Power-Aware Gaming


9/24 12:00
SONY
ゲーム
UMD®
POWER
HOLD
HOME
VOL
SELECT
START


N81
Music
Now playing
My Song
Music
Podcasts
Playlists
Shuffle all songs
Options
Close
NOKIA


X
Y
A
B
START
SELECT


VAIO

major power consumers:

LCD

CPU

WNIC

e.g., on an iPAQ

LCD (~1W)
CPU (1-3W)
WNIC (1.4W)

"Integrated Power Management for Video Streaming to Mobile Handheld Devices"
Shivajit Mohapatra, Radu Cornea, Nikil Dutt, Alex Nicolau & Nalini Venkatasubramanian
ACM Multimedia 2003

Reducing WNIC's Power Consumption

State	Documented	Measured
WaveLAN - suspended	0W	0W
WaveLAN - receive	1.48W	1.52W
WaveLAN - transmit	3.00W	3.10W

Table 1: Power Requirements of the Lucent WaveLAN PCMCIA Wireless Ethernet card.

Power Management Techniques for Mobile Communication

Robin Kravets
College of Computing
Georgia Institute of Technology
801 Atlantic Drive
Atlanta, GA 30309
robink@cc.gatech.edu

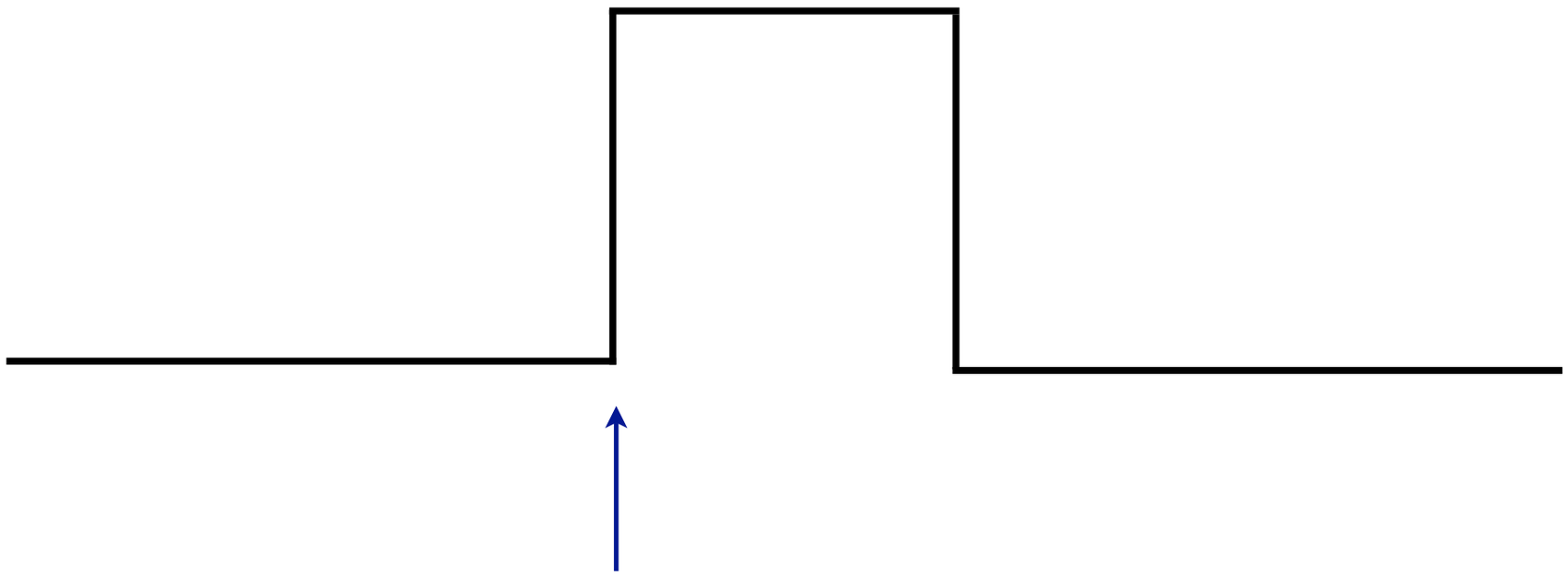
P. Krishnan
Bell Labs, Lucent Technologies
101 Crawfords Corner Rd.
Holmdel, NJ 07733-3030
pk@research.bell-labs.com

