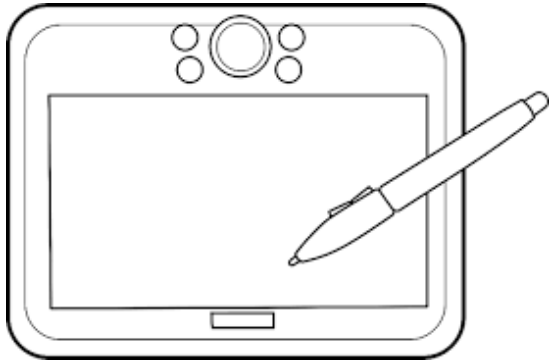
Adaptive Touch Sampling for Energy-Efficient Mobile Platforms

Kyungtae Han
Intel Labs, USA

Alexander W. Min, Dongho Hong, Yong-joon Park
Intel Corporation, USA

April 16, 2015

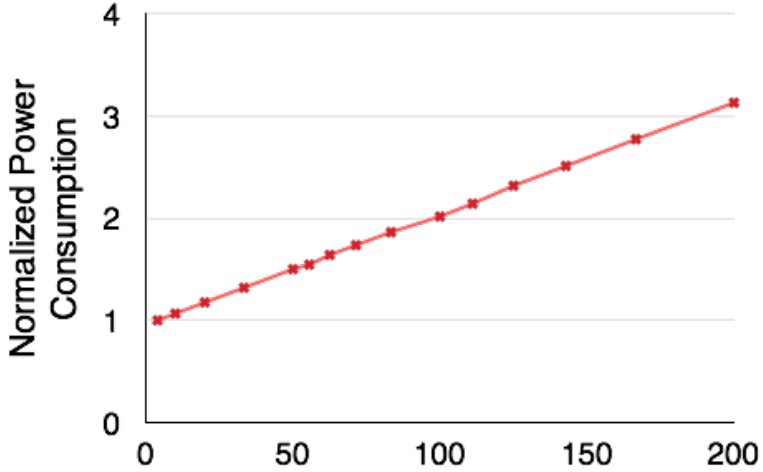
Touch Interface in Today's Mobile Platforms



To enable interactive and responsive applications

Challenges in Mobile Touch Systems

- Power-responsiveness tradeoff – *Find optimum responsiveness for users & apps*



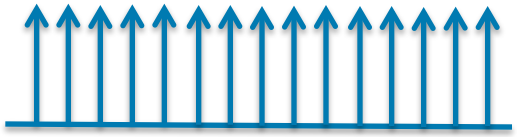
Today's approach: One optimal frequency for all users and apps

Sample power consumption in touch controller and panel

Our Approach: Adaptive Sampling

- Intelligently adapts the touch responsiveness "*on-the-fly*" based on user touch behavior

