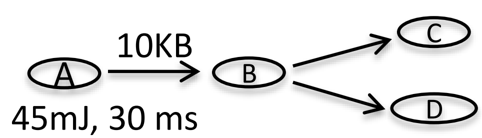
when to offload?

profiler:

handles dynamics of devices, program behavior, and environment (Network, Server Load)



- Vertex: method annotated with computation energy and delay for execution
- Edge: method invocation annotated with total state transferred



decision engine:

partition a running app

use an Integer Linear Program (ILP) to optimize for performance, energy, or other metrics…

Example – Maximize:

$\sum_{v \epsilon V} (I_v \times E_v) \quad - \quad \sum_{(u,v)\, \epsilon\, E} (|I_u - I_v| \times C_{u,v})$

energy saved — cost of offload

Such that:

$\sum_{v \epsilon V} (I_v \times T_v) \quad + \quad \sum_{(u,v)\, \epsilon\, E} (|I_u - I_v| \times B_{u,v}) <= Lat.$

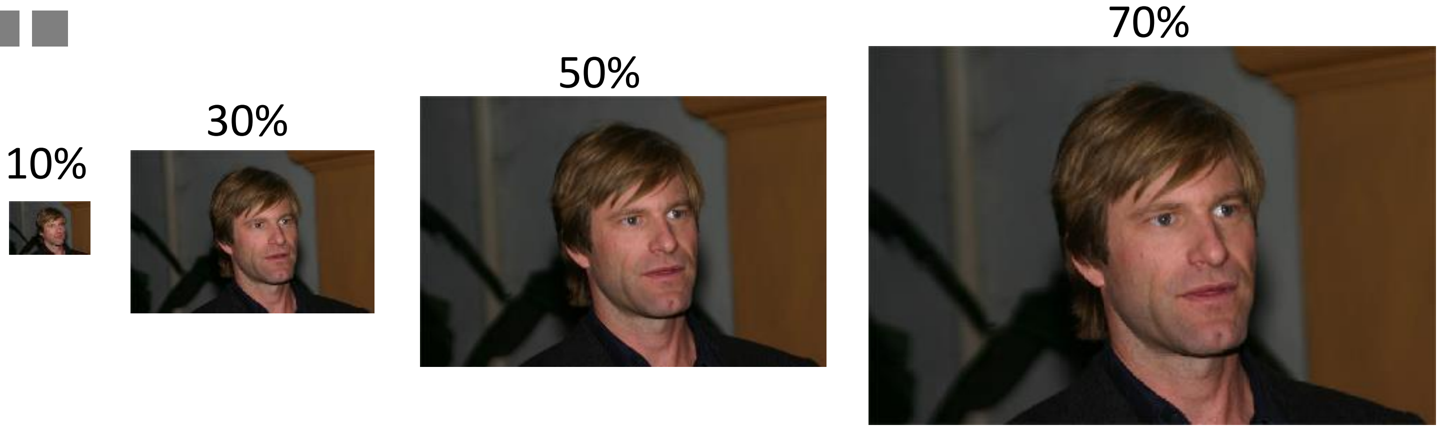
execution time — time to offload

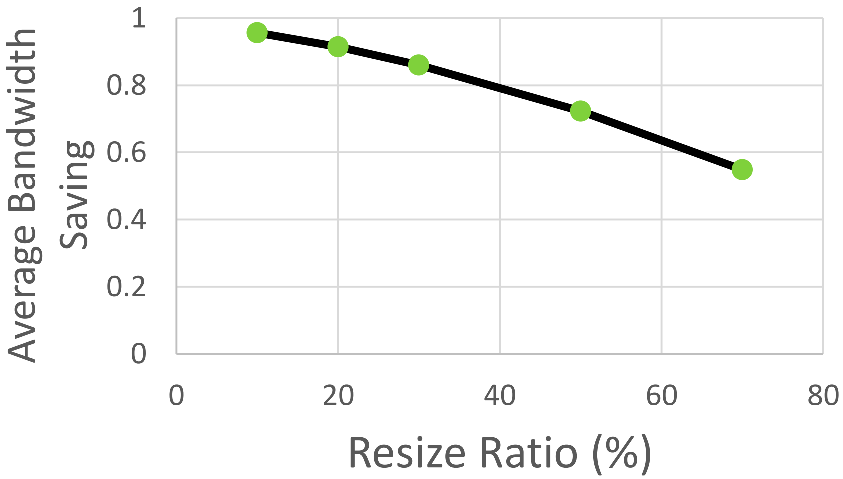
and

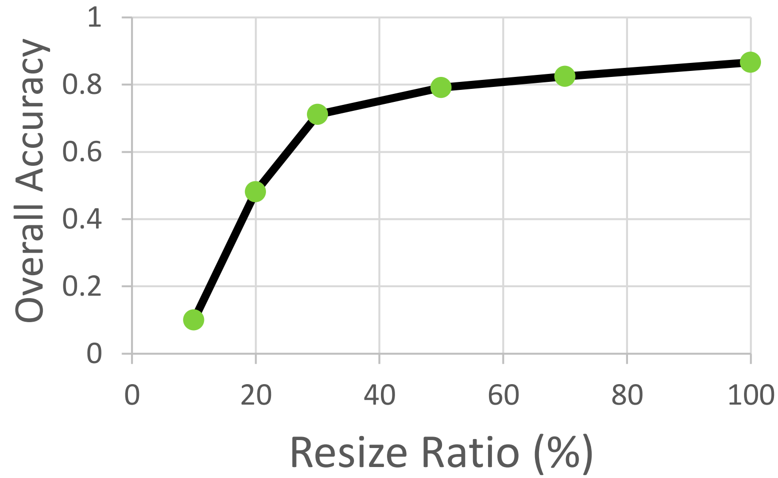
$I_v <= R_v$ for all $v \epsilon V$

reducing the communications cost