Assume two modes:

active and **suspend**

Idea: suspend WNIC and suppress updates when appropriate

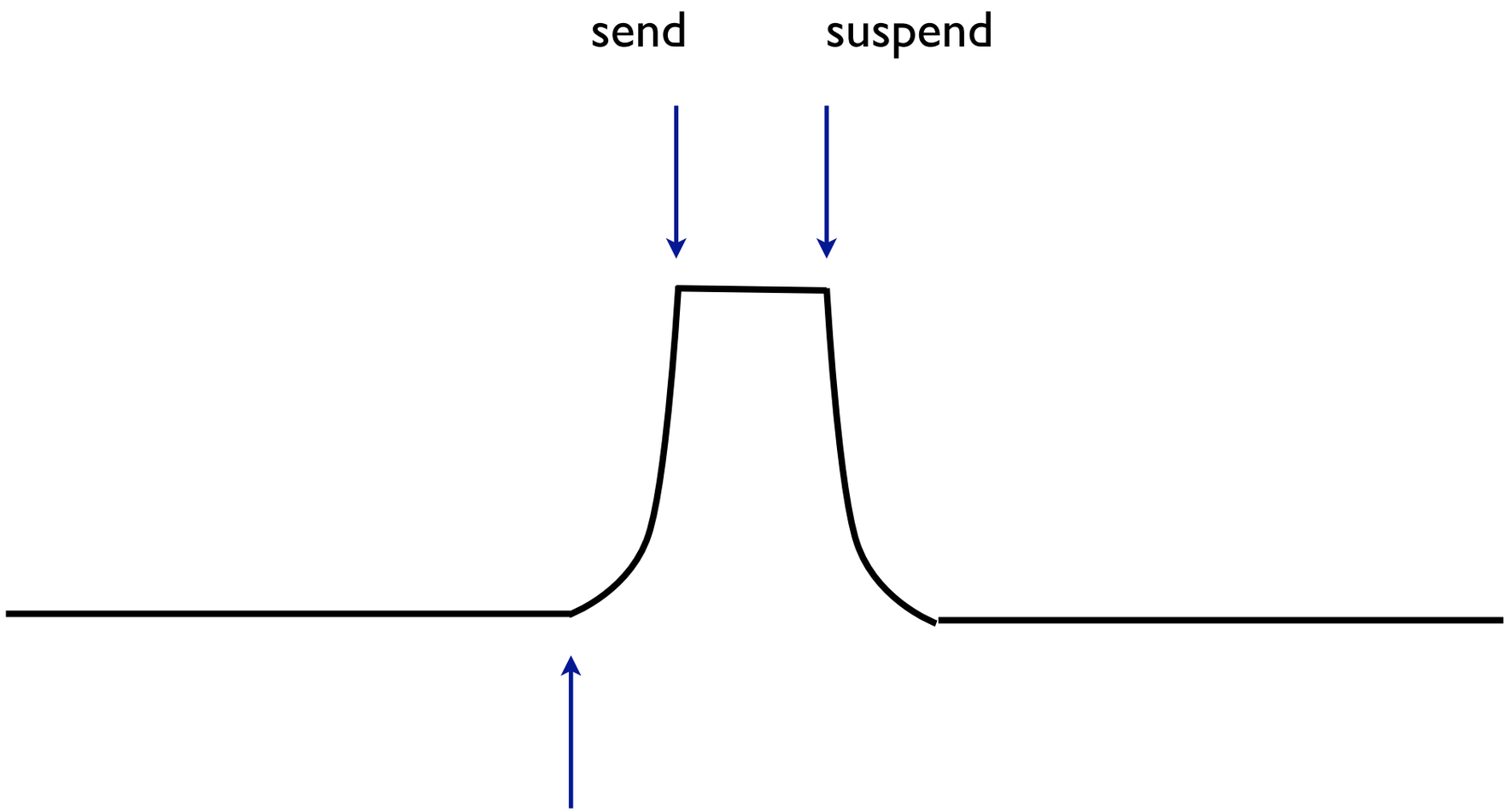
Recall: can suppress updates when diff in position is below threshold