Fast touch behavior



Fast sampling rate

Slow touch behavior

Slow sampling rate

Result: Up-to 44% power savings in touch controller and panel

Outline

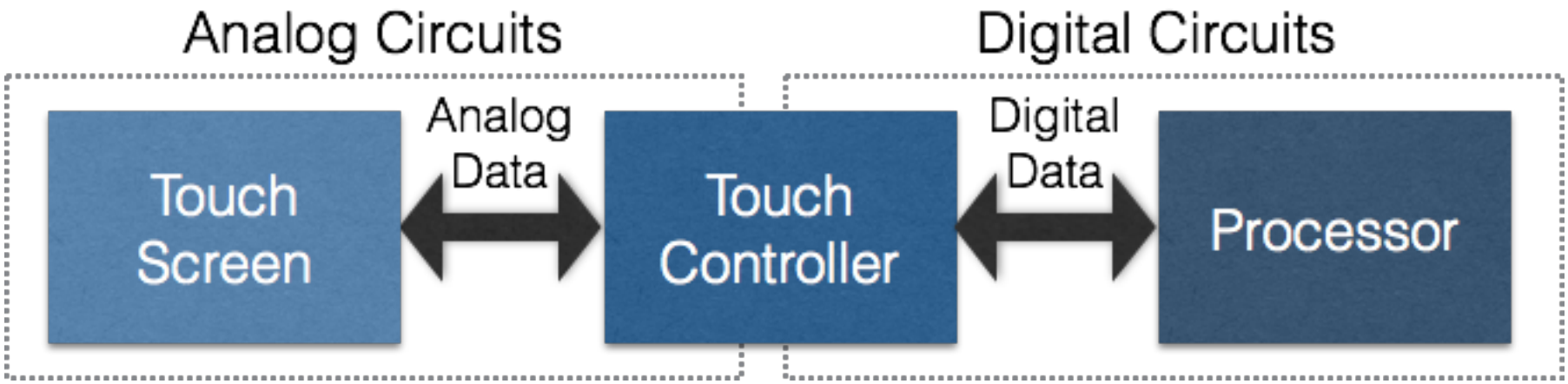
Background

Our Approach

Evaluation

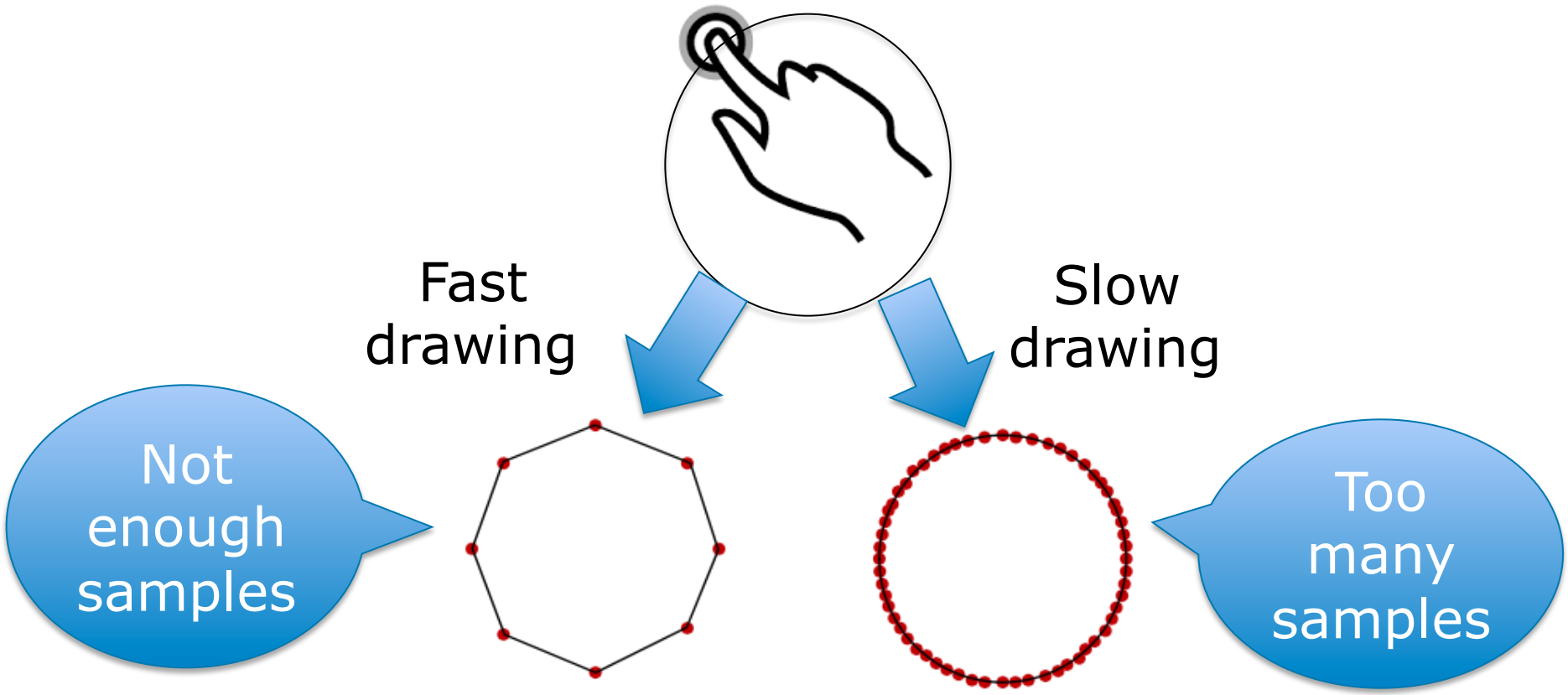
Conclusion

Touch Screen System



User touch events are delivered to Processor at the rate of touch scan interval (frequency)