impact of resizing/subsampling on accuracy



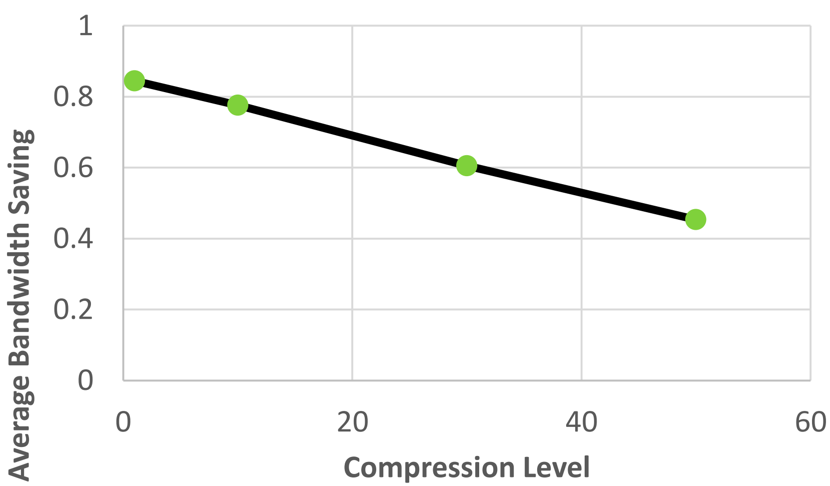


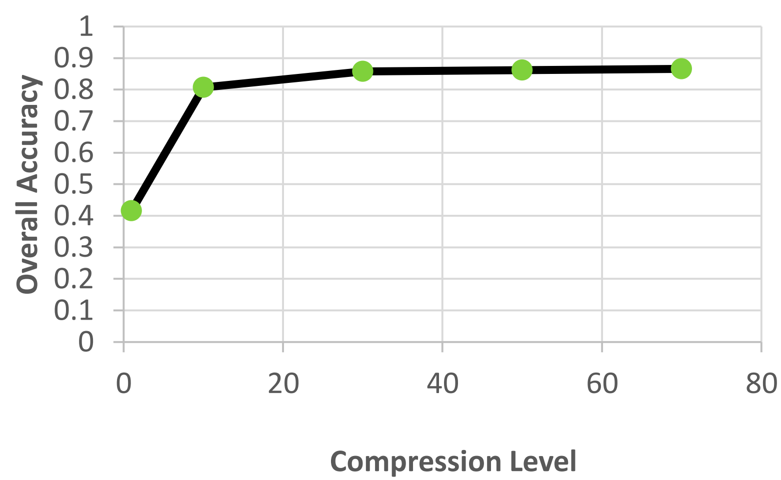


Microsoft

reducing the communications cost

impact of compression on accuracy



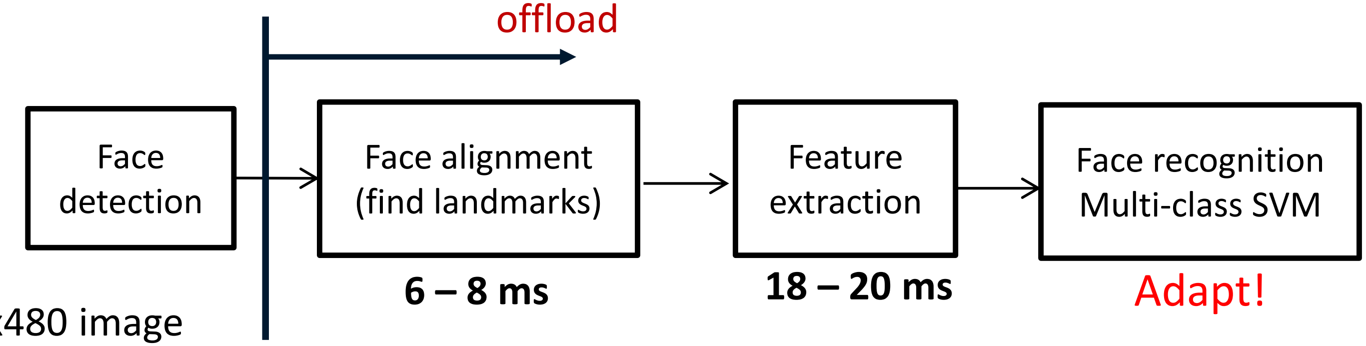


what's the optimal compression level and image size?

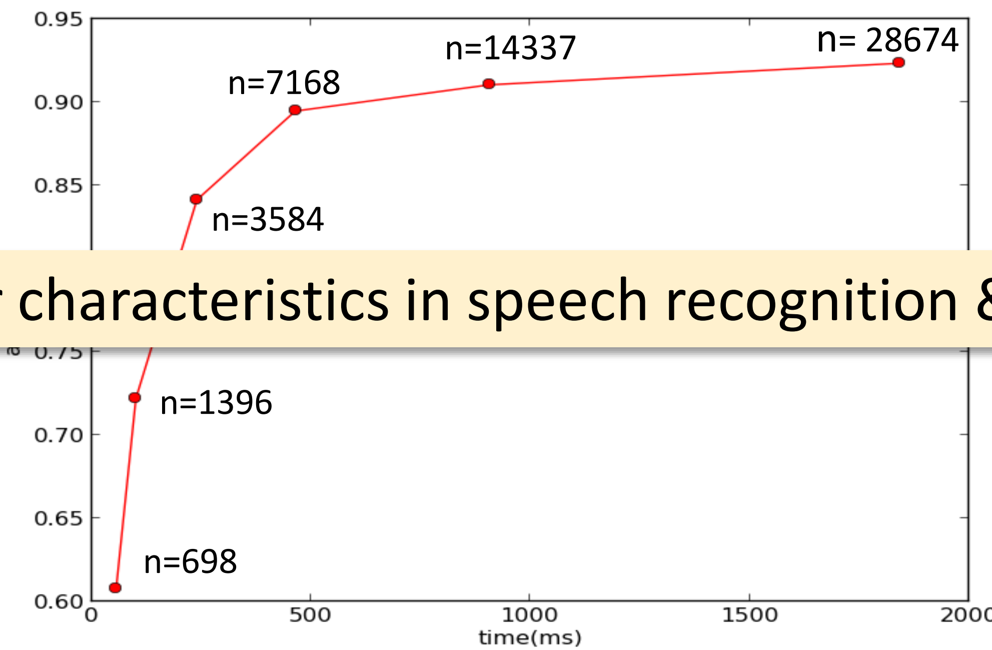
Microsoft

reducing latency

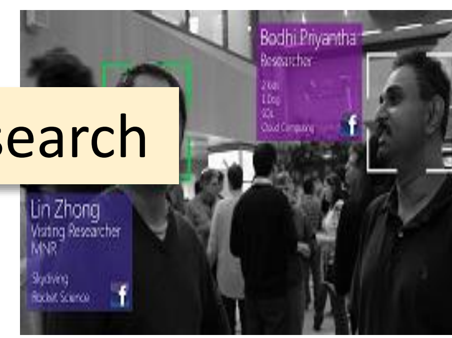
the lower the latency, the better the results