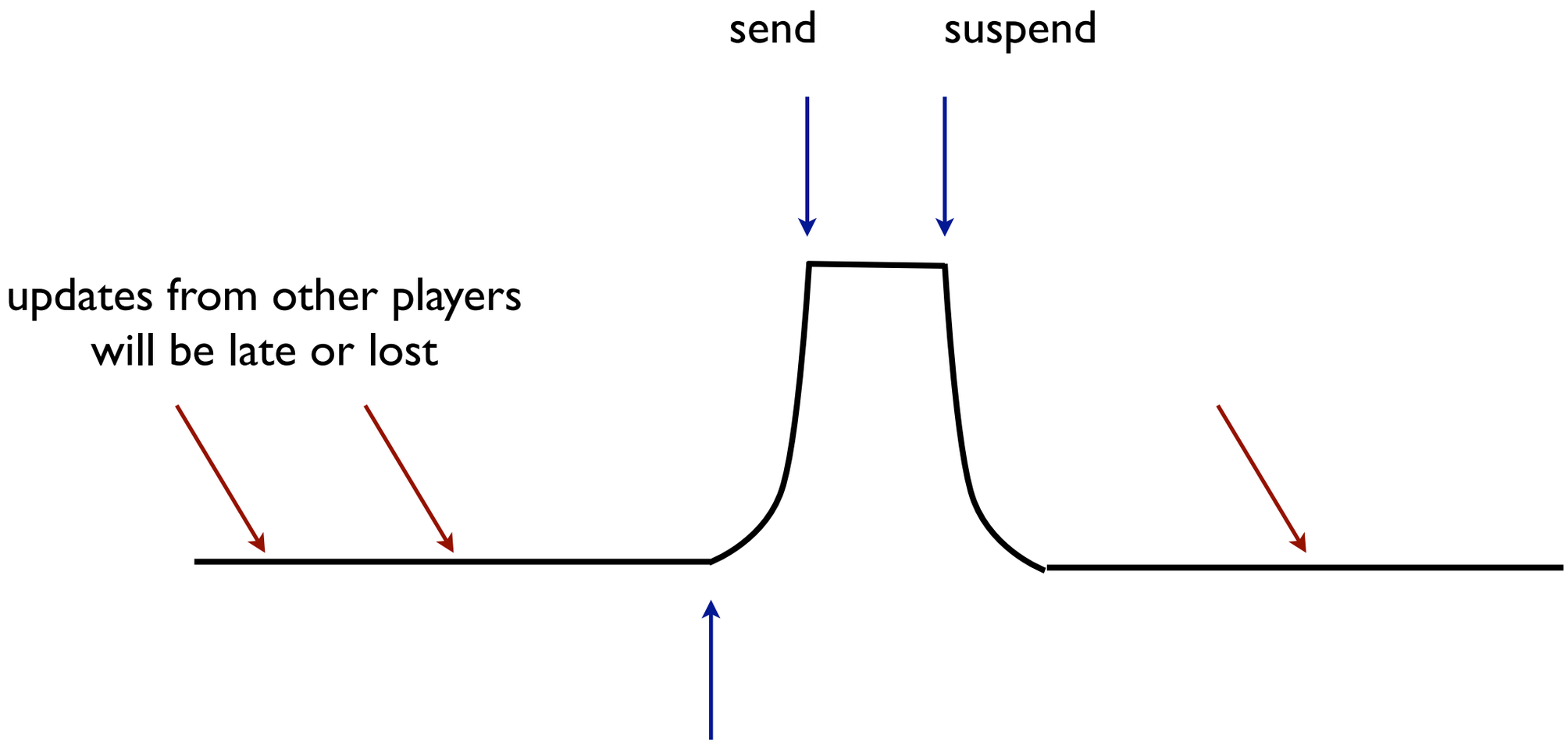
Exceed DR threshold: switch to active, send, suspend again

in practice, mode switching takes