Touch Output of Circle Drawing



Conventional Approach: Fixed Scan Interval

Outline

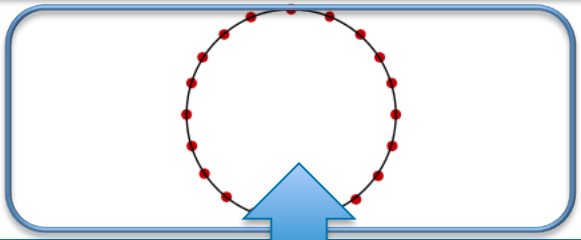
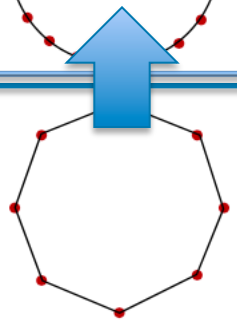
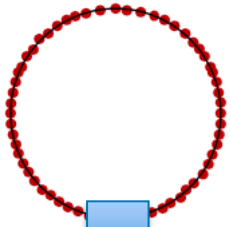
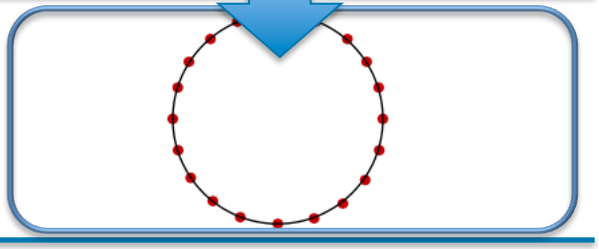
Background