For a 640x480 image
Client: 766ms
Server: 138ms



similar characteristics in speech recognition & search

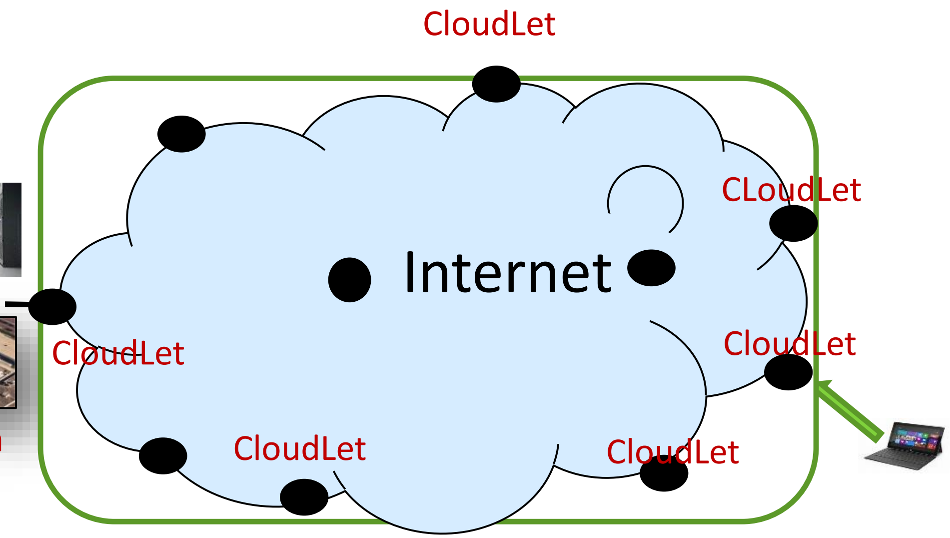


Face prediction Time

reduce latency to the clouds via cloudlets

build an extensive infrastructure of micro datacenters (tens of servers with several TBs of storage, $30K-$200K/each) & place them in strategic locations around the internet

Mega Data Center



+

tunnel with strong SLAs from selected CloudLet to DCs

cloudlets (micro datacenters)

definition -
a resource rich computing infrastructure with high-speed Internet connectivity to the cloud.

the mobile device uses this infrastructure to augment its capabilities and to enable applications that were previously not possible

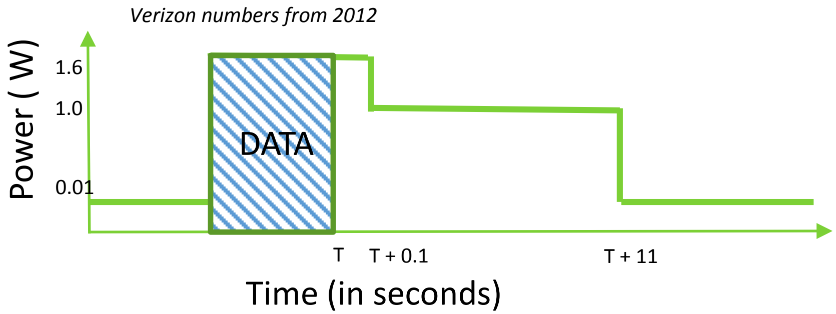
cloud offloading without and with mDCs

mDCs can help with battery life in other ways

fast dormancy

network latencies negatively impact battery life:

- LTE consumes > 1.5W when active
- LTE chip active for ~10 secs of extra tail time (1W power)



