

Mercury: Monitoring Patients with Parkinson's Disease using Wearable Wireless Sensors

Konrad Lorincz¹, Bor-rong Chen¹, Geoff Werner-Allen¹, Atanu Roy Chowdhury¹, Benjamin Kuris², Steve M. Ayer², Shyamal Patel³, Paolo Bonato^{3,4}, and Matt Welsh¹

¹ School of Engineering and Applied Sciences, Harvard University

² Intel Digital Health, Advanced Technology Group, Cambridge MA

³ Department of Physical Medicine and Rehabilitation, Harvard Medical School

⁴ Harvard-MIT Division of Health Sciences & Technology



Introduction

Mercury is a wireless sensor network platform designed for long-term high-resolution motion capture of patients in home settings. Patient wears up to 8 wireless sensors each with triaxial accelerometers and gyroscopes, low-power radios, and large (multi-gigabyte) flash memories. Sensor data is stored to local flash and high-level features are computed from the raw signals. Data is opportunistically downloaded to a base station in the patient's home based on bandwidth and energy availability, for delivery to the clinic in near real time.

Challenges

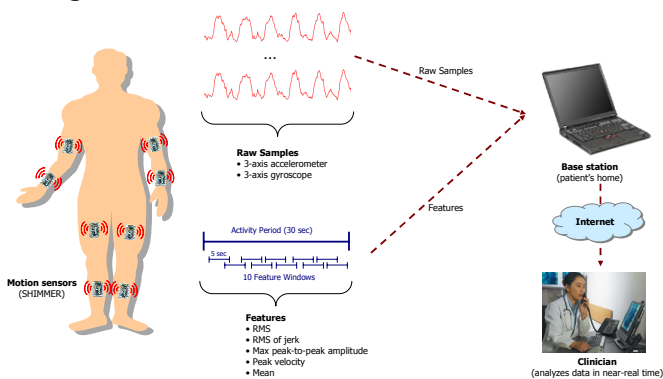
- Energy limitations
- Unpredictable load
- Computational limitations
- Robustness for long-term operation

Application: Parkinson's Disease

Background

- **Parkinson's:** degenerative disease that affects control of muscles
- **Primary problem:** deficiency of dopamine do to degeneration of neurons
- **Primary therapy:** augmentation or replacement of dopamine via the precursor *levodopa* which activate dopamine receptors

Usage Scenario



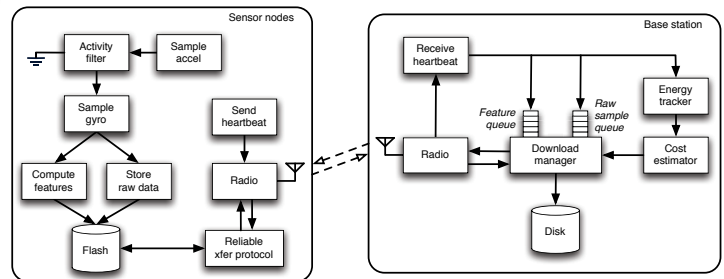
Hardware Platform

SHIMMER

- Low power:
 - TI MSP430 microprocessor
 - Chipcon CC2420 802.15.4 radio
- Large flash storage
 - MicroSD card slot (up to 2 GBytes of flash)!
- Small size & weight
 - Size: 1.75" x 0.8" x 0.5"
 - Weight: 10g
- Sensors
 - 3-Axis accelerometer (1.5 – 6G)
 - 3-Axis gyroscope (500 °/s)



Mercury Architecture



Sensor Nodes

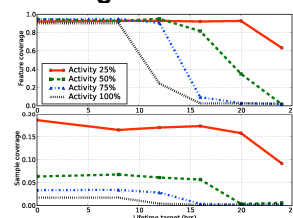
- Sampling and activity filter
 - Inactive periods: activity filter drops samples to save power
 - Active periods: computes features and logs raw samples and features to flash
- Time synchronization
 - Using the FTSP protocol with microsecond accuracy
- Feature extraction: in real-time during activity periods
- Status messages (every 10 sec)
 - Number of sample and feature blocks
 - Available energy and radio link quality (using Pixie)

Base Station

- Schedules downloads (using Lance)
- Policies optimize for
 - data coverage and fidelity while meeting lifetime target
- Prioritizes features over samples

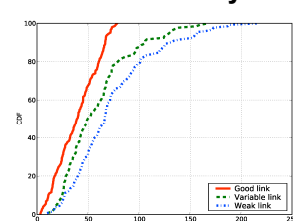
Evaluation

Coverage and Lifetime



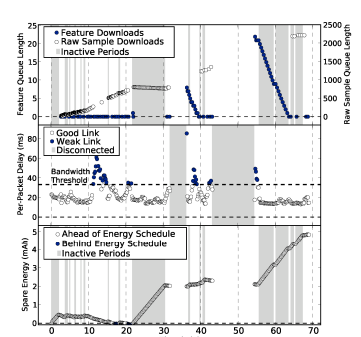
- Amount of data downloaded decreases with higher lifetime targets and activities
- Features prioritized over samples

Download Latency



- Latency increases as link quality degrades

Node Behavior



- Only downloads samples when feature queue length is zero
- Spare energy increases during periods of inactivity and disconnections