Our Approach

Evaluation

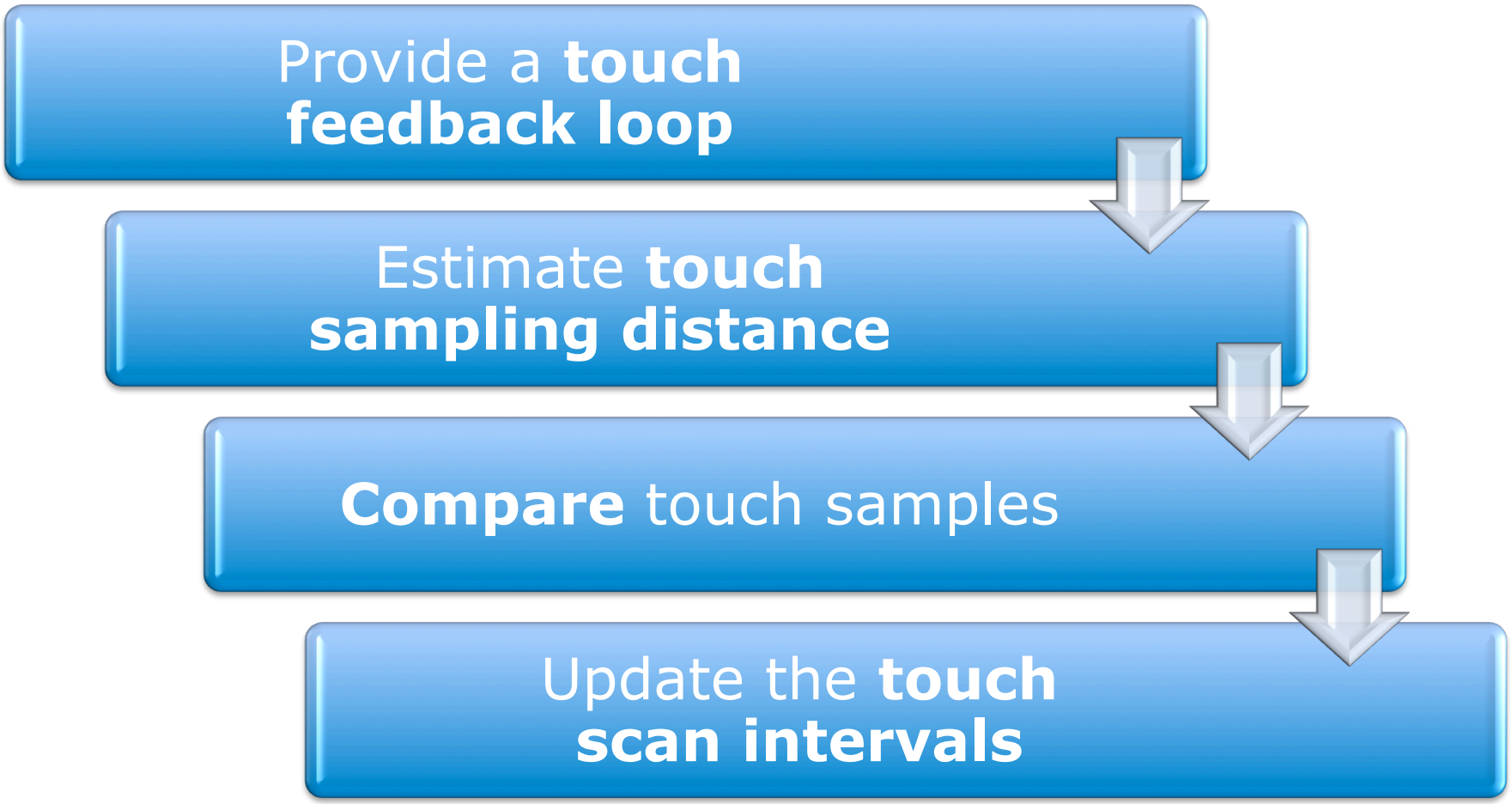
Conclusion

Adaptive Scan Interval

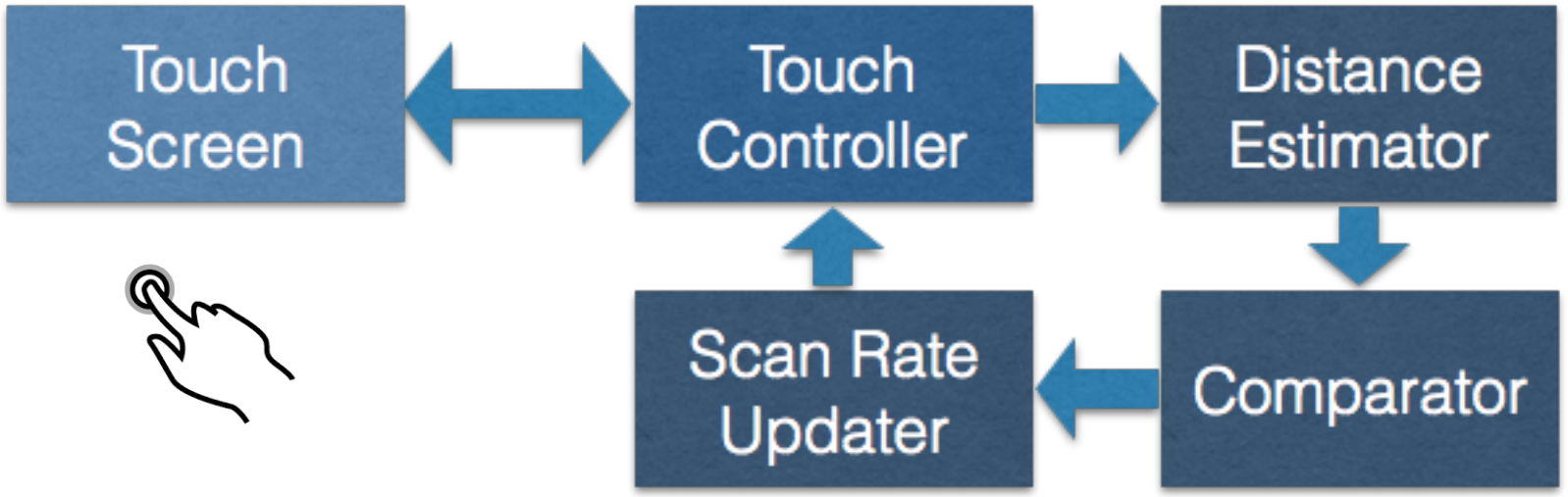
Scan Interval	Fast drawing	Slow drawing
Short		
Medium		
Long		