....but how did we get here

a bit of context/history…4 years ago

The New York Times

Customers Angered as iPhones Overload AT&T

By JENNA WORTHAM
Published: September 2, 2009

The New York Times

DIGITAL DOMAIN

AT&T Takes the Blame, Even for the iPhone's Faults

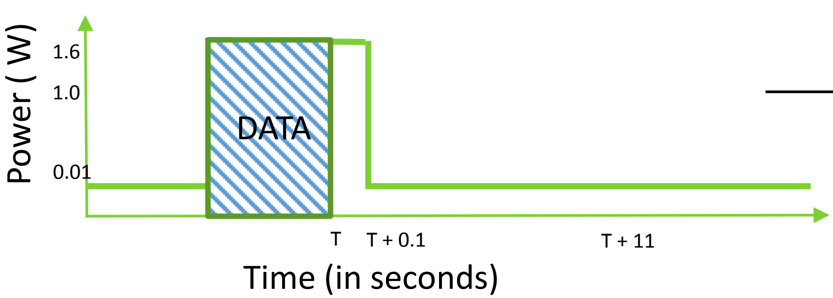
By RANDALL STROSS
Published: December 12, 2009

Report: AT&T Reputation Tarnished by iPhone Flaws

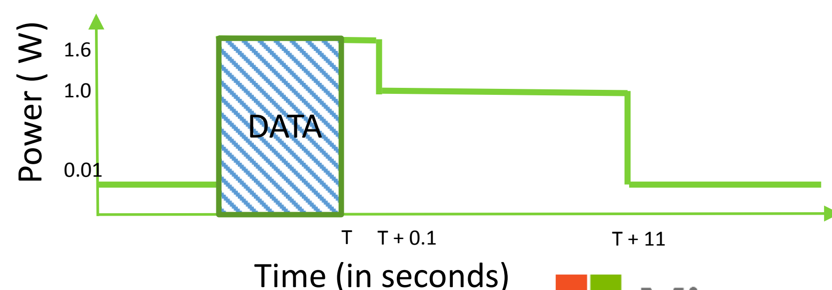
By Tony Bradley, PCWorld Dec 14, 2009 2:01 PM

original design: bring radio to low power state immediately

Verizon numbers from 2012



Mobile Operator (MO) requirement: keep LTE chip **active for ~10 sec**. of extra tail time (to reduce the signaling load)



Microsoft

mDCs can help with battery life as well

fast dormancy

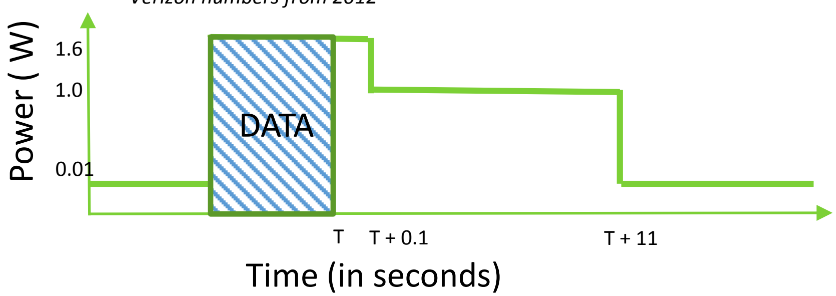
network latencies negatively impact battery life:

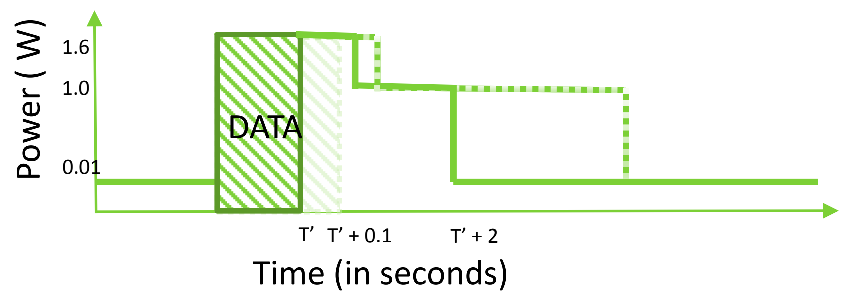
- LTE consumes > 1.5W when active
- LTE chip active for ~10 secs of extra tail time (1W power)

with mDCs:

- faster transfers => less time in highest power state
- UE can aggressively enter lowest power state

Verizon numbers from 2012





Energy savings / transfer: 1.6W*speedup + 1W*9sec = 10.6J (assuming speedup of 1 second)

