①

MOBILE AND SENSOR SYSTEMS

Large numbers of people only have a mobile device, especially in the developing world

especially with growth of cellular networks.

↳ half of average adults internet usage is spent on mobile devices

↳ Between 2013 and 2015, mobile app usage ↑ by 90%. 50% of uses grab minicalels after waking up

↳ Device preference changes overtime with PCs increasing during working day

↳ Utility of sensor systems increased because of wearables

↳ computation units are specifically built to maximise performance

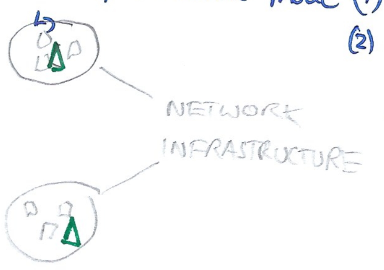
Challenges in Mobile Computing:
- ↳ ① Resource Constrained — battery
- ↳ ② Connectivity variable in performance and reliability
  - ↳ coverage issue + tradeoff with energy consumption
- ↳ ③ Less secure.
  - ↳ devices can easily be stolen
- ↳ ④ Distributed Systems
  - ↳ ① Remote comms
  - ↳ ② Fault-Tolerance
  - ↳ ③ Remote information access
  - ↳ ④ Distributed security
- ↳ ⑤ Database issues
- ↳ ⑥ Energy issues
- ↳ ⑦ HCI issues — limited interface, ergonomics
- ↳ ⑧ Privacy issues

issues in designing mobile systems

↳ Communication structures:

↳ infrastructure mode: (1) base station connects mobiles into wired network

(2) handoff: mobile changes basestation



Access to wireless shared among transmitters using multiplexing

↳ (1) Time, (2) Space, (3) Freq, (4) Code

↳ But, bit rigid as a method - can lead to a bottleneck

↳ CSMA/CA etc

↳ Ad-hoc: No base stations, nodes transmit to other nodes within link coverage. Nodes organise themselves into ~~nodes~~ a network

CARRIER SENSING MULTIPLE ACCESS WITH COLLISION AVOIDANCE — CSMA/CA

If free, transmit. If not, wait with randomized back off strategy. Transmit when medium is sensed free.

↳ Hidden Terminals: A → B. C cannot receive from A, C wants to send to B, senses a free medium. Collision at B, hence A is hidden for C.

A — B — C

↳ Exposed Terminals: B → A. C wants to send to D. C has to wait for no reason.

↳ C exposed to B.

Multiple Access with Collision Avoidance (for Wireless) : MACA(W)

① A asks B if B able to receive with a RTS (request to send)

② B agrees, sends Clear to Send (CTS)

③ A sends, B ACKs ↳ Potential interferers overhear RTS and CTS and how long it will last. Store information in Network Allocation Vector.

Ad-Hoc Network Routing

All these only work when connecting path exists *

Destination Sequenced Distance Vector Routing → DSDV - using proactive routing, routes are maintained when not needed. Each node maintains a table with a route to every node. Entry of table has sequence number

↳ Dest., Next Hop., Hops req'd, Seq number

↳① Each node periodically transmits updates with its own sequence number
  ↳ can also routing table updates for incremental link changes

↳② Update routing table

↳③ When two identical routes received with different seq. numbers, left with greatest destination sequence numbers

With new link K: → ① Transmit <K, K, 0, 101>
  ② A receives this and inserts <K, ~~K~~ ~~A~~ K, 1, 101> in routing table
    ↳ propagates to neighbours, which adds <K, A, 2, 101> and continues propogation

Limitations: → ① Circulating + Maintaining table is expensive
  ↳ ② Updates worthwhile only if lots of changes

Dynamic Source Routing: routes searched for only when communication is required
↳ best for low mobility and stable origin - destinations

↳ when node needs to communicate, sends route request packet
↳ Nodes receive + add themselves to path and propogate to neighbours
↳ When destination found, path sent back to source
↳ Sequence numbers used to avoid routing loops.
  ↳ routes cached for some time.