Our Approach: Scan Interval is dynamically changed based on user touch behavior

Adaptive Touch Sampling Approach

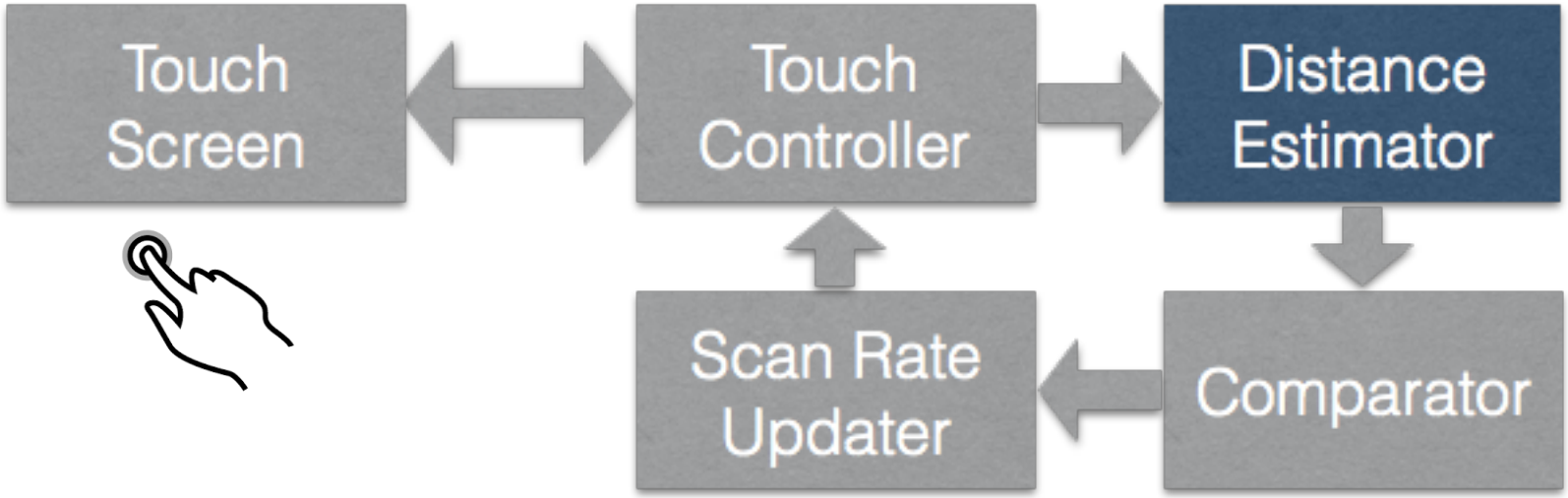


Proposed Adaptive Touch Scan Rate Architecture



Feedback loop keeping the same distance between traveled and reference

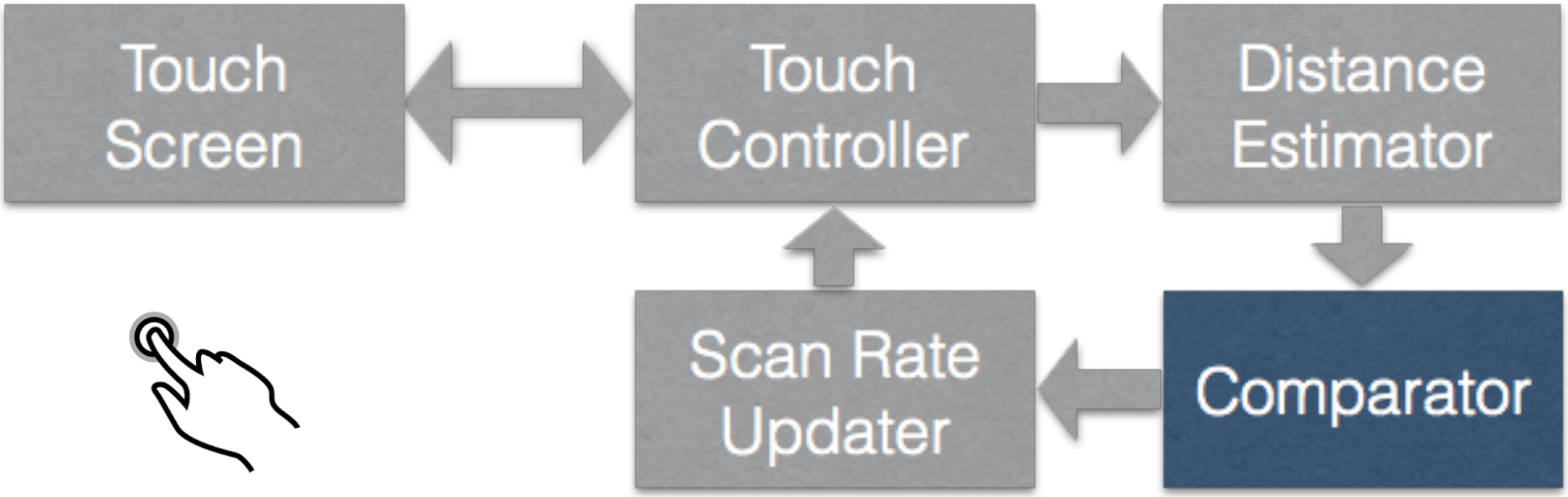
System Architecture – Distance Estimator



