Odessa: Enabling Interactive Perception Applications on Mobile Devices

Moo-Ryong Ra*, Anmol Sheth+, Lily Mummertx, Padmanabhan Pillai’, David Wetherall°, Ramesh Govindan*

*USC ENL, +Technicolor, xGoogle, ‘Intel, °University of Washington
Emerging Mobile Perception Applications

- **Sensing**
  - GPS
  - Accelerometer

- **Computation**
  - Dual-Core CPU

- **Communication**
  - Cloud Infrastructure

- **Sensing Applications**
  - Activity Recognition
  - Health, Traffic Monitoring
  - Location-Based Service
  - Participatory Sensing
Vision-based Interactive Mobile Perception Applications

- Face Recognition
- Object and Pose Recognition
- Gesture Recognition
Common Characteristics

Interactive

- Crisp response time (10 ms ~ 200 ms)

High Data-Rate

- Processing video data of 30 fps

Compute Intensive

- Computer Vision based algorithms
### Performance

<table>
<thead>
<tr>
<th>Application</th>
<th>Throughput</th>
<th>Makespan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face Recognition</td>
<td>2.50 fps</td>
<td>2.09 s</td>
</tr>
<tr>
<td>Object and Pose Recognition</td>
<td>0.09 fps</td>
<td>15.8 s</td>
</tr>
<tr>
<td>Gesture Recognition</td>
<td>0.42 fps</td>
<td>2.54 s</td>
</tr>
</tbody>
</table>

*All running locally on mobile device*

*Video of 1 fps*
Two Speed-up Techniques

Pipeline Parallelism

Network

Application Data Flow Graph

Frame 1

Frame 2

Frame 3

Screen
Main Focus

Data Flow Structure

System Support

Offloading + Parallelism

Enable Mobile Interactive Perception Application
Contributions

What factors impact offloading and parallelism?
Measurement

How do we improve throughput and makespan simultaneously?
Odessa Design

How much benefits can we get?
Evaluation
Measurement

- Input Data Variability
- Varying Capabilities of Mobile Platform
- Network Performance
- Effects of Parallelism
Lesson I: Input Variability

The system should adapt to the variability at runtime

Impact of input variability

Object and Pose Recognition

Motivation
Problem
Measurement
Design
Evaluation
Lesson II: Effects of Data Parallelism

Object and Pose Recognition

<table>
<thead>
<tr>
<th># of Threads</th>
<th>Thread 1</th>
<th>Thread 2</th>
<th>Thread 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,203 ms</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>741 ms</td>
<td>465 ms</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>443 ms</td>
<td>505 ms</td>
<td>233 ms</td>
</tr>
</tbody>
</table>

The level of data parallelism affects accuracy and performance.

Input Complexity

Segmentation Method
Summary: Major Lessons

Offloading decisions must be made in an adaptive way.

The level of data parallelism cannot be determined a priori.

A static choice of pipeline parallelism can cause sub-optimal performance.
Odessa

Offloading DECision System for Streaming Applications

- Application
  - Odessa Profiler
  - Sprout
  - Cloud Infrastructure

Network

Runtime

Application

Profiler | Decision Engine
Odessa | Sprout
Mobile Device

Input Video Stream
Incremental Decision Making Process

Incremental decisions adapt quickly to input and platform variability.
Evaluation Methodology

Implementation

Experiments

- Odessa Adaptation
- Resulting Partitions
- Performance Comparison

Linux / C++

- 1-core Netbook
- 2-core Laptop
- 8-core Server

Canned Input Data

Motivation

Problem

Approach

Design

Evaluation
Data-Flow Graph

Face Recognition

Object Pose Estimation

Gesture Recognition
Odessa finds a desirable configuration automatically.
### Resulting Partitions in Different Devices

<table>
<thead>
<tr>
<th>Client Device</th>
<th>Stage Offloaded and Instances</th>
<th>Degree of Pipeline Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Device</td>
<td>Face detection (2)</td>
<td>3.39</td>
</tr>
<tr>
<td>Dual Core Notebook</td>
<td>Face Detection (1) Motion-SIFT Feature (4)</td>
<td>3.06</td>
</tr>
<tr>
<td></td>
<td>Face Detection (1) Motion-SIFT Feature (9)</td>
<td>5.14</td>
</tr>
</tbody>
</table>

Resulting partitions are often very different for different client devices.
## Performance Comparison with Other Strategy

### Object and Pose Recognition Application

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Throughput (FPS)</th>
<th>Makespan (Latency)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offline-Optimal</td>
<td>6.49</td>
<td>430 ms</td>
</tr>
<tr>
<td>Odessa</td>
<td>6.27</td>
<td>807 ms</td>
</tr>
</tbody>
</table>

Odessa performs 4x better than the partition suggested by domain expert, close to the offline optimal strategy.
Related Work

- **ILP solver** for saving energy: [MAUI] [CloneCloud]
- **Graph-based** partitioning: [Gu’04] [Li’02] [Pillai’09] [Coign]
- **Static Partitioning**: [Wishbone] [Coign]
- A set of **pre-specified** partitions: [CloneCloud] [Chroma] [Spectra]
Summary of Odessa

Adaptive & Incremental runtime for mobile perception applications

- Odessa system design using novel workloads.
- Understanding of the factors which contribute to the offloading and parallelism decisions.
- Extensive evaluation on prototype implementation.
Thank you

“Any questions?”