time (50-100ms, up to 600ms observed)

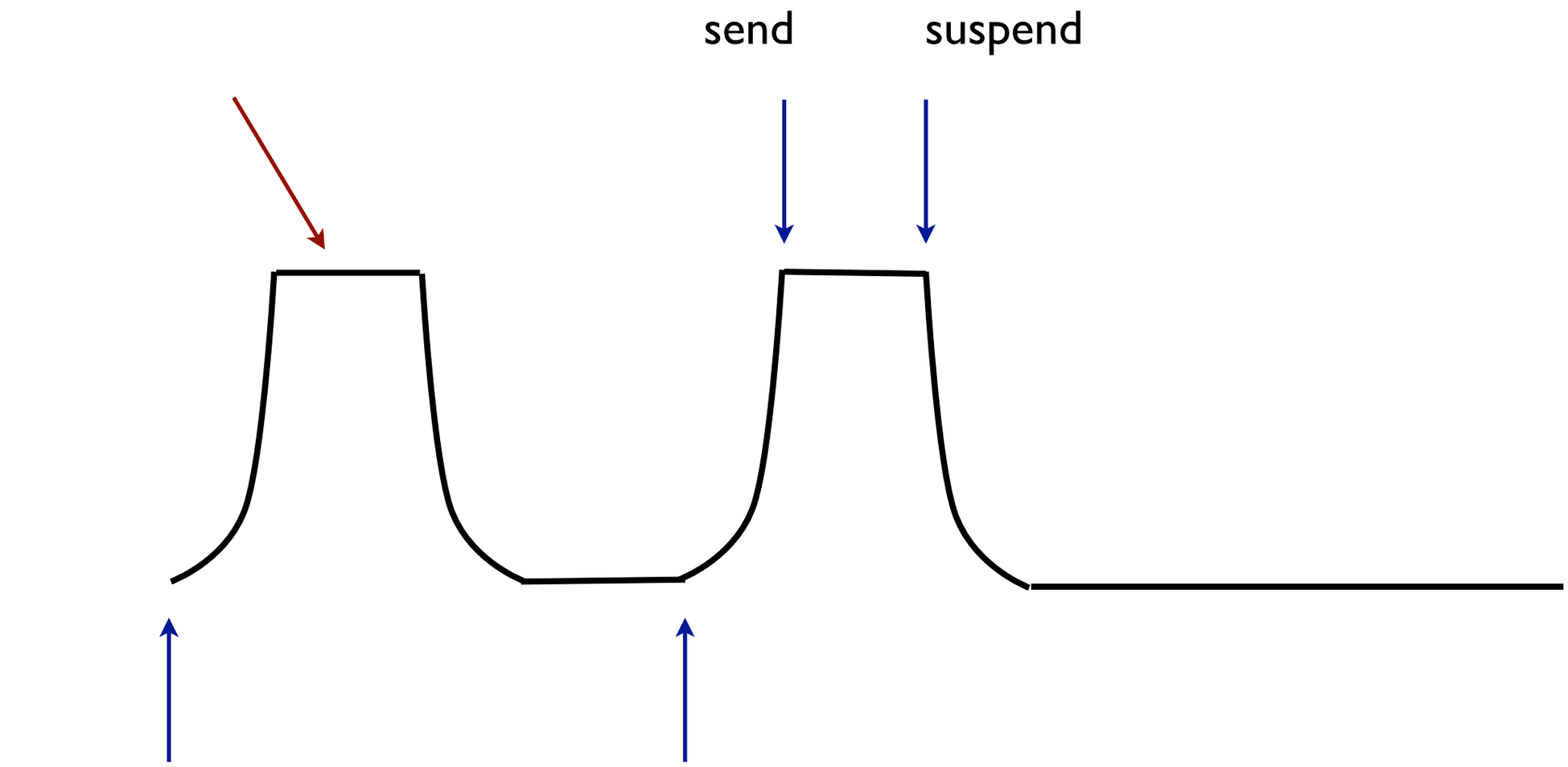
Reduce DR threshold to compensate for switching time