$$\Delta d_i = |x_i - x_{i-1}| + |y_i - y_{i-1}|$$

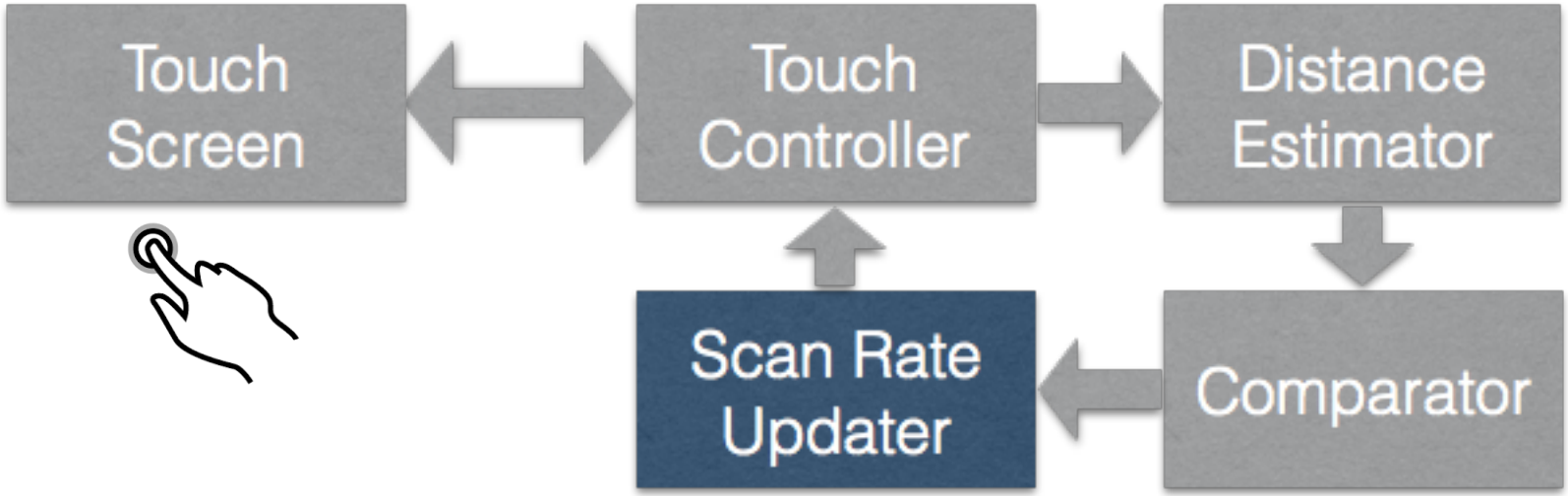
Calculate the sample distance between two consecutive touch samples

System Architecture – Comparator



Calculate the error between sample distance and pre-defined reference distance

System Architecture – Scan Rate Update



$$u(t) = K_p e(t) + K_i \int_0^t e(r)dr + K_d \frac{d}{dt} e(t)$$

Update the touch scan interval based on error components with their associated weights

Adaptive Touch Scan Rate Algorithm

Algorithm 1 ADAPTIVE TOUCH SCAN RATE ALGORITHM

```
set Kp, Ki, Kd, initial integral, initial scan interval, max scan interval,
    min scan interval and target distance
while (true)
    scan interval = initial scan interval
    integral = initial integral
    previous error = 0
    while (consecutive touch events)
        calculate Δd
        error = target distance - Δd
        integral = integral + error × scan interval
        derivative = (error - previous error) / scan interval
        scan interval = Kp × error + Ki × integral + Kd × derivative
        if scan interval > max scan interval then
            scan interval = max scan interval
        if scan interval < min scan interval then
            scan interval = min scan interval
        previous error = error
        update touch controller with scan interval
    end while
end while
```