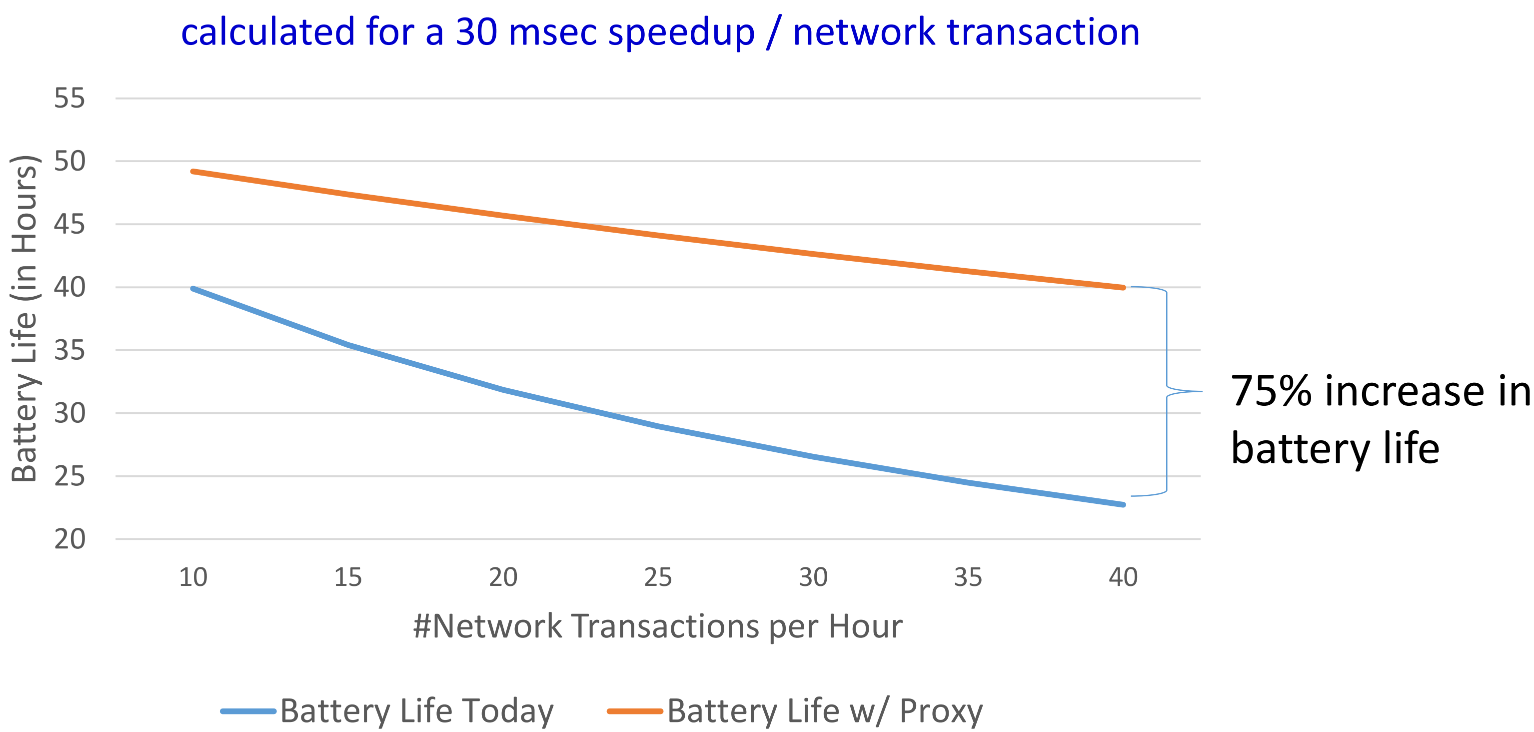
for 20 network transfers per hour (notifications, email, etc.), with 1 sec speedup, energy savings per 24 hr. = 6624 J

→ Saving of **26%** in a 1500 mAH cell phone battery*

* Samsung Standard LI-ION battery with rating of 1500mAh/3.7Vdc

especially good for mobile

battery life improvement



these types of saving occur across the board for all battery types and all types of mobile devices

* Samsung Standard LI-ION battery with rating of 1500mAh/3.7Vdc

Microsoft

conclusions

take a holistic view to energy management is where the next big gains will come

- scenario + algorithms + systems software + network + hardware + ….

in the real-time visual analytics case:

- gated imaging + cost-sensitive classification for (adaptive) detection + proportional-power imaging + cloudlets minimizes processing cost reduces battery consumption

Microsoft

Thanks!