Zone Routing Protocol: combines reactive and proactive. Zone around node N is collected proactively. Inner zone done using reactive method.

Delay Tolerant Networks and Protocols: does not assume temporally connected path among nodes.

↳ Epidemic Routing: flooding protocol - nodes store packets before forwarding ↳ once it has moved to another neighborhood
in particular the forwarding number ↳ If tuned, reaches optimal delivery
↳ Needs large memory (else buffers fill + packets ejected and lost)

Context Aware Routing

Can exploit knowledge of node mobility to do better, using predictability as a predictor
↳ estimate chance for all neighbours of eventually reaching the destination
↳ Hence, decide whether to store or forward the packet.
  ↳ (1) Host mobility, (2) Host colocations with dest node, (3) Kalman filter
  (4) Utility functions based on these

Wireless Sensor Networks

Sensor Nodes ⇔ Patch Network ↔ Transit Network ↔ IP ⇒ Data Service

↳ limited computational resources \
↳ Prone to failure / OS needs to support concurrency \
↳ Consists of : (1) Sensing device(s) (2) Low power radio, (3) Small storage \
↳ Topology changes frequently

↳ need to be careful about power consumption by limiting number of transmissions. But idle listening takes same power as transmitting: Use Radio Duty Cycling

Dynamic Duty Cycling

Synchronised (SMAC) - negotiate schedule among neighboring nods

Asynchronous (BMAC / XMAC) - preamble sampling to connect transmitter to receivers

SMAC: exchange schedule of wakeup schedule and active period. When awake, perform RTS / CTS. Split into periods of (1) SYNCH, (2) RTS and (3) CTS

(1) SYNCH ↳ divided into time slots with CSMA and backoffs to send schedule to neighbours

↳ Y chooses slot & if no ~~slot~~ signal received in this slot, transmit schedule to X. Else, wait for next wake up of X

(2) RTS ↳ X listens for RTS packet

(3) CTS ↳ X sends one and extends wake up time.

S-MAC Synchronized Islands

① Nodes try to get schedule synchronization from neighbouring nodes \
② Else, pick some schedule to start with \
③ If new joins, can lead to synchronized island \
④ To bridge gap, follow both schemes.

Preamble Sampling (Low Power Listening)

Have receiver sleep and only periodically sample the channel - long preamble to ensure receiver stays awake to catch actual packet

↳ Overhearing - all receivers have to stay awake to find intended recipient \
↳ Energy Consumption - preamble uses energy \
↳ Latency - per hop latency introduces by long preamble

X - MAC

↳ short preamble, target in the preamble, adding wait time between preambles

short preamble: reduce latency and reduce energy consumption \
target in the preamble: minimize overhearing \
adding wait time between preambles: → reduces latency where dest is awake before preamble

Low - Energy - Adaptive - Clustering Hierarchy (LEACH)

Assumption: Dense network of nodes, reporting directly to a central sink.

Idea: Group nodes into clusters with rotating clusterhead which advertise themselves \
↳ Nodes join clusterhead with highest signal

↳ CDMA for all member transmission. TDMA schedule used within cluster

↳ CHs collect & aggregate data from all cluster members - report aggregated data to sink using CSMA

(Slide 26, Lecture 3)

④

IoT : [Devices, Machines, Environments], physical objects

Infrastructure for information society - interconnecting 'things' evolving interoperable information and communication technologies (can be connected to the internet)

IoT Device ⟷ IoT Gateway ⟷ IoT Cloud
edge analytics
local storage

Communication Technologies: (1) Near field, (2) Personal area network (PAN), (3) Local Area Network (LAN), (4) Wide Area Network (WAN)
(Slide 31, Lecture 3)
↳ can also classify by range and bandwidth
↳ (Slide 32, Lecture 3)