Almost exceed DR threshold:
switch to active


send
suspend
updates from other players
will be late or lost
Almost exceed DR threshold:
switch to active

Consider a client/server model with periodic server updates


send
suspend
Expecting server update
Almost exceed DR threshold:
switch to active

Note: not practical unless server updates rarely or mode switching latency reduces

Problem: Not all updates are predictable even in C/S architecture. More so in P2P architecture.

Cannot predict opponents movement

Reducing WNIC's Power Consumption

remain open!

Reducing CPU's Power Consumption

Game loop:

compute states

render scene

get input

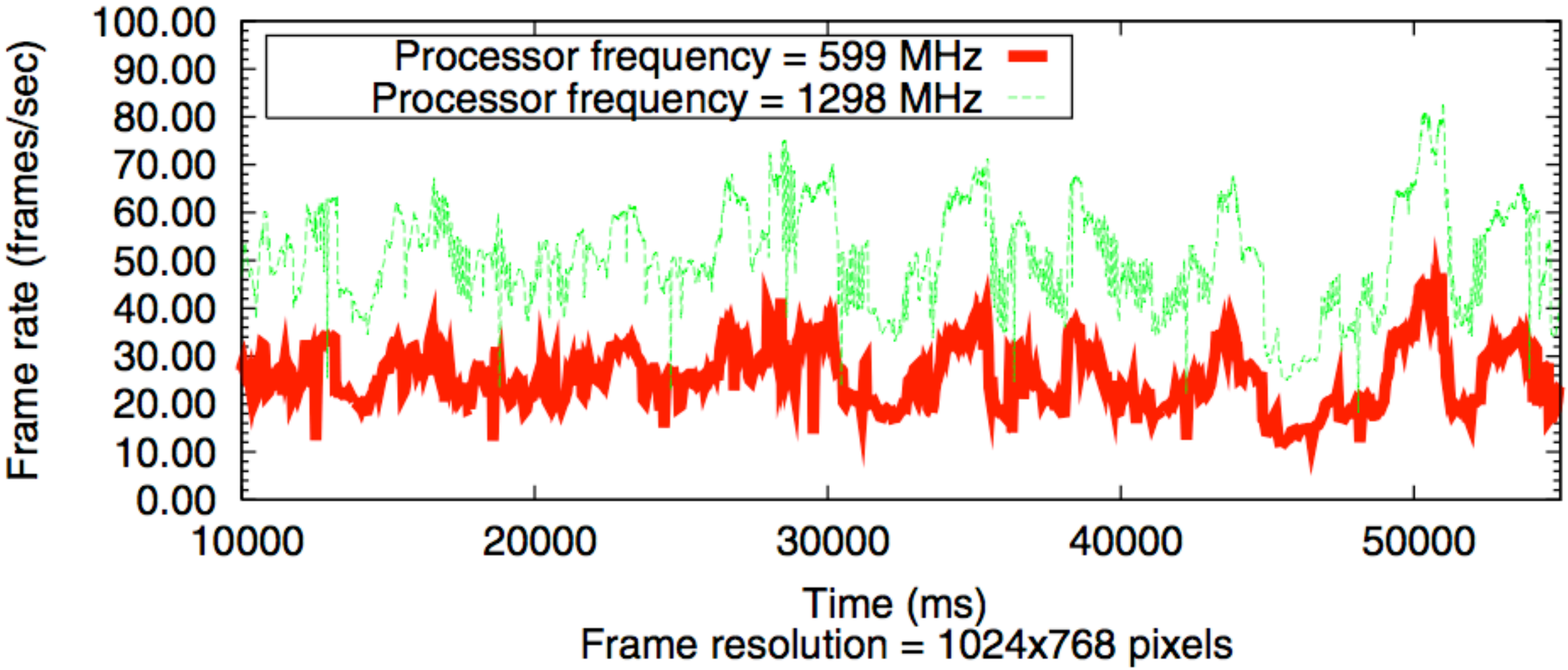
faster CPU : higher frame rate

simpler scene : higher frame rate

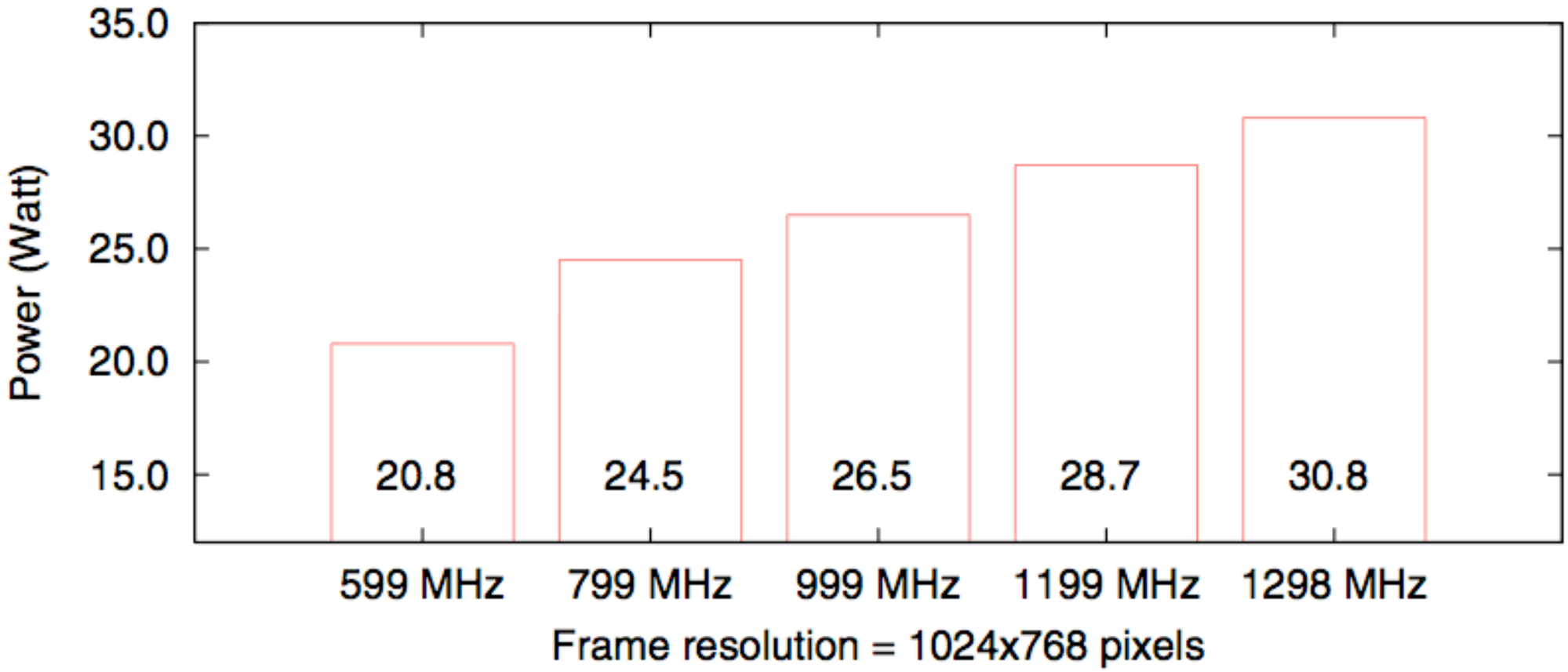
Idea: slow down CPU when scene is simpler (achieving constant frame rate)

