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
continuous mobile vision
reality vs. movies

Steve Mann (early 90s)
iRobot (2004)
Mission Impossible 4 (2011)
COBOT, CMU (2013)
C-3PO (1977)

Victor Bahl, MSR
perennial challenges

- 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 SenseCam for memory assistance

Augmented Reality

Where am I?

Victor Bahl, MSR
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...

Wi-Fi
500mW
5Mbps
70 mW avg

WWAN
700mW
10GB/mo
Phone
700 mW avg
150 g

Cloud
.01 server

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
power consumption of a typical image sensor

- Reduce frame rate

- Reduce resolution

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Frame Rate</th>
<th>Power Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 MP</td>
<td>5 fps</td>
<td>345 mW</td>
</tr>
<tr>
<td>1 MP</td>
<td>5 fps</td>
<td>250 mW</td>
</tr>
<tr>
<td>0.3 MP</td>
<td>5 fps</td>
<td>232 mW</td>
</tr>
<tr>
<td>0.3 MP</td>
<td>15 fps</td>
<td>245 mW</td>
</tr>
<tr>
<td>0.3 MP</td>
<td>30 fps</td>
<td>268 mW</td>
</tr>
<tr>
<td>1 MP</td>
<td>15 fps</td>
<td>295 mW</td>
</tr>
</tbody>
</table>

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

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

Profiled 5 image sensors from 2 manufacturers

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

Victor Bahl, MSR
digging deeper (1 MP, 5 fps)

\[ E_{frame} = P_{active} T_{active} + P_{idle} T_{idle} \]
reduce power by reducing pixel readout time

one pixel is read out per clock period

\[ E_{frame} = P_{active} T_{active} + P_{idle} T_{idle} \]

\[ T_{active} = \frac{N}{f} \]

Number of Pixels divided by Clock Frequency
reducing pixel count (N)

active readout  active readout

region-of-interest (windowing)
scaled resolution (pixel skipping)

Power
Time

Power
Time
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
reduce power by adjusting clock frequency

Adjust clock frequency to minimize power

\[ T_{\text{active}} = \frac{N}{f} \]

Adjust this

At low frame rates, run the clock as slow as possible
summarizing power reduction techniques for image sensors

- reduce $T_{\text{active}}$ & increase $T_{\text{idle}}$
  - decrease frame rate
  - reduce total pixel readout time (by reducing N)
  - adapt clock frequency

- instead of idle-ing put sensor in *standby state*

- reduce $P_{\text{active}}$ (not covered in this talk, see paper)
applying these techniques

unoptimized

aggressive standby & clock optimization
stated another way
actual numbers

---

*Power Consumption vs. Number of Pixels Accessed for Various Proportional-Power Optimizations*
### Impact on Vision Algorithms

#### Image Registration

<table>
<thead>
<tr>
<th></th>
<th>Image Registration Success</th>
<th>Person Detection Success</th>
<th>Actual Power Reduction with software assist</th>
<th>Estimated Power Reduction with hardware assist</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Full Resolution</strong></td>
<td>99.9%</td>
<td>94.4%</td>
<td>51%</td>
<td>84%</td>
</tr>
<tr>
<td><strong>Frame Rate: 3 FPS</strong></td>
<td><strong>95.7%</strong></td>
<td>83.3%</td>
<td><strong>95%</strong></td>
<td>98%</td>
</tr>
<tr>
<td><strong>30% Window</strong></td>
<td>96.5%</td>
<td>77.8%</td>
<td>63%</td>
<td>91%</td>
</tr>
<tr>
<td><strong>Subsampled by 2</strong></td>
<td>91.8%</td>
<td>72.2%</td>
<td>71%</td>
<td>94%</td>
</tr>
</tbody>
</table>

### Image Registration

- 480 x 270

#### Person Detection

- [Image of person detection]

---

*Note: The table above illustrates the success rates and power reduction with software and hardware assists for different resolution settings.*
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).

data gathering application
• ~5 video frame per second
• sync with timestamps
• collect all possible sensors

Seungyeop Han’s version

thermal camera
• 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
analyze the thermal camera data

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
detect objects

T1 > 30°C
reject

30 > T2 > 23
reject

T3 < 32
reject

T1
T3

0: 320x240
1: 80x60
2: 32x24
3: 16x12

Microsoft
reject windows with no objects

Ensemble structure allows many windows may be rejected by few gating pixels
so can we use lower power sensors to filter out uninteresting frames?

input visual stream

low-power sensors

is the frame interesting?

YES
challenge: gating thresholds vary with time

motivates online re-estimation of adaptive detectors
results

- Adaptive, no online
- Adaptive, online
- Not adaptive, gated
- RGB only (Viola/Jones)

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

Est. power consumed under various schemes
Assume 40nJ/read, 5nJ/instruction executed
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?
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?
when to offload?

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

decision engine:
partition a running app

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

Example – Maximize:

\[
\sum_{v \in V} ( I_v \times E_v ) - \sum_{(u,v) \in E} \left( |I_u - I_v| \times C_{u,v} \right)
\]

energy saved cost of offload

Such that:

\[
\sum_{v \in V} ( I_v \times T_v ) + \sum_{(u,v) \in E} \left( |I_u - I_v| \times B_{u,v} \right) \leq Lat.
\]

execution time time to offload

and

\[ I_v \leq R_v \text{ for all } v \in V \]

- Vertex: method annotated with computation energy and delay for execution
- Edge: method invocation annotated with total state transferred
reducing the communications cost
impact of resizing/subsampling on accuracy

Overall Accuracy

Average Bandwidth Saving

Resize Ratio (%)

0 20 40 60 80

0 0.2 0.4 0.6 0.8 1

0 20 40 60 80 100

0 0.2 0.4 0.6 0.8 1

Resize Ratio (%)
reducing the communications cost
impact of compression on accuracy

What’s the optimal compression level and image size?
reducing latency
the lower the latency, the better the results

Face detection  →  Face alignment (find landmarks)  →  Feature extraction  →  Face recognition
Multi-class SVM

6 – 8 ms  →  18 – 20 ms

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
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

original design: bring radio to low power state immediately

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

**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

calculated for a 30 msec speedup / network transaction

75% increase in battery life

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
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
Thanks!