Quantify and Optimize User Interactions with Android Devices

Xiao-Feng Li
xiaofeng.li@gmail.com

OCT, 2011
Summary

• Performance is not enough to characterize a client device

• An systematic approach has been established to engineering user interactions
  – AWS is a comprehensive engineering workload suite
  – UXtune is a toolkit to assist analysis and optimization

• Android user interactions can be impacted by the technical factors across the whole software stack

User interaction is a new challenge but not insurmountable
Agenda

• **Client device user interactions**
• Android user interaction optimizations
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• Factors that impact Android user interactions
• Summary
• Information
Performance Is NOT Enough

- Performance does not reflect consistently as user perception
- Performance is only about the behavior of system in *steady state*
- Then, what are missing?
Observation of Touch Fling Operation

• Fling the picture in Android Gallery application
  – Current picture slides and switches to next picture
Frames of A Fling Process on Device A

Frame Times of Touch Fling

Notice the followings:
• Max frame time
• #frames > 30ms
• Frame time variance
• FPS

* Data of one fling
Frames of A Fling on Device B (Higher FPS)

Notice the followings:
- Max frame time
- #frames > 30ms
- Frame time variance
- FPS

* Data of one fling
A Fling on Device B After Optimization

Frame Times of Touch Fling

Notice the followings:
• Max frame time
• #frames > 30ms
• Frame time variance
• FPS

* Data of one fling
User Interactions with Client Device

- A sequence of interactions

- Does the input trigger the target correctly?
- Does the system act responsively?
- Does the graphics transition smoothly?
- Does the object move coherently?
User Interactions in Software Engineering

- **Dynamic state transitions** in the software stack
  - Traditional performance more about *steady state*
  - Performance is the links of the interaction chain
- User Interaction evaluation is a superset of traditional performance evaluation
- For example: a video workload
  - "launch player" → "start playing" → "seek progress" → "video playback" → "back to home screen"
  - Traditionally, only "video playback" is evaluated
A Bit on User Experience Philosophy

• UX is not an objective process, but an interactive process with subjective factors
  – Consider watching movie or listening to music
• No silver-bullet to measure UX
  – Current academic research status with eye-tracking, heart-beat, poll, etc.
UX Philosophy Triangle

Ethos: “Do it correctly”

Logos: “Accomplish it systemically”

Pathos: “Follow it naturally”
Agenda

• Client device user interactions
• **Android user interaction optimizations**
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• Factors that impact Android user interactions
• Summary
• Information
Interaction Measurement Criteria

• Measure the critical path of user interactions in software stack

• Criteria
  – **Perceivable** (by a human being)
  – **Measureable** (by different teams)
  – **Repeatable** (in multiple measurements)
  – **Comparable** (between different measured systems)
  – **Reasonable** (about the causality)
  – **Verifiable** (for an optimization)
  – **Automatable** (largely unattended, not strictly)
Interaction Measurement Aspects

• User controls device (subject \(\rightarrow\) object)
  1. **Accuracy/fuzziness**: Range/resolution of inputs that can trigger a correct response
  2. **Coherence**: Object move delay, difference in move trajectory

• Device reacts to user (object \(\rightarrow\) subject)
  3. **Responsiveness**: Time between an input delivered to the device response, and to the action finish
  4. **Smoothness**: Maximal frame time, frame time variance, FPS and frame drop rate
Example Industry Experience Values

<table>
<thead>
<tr>
<th></th>
<th>Best</th>
<th>Good</th>
<th>Acceptable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response delay</td>
<td>100ms</td>
<td>200ms</td>
<td>500ms</td>
</tr>
<tr>
<td>Graphics animation</td>
<td>120fps</td>
<td>60fps</td>
<td>30fps</td>
</tr>
<tr>
<td>Video playback</td>
<td>60fps</td>
<td>30fps</td>
<td>20fps</td>
</tr>
</tbody>
</table>
Example "Drag": Drag Icon in Homescreen

- Measure drag coherence
  - P1: the position where the icon starts to move at T1
  - T2: the time when the icon reaches P1
  - P2: the position where the finger touches at time T2
  - P1 - P0, P2 - P1, T2 - T1 are the smaller the better
Example “Drag”: Map To Engineering Values

- **Metrics computation**
  - $T_1 =$ Time when Frame F1 is drawn by SurfaceFlinger
  - $P_1 =$ Position value of the touch event at time $T_1$
  - $T_2 =$ Time of the frame when icon’s position is $P_1$
  - $P_2 =$ Position value of the touch event at time $T_2$
Example “Drag”: Dev Engineering Workload

- Develop the application APK

```java
onTouch()

Detected LongPress → Detected Drag → DragView Move → Touch Release

Log the position

onDraw()

Icon Selected → Create DragView → Draw New Position → Icon Layout

Log the time
```

- Note: the touch event time-stamp is not the exact finger touch time. There is a few ms difference. It does not impact the drag lag distance optimization purpose.
Example “Drag”: Optimization Algorithm

- **Analysis**
  - T0: the event is delivered to system
  - T1: the app finishes drawing the first frame of movement
    - T1 - T0 is the Android processing time, cannot be 0
    - P1 - P0 is the distance finger moves during T1 - T0
- **Two complementary optimization approaches**
  - Optimize the execution path to reduce T1 - T0
  - Draw the icon at predicted position such as P2
Example “Drag”: Predicting Drag Position