“slow down” CPU : dynamic voltage and frequency scaling

$$\text{Power} \propto V^2 f$$



Frame rate of Quake II


35.0
30.0
25.0
20.0
15.0
Power (Watt)
20.8
24.5
26.5
28.7
30.8
599 MHz
799 MHz
999 MHz
1199 MHz
1298 MHz
Frame resolution = 1024x768 pixels

let

c be the estimated cycle requirement to render a scene

f be the min acceptable frame rate

let

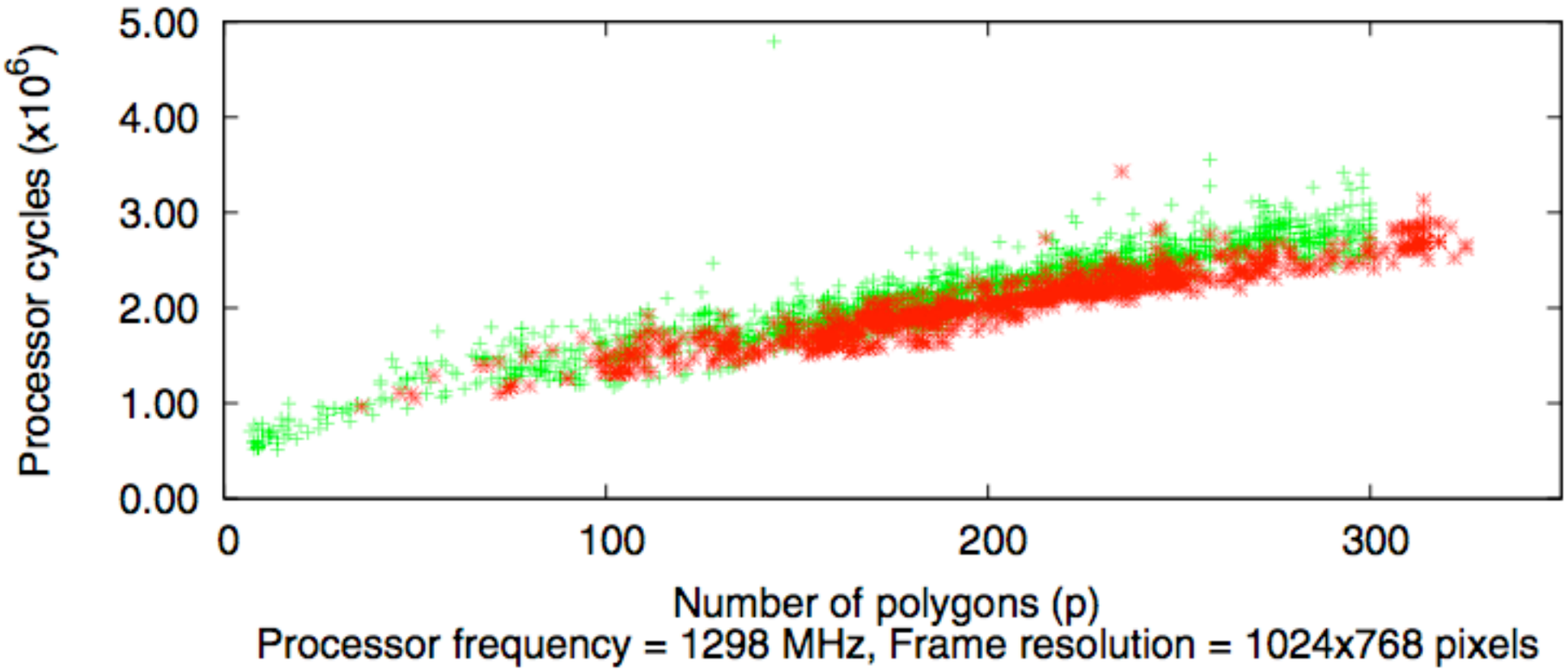
c be the estimated cycle requirement to render a scene