LoRa WAN
↳ non-cellular LPWAN, in license-free spectrum
↳ started in 433MHz } → method of radio modulation that means very long range

LPWAN: Low Power WAN - low bandwidth, high range, cheap, deep indoor penetration
↳ But latency can be higher and loss of packet more likely

Three types: ① Battery Powered - Class A : (1) Bidirectional
↳ High latency but low power usage
(2) End-device initiates comms
(3) Server downlinks during predetermined response window
2 downlink slots per uplink

② Low Latency - Class B : (1) Bidirectional
↳ High power
(2) Periodic synchronisation beacon
(3) Lots of periodic downlinks

③ No latency (Mains Powered Devices) - Class C : (1) Bidirectional
↳ Very high power usage
(2) Server downlink at any time with ~~save~~ end-device constantly receiving

Sensor Network Diffusion

Can use adhoc routing, but (1) overhead, (2) different aim - data to single or multiple source.

Aim: subscribe once, events happen multiple times. ← [↳ sink interested in all results
↳ sink interested in change in results
→ But unknown which node can provide data + multiple nodes to ask for data.

Directed Diffusion: Relies on local interactions, with nodes sending subscriptions diffused through the network - each node stores subscriptions of neighbors and disseminate
→ there time out
→ gradient = rate of which are events is req'd

Sensors produce data routed according to subscriptions (diffused). Intermediate nodes filter/aggregate data

GRAD = rate that neighbours receive data

Data can reach a node through different routes: Gradient Reinforcement ← [intermediate node sends back ~~rate~~ data at rate based on gradient → GRAD
↳ either send occasionally or aggregate all results

↳ when gradient established, rate defined as low - sinks reinforce good paths and expires (after timeout) if not reinforced.

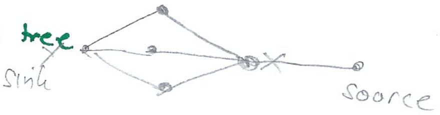
⑤

Two-Phase Pull

↳ Phase 1 : Nodes distribute interests (subscriptions) as attribute-value pairs which are flooded in the network. But need to be able to identify where the interests came from :- form a tree but need to identify in this case :

↳ ① Forward to all parents in tree

↳ ② Only forward to one parent



↳ ③ Provisionally send to all parents but get sinks to ~~send~~ set gradients.

↳ Diffusion means interests are unnecessarily sent everywhere.

∴ PUSH DIFFUSION : don't flood interests, flood just data and gradients reinforce the interests

↳ (1) Purely theoretical

↳ (2) World issues : link load / dependency / stability

↳ (3)

Data Aggregation

↳ Metrics : (1) Accuracy, (2) Completeness, (3) Latency, (4) Message Overhead

MT Routing : idea is that number of hops may not necessarily be the best performance indicator for wireless sensor network ⇒ should take into account network factors. Also, link connectivity not spherical, as assumed.

↳ Good estimators should be : (1) Stable, (2) Simple to compute (low memory footprint), (3) React quickly to quality changes.

↳ NEIGHBOUR BROADCAST is a good passive estimate

WMEWMA : track seq number of packets for each source to infer losses

$$\text{window mean} - \text{EWMA}(\boxed{t_x}) = a\,(\underbrace{\overset{\text{num of packets received}}{\boxed{\text{MA}(t_x)}}}) + (1-a)\,\text{EWMA}(t_{x-1})$$

↳ last time interval ; a = weight

Neighbourhood Management : has Neighbourhood table - records info about nodes from which we receive packets

↳ good number of neighbours

Link Estimation Routing : estimate inbound links, each node selects a parent using link estimation table

↳ changes when the link deteriorates

Distance Vector Routing : as per standard distance vector - metric is usually the hop count, but this can underestimate costs because of retransmission - just consider retransmission costs.