Estimate touch sampling distance

Compare touch samples

Calculate scan interval

Update the touch scan interval

Touch Feedback loop

Outline

Background

Our Approach

Evaluation

Conclusion

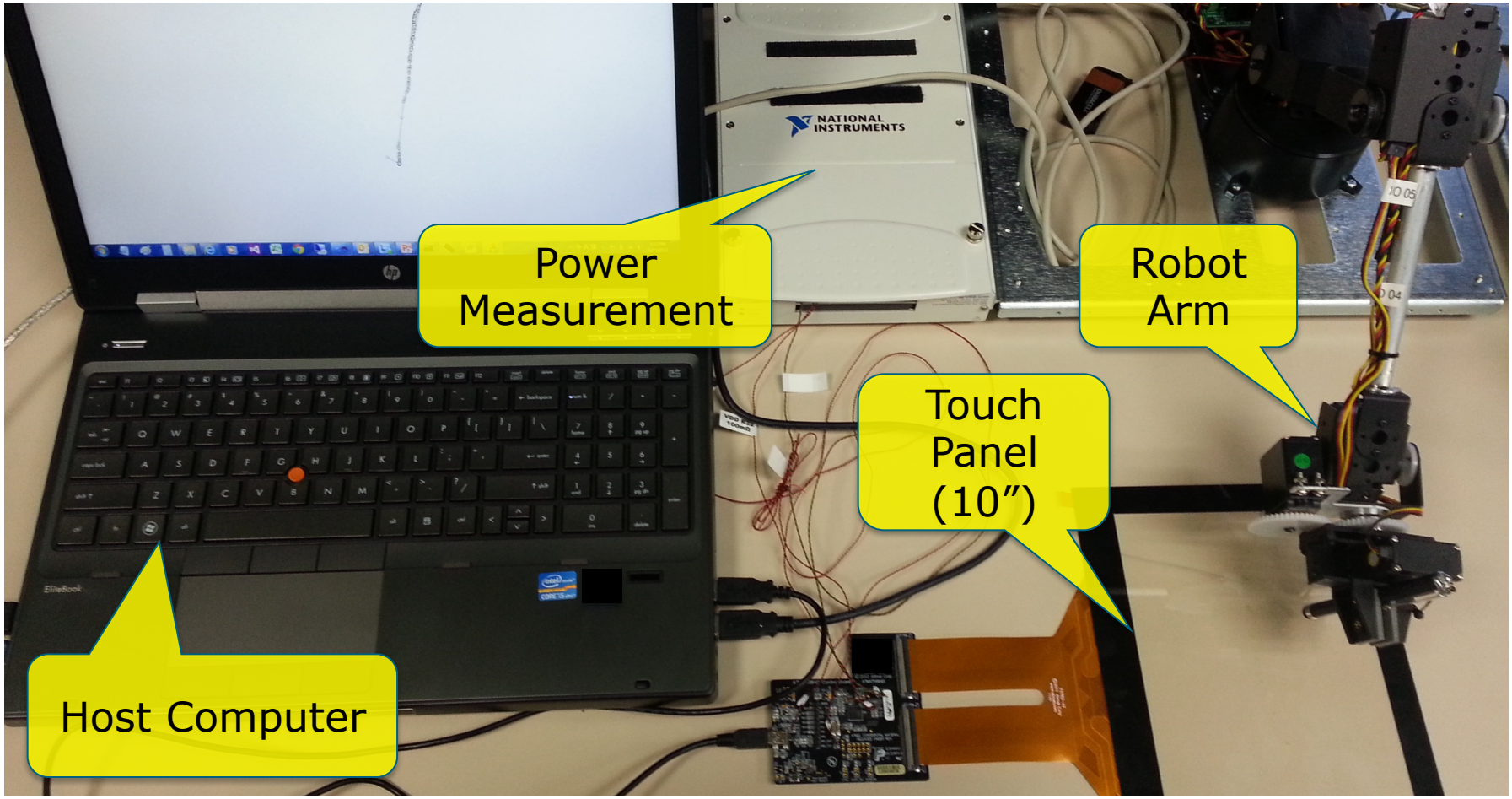
Touch Drawing Simulations

- Use robot arm for controlled experiments of touch drawing
- Fast motion drawings

  23.33 cm/sec, 40.00 cm/sec in Avg.
- Slow motion drawings

  2.75 ~ 12.73 cm/sec in Avg.

Experiment Setup



Fast Touch Drawing

Today Approach

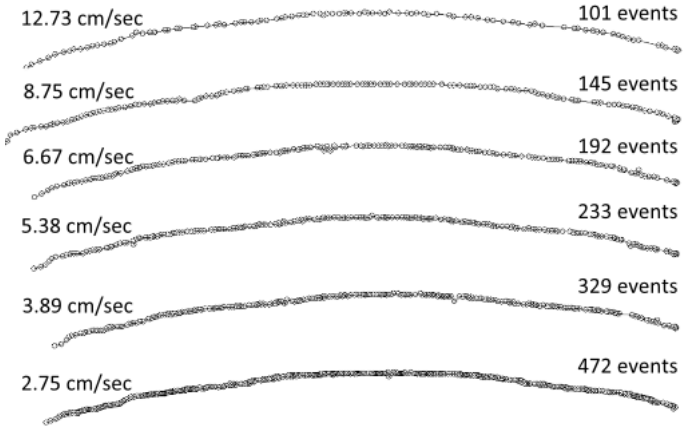
Our Approach

Number of Touch Samples (per one draw)