f be the min acceptable frame rate

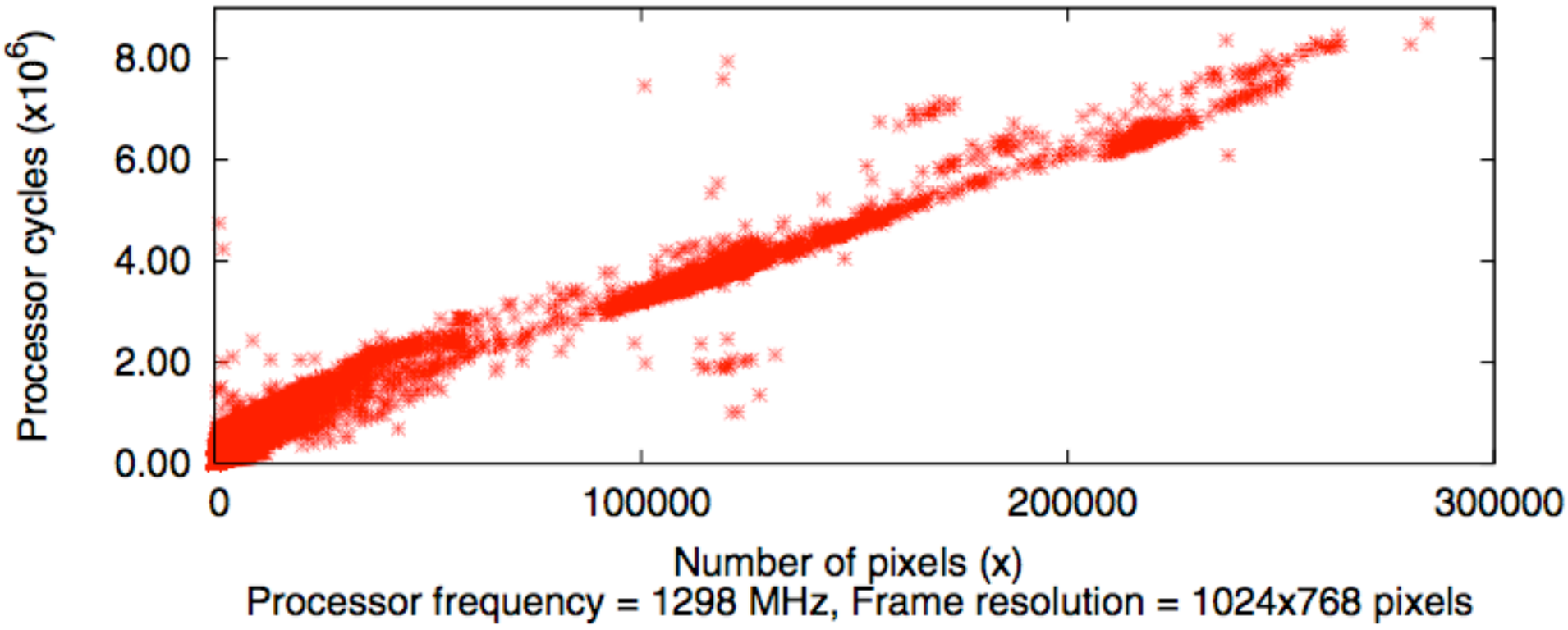
set CPU frequency to **cf** (rounded up to the nearest supported frequency)

How to estimate cycle requirement for a scene?

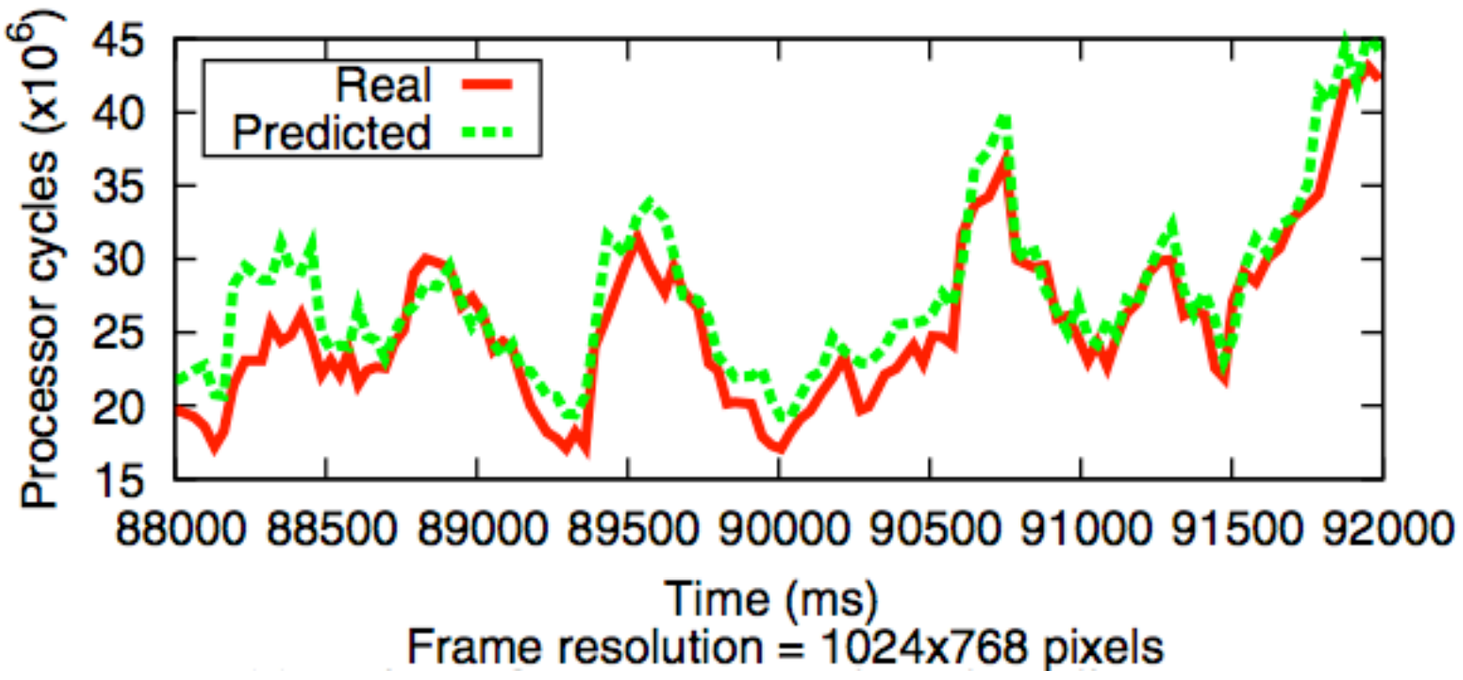
Look at the structure of a scene:
number of polygons,
number of objects ..