• Move icon to the finger position when next frame is drawn
  – \( \text{SPEED}_{\text{finger}} = \frac{(P1 - P0)}{(T1 - T0)} \)
  – \( \text{TIME}_{\text{frame}} = \frac{1}{\text{FPS}} \)
  – \( \text{MOVE}_{\text{finger}} = \text{SPEED}_{\text{finger}} \times \text{TIME}_{\text{frame}} \)
  – \( \text{NextPOS}_{\text{finger}} = \text{MOVE}_{\text{finger}} + \text{CurrentPOS}_{\text{finger}} \)
  – \( \text{NextPOS}_{\text{icon}} = \text{NextPOS}_{\text{finger}} \)

• In reality
  – POSITION = (x, y)
  – Avoid icon surpassing finger
Optimize User Interaction Systematically

• What we need:
  – A well-established methodology
  – An engineering workload suite
  – An analysis/tuning toolkit
  – Sightings/requests/feedbacks from users

The key is to map user behavior into software metrics
Agenda

• Client device user interactions
• Android user interaction optimizations
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• Factors that impact Android user interactions
• Summary
• Information
Android Workload Suite (AWS)

• **Goals**
  – Reflect the representative usage of Android devices
  – Evaluate Performance, Power and User interactions

• **AWS usages**
  – Drive and validate Android optimizations
  – Support comparative and competitive analysis

• *(Details in another slide deck)*
UXtune: An Analysis/Tuning Toolkit

• To analyze and optimize Android, we need
  – Repeatable inputs operating the device
    • Android input-Gestures
  – Sequence of interaction events between the system components, such as event, frame, thread, etc.
    • Android UXtune
  – Metrics outputs characterizing the behavior
    • Android meter-FPS
    • Android app-launch
    • Android touch-pressure

• (Details in another slide deck)
Agenda

• Business perspective and executive summary
• Client device user interactions
• Android user interaction optimizations
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• Factors that impact Android user interactions
• Summary
• Information
Browser Scroll Response Time

Input Manager

Browser

Event1 → Event2 → ...... → Eventk → ......

Frame 1 → ......

Gesture detecting

Frame drawing

Response Time
Scroll Response Time Optimizations

• Gesture detecting time
  – ACTION_DOWN event: record the start position
  – ACTION_MOVE event: compute the move Distance
  – Scroll is detected when Distance > Threshold

• Frame drawing time
  – Skipped in this talk
User Input Event Dispatching

- **InputHandler**
  - **InputChannel**
  - **InputConsumer**
  - **MessageQueue**
  - **InputReader**
  - **InputDispatcher**
  - **InputPublisher**
  - **ViewRoot**
  - **WebView**
  - **AshMem**

- **EventHub**

- Events flow:
  - Raw event
  - Event
  - Packaged move event
  - Notification

- Dates: 2011-09-12
Move Events Throttling

- **InputDispatcher** throttles the move-event emission rate
  - No need to emit move-event faster than platform maximal FPS
  - Set a minimum time interval between move-events dispatching, commonly \(1/\text{FPS}\)
  - The move-events between two dispatches are grouped together
Experiment of Event Emission Throttling

- Without throttling, gesture detection time can be reduced by up to $1/fps$ in a common device

<table>
<thead>
<tr>
<th>Time of Event</th>
<th>Throttling Delay</th>
<th>Time to Browser with throttling</th>
<th>Time to Browser Without throttling</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>12.096</td>
<td>8</td>
<td>17</td>
</tr>
<tr>
<td>15</td>
<td>3.096</td>
<td>21</td>
<td>46</td>
</tr>
<tr>
<td>25</td>
<td>12.14</td>
<td>41</td>
<td>27</td>
</tr>
<tr>
<td>45</td>
<td>11.894</td>
<td>59</td>
<td>46</td>
</tr>
<tr>
<td>71</td>
<td>4.191</td>
<td>79</td>
<td>72</td>
</tr>
<tr>
<td>78</td>
<td>15.355</td>
<td>79</td>
<td>79</td>
</tr>
<tr>
<td>87</td>
<td>6.355</td>
<td>98</td>
<td>88</td>
</tr>
<tr>
<td>96</td>
<td>15.897</td>
<td>97</td>
<td>97</td>
</tr>
<tr>
<td>106</td>
<td>5.897</td>
<td>116</td>
<td></td>
</tr>
</tbody>
</table>
Agenda

• Client device user interactions
• Android user interaction optimizations
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• **Factors that impact Android user interactions**
• Summary
• Information
## Factors That Impact User Interactions

<table>
<thead>
<tr>
<th>SOC factors</th>
<th>OS/Runtime</th>
<th>Input factors</th>
<th>Animation design</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Memory bus bandwidth, HW gfx acceleration</td>
<td>- Multi-core software synchronization, load balance</td>
<td>- Event emission rate</td>
<td>- Property animation design</td>
</tr>
<tr>
<td>- Touch screen pressure resolution</td>
<td>- Thread scheduling priority, UI thread vs. other computations</td>
<td>- Sensor event detection</td>
<td>- Application transition</td>
</tr>
<tr>
<td></td>
<td>- Runtime engine design</td>
<td>- Touch gesture detection</td>
<td>- Gesture/Sensor inputs response</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Operation smoothness</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Drag coherence</td>
</tr>
</tbody>
</table>

*2011-09-12*
Agenda

• Client device user interactions
• Android user interaction optimizations
  – Android Workload Suite and Android UXtune
• Case studies of Android optimizations
• Factors that impact Android user interactions
• Summary
• Information
Summary

• Performance is not enough to characterize a client device

• An systematic approach has been established to engineering user interactions
  – AWS is a comprehensive engineering workload suite
  – UXtune is a toolkit to assist analysis and optimization

• Android user interactions can be impacted by the technical factors across the whole software stack

User Interaction is a new challenge but not insurmountable
Information

• Online resource for reference
  – http://ux.stackexchange.com/
  – http://www.measuringux.com/