Drawing Test	Today Approach	Our Approach
Fast 1 (23.33 cm/sec)	52	59
Fast 2 (40.00 cm/sec)	33	61

Number of samples are increased for fast drawing

Slow Touch Drawing



Today Approach

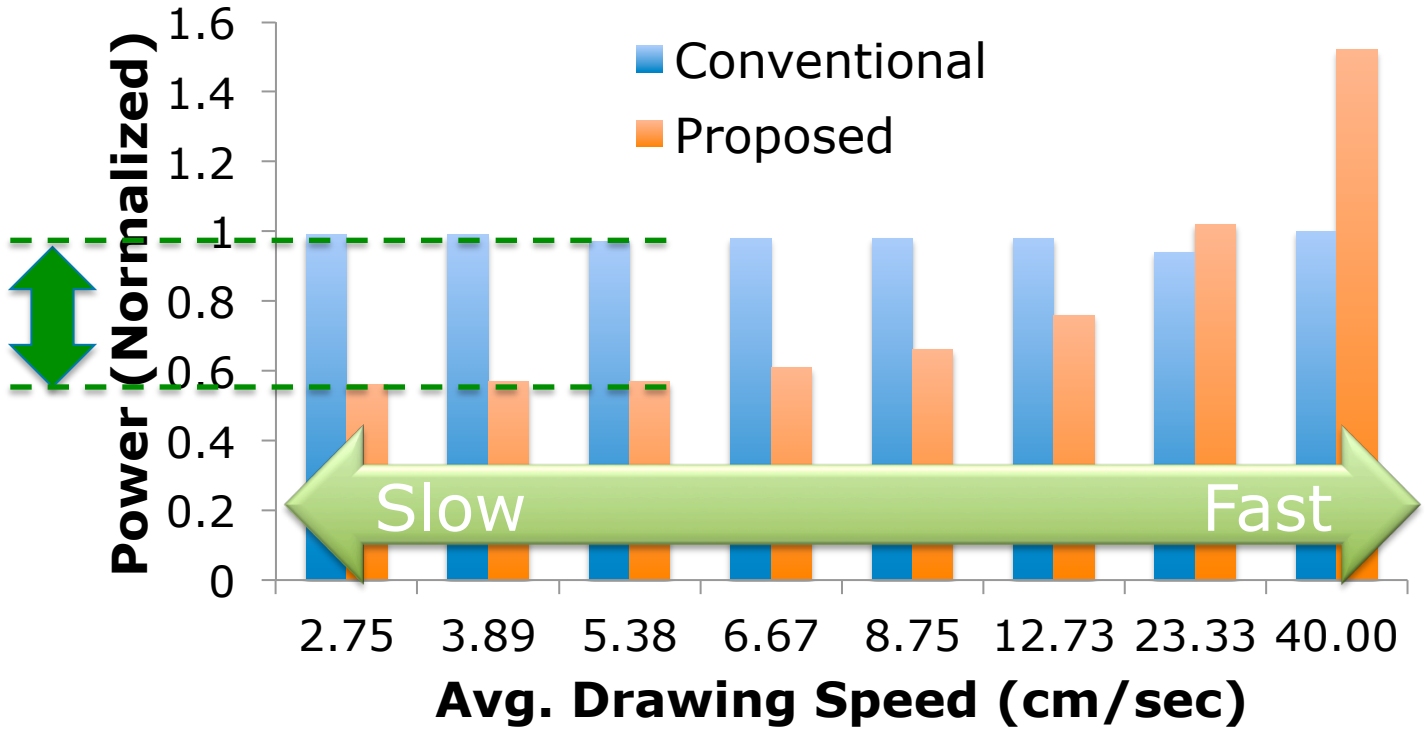
Our Approach

Slow Touch Drawing

Drawing Test (cm/sec)	Today Approach	Our Approach
Slow 1 (12.73)	101	64
Slow 2 (6.67)	192	74
Slow 3 (4.52)	281	91
Slow 4 (3.41)	379	113
Slow 5 (2.75)	472	136

Number of samples are decreased for slow drawing

Power Consumption for Different Drawing Speed



Up-to 44% power savings for Slow Drawing while responsiveness improvement for Fast Drawing

Conclusion

- Touch subsystem is energy hungry
- Our approach demonstrates adaptation of touch sampling rate to user touch behavior
- The energy consumption can be dramatically reduced by 44%
- Can be applicable to other human and sensor interfaces to improve energy efficiencies

THANK YOU!