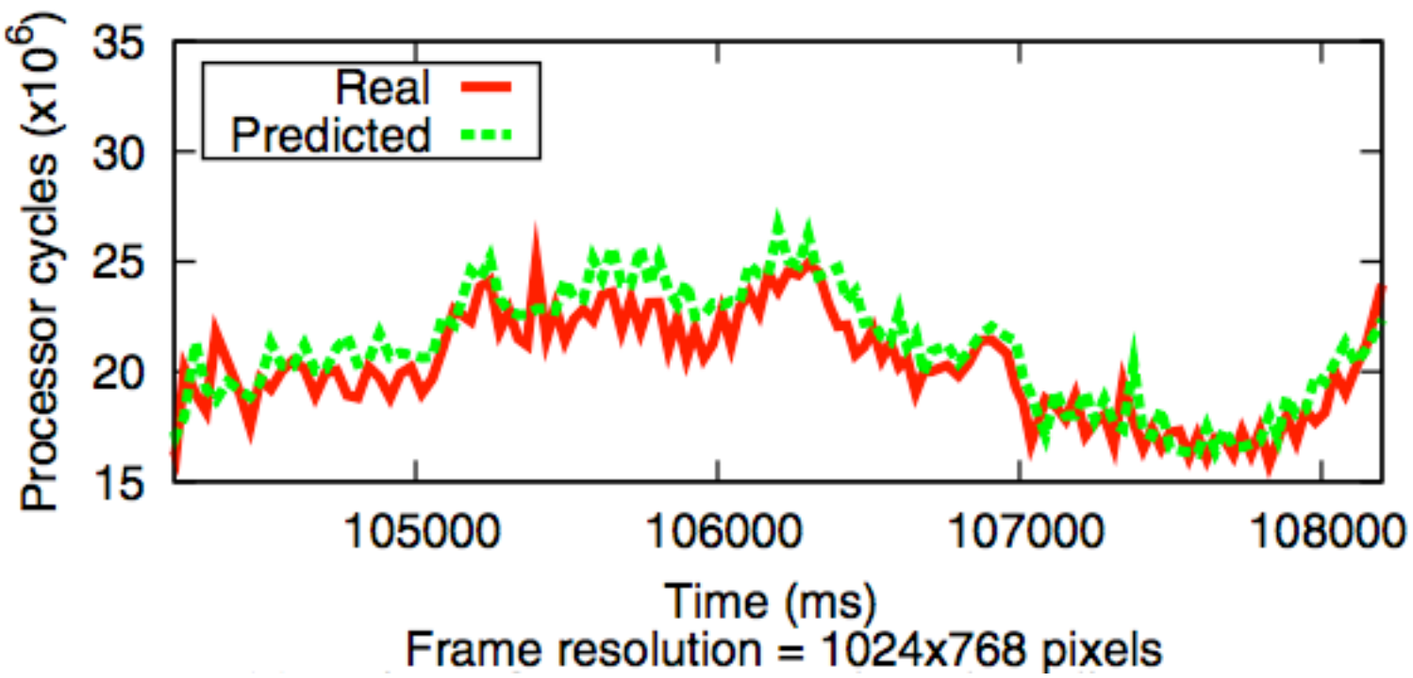


Rasterization workload for "brush model" (walls, floor, objects etc)



Rasterization workload for “alias model” (monsters, soldiers, weapon etc)


Processor cycles (x10^6)
45
40
35
30
25
20
15
Real
Predicted
88000 88500 89000 89500 90000 90500 91000 91500 92000
Time (ms)
Frame resolution = 1024x768 pixels


Processor cycles (x10^6)
35
30
25
20
15
Real
Predicted
105000
106000
107000
108000
Time (ms)
Frame resolution = 1024x768 pixels

Save power by 15% on a PDA running at 5 fps

But real games use GPU,
not CPU to render

Reducing Mobile GPU's Power Consumption

remain open!