Sensor Data Inference

↳ inference characteristics : (1) offline / online inference
(2) Continuous / Isolated / Periodic Inference ↳ Need ground truth

Applications : (1) Individual sensing, (2) Group / Community sensing, (3) Urban-scale sensing

↳ fitness + health

↳ to sense common activities and help achieving group goals

↳ same application installed

Challenges : (1) Within-class variation : people's gait seriously affect readings on sensors
(2) Range of activities which are similar - ground truth annotation is hard

Activity Recognition : recognize actions of individual from series of observations on individuals actions and environmental conditions

↳ sensors produce sequential data

↳ (1) Sensors work at different sampling rate, (2) Data can be corrupted and contain error

↳ Pipeline : (1) Raw data → (2) Preprocessing → (3) Segmentation → (4) Feature → (5) Classification

Preprocessing: (1) synchronizes and (2) removes artifacts
↳ (1) Calibration, (2) Unit conversion, (3) Normalization, (4) Resampling
(5) Synchronization

Need good ground truth data
↳ also for similar states to ones we are trying to detect

- **(Data) Segmentation**: Localize temporal patterns of interest ⇒ sliding window works well.
- **Feature Extraction**: (1) Physical Activity with accelerometer – features are: (1) Mean, (2) Standard Deviation, (3) Number of peaks
  ② Conversation Detection FFT of audio – mean ⁽¹⁾ and standard deviation ⁽²⁾ of FFT power, (simple threshold line)

Classification

Maps feature vector to one of pre-defined sets of high-level classes, e.g. (1) K Nearest, (2) Naive Bayes, (3) Decision trees, (4) HMM

Deep Learning: Sensing → Feature Extraction → Classification ⇒
⇓
Sensing → Deep Neural Network ⇒ less tied to specific domain and tasks

Move away from features towards combine feature | hand-crafted models that sensing + classification

CNNs, RNNs

↳ Learn discriminative features from raw data (end-to-end learning)
↳ Mel Cepstral Frequency Coefficients seem to be similar to representation directly from data

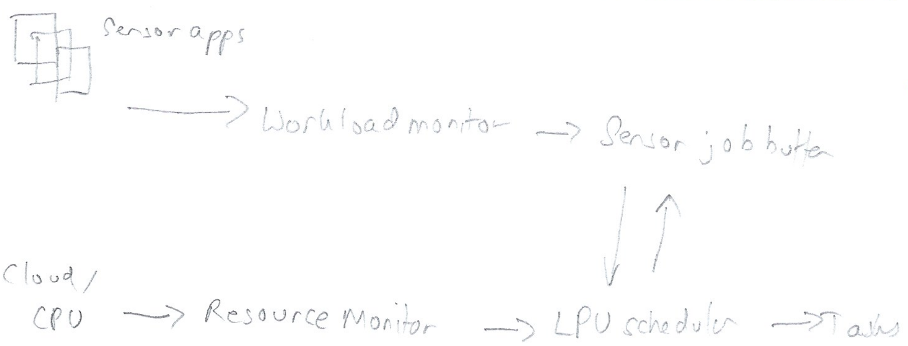
BUT, sensor data lacks large scale labelled ground-truth datasets - lead to overfitting
↳ Transfer Learning ①, Classifier Ensembles ②

Energy and Systems Considerations

MAUI: mobile device framework which profiles code components and decides whether to run locally or remotely considering latency requirements
↳ CPU only
↳ Costs related to transfer of data
↳ Dynamic decisions based on network constraints

but optimized GPU is better

LEO



low overhead

truncations or stochastic rounding

* apply after pruning
Can be applied in a number of ways (1) K-means, (2) Hashing (3) Huffman Coding (4) weight sharing
Quantize at every step in training but leave back prop parameters

Tradeoffs: Accuracy per £ in Mem / latency higher precision

Machine Learning Inference on a device: Why? – data privacy
– limit bandwidth out
↳ Applications: Video applications on image sensor for traffic characterisation
↳ Drone / robot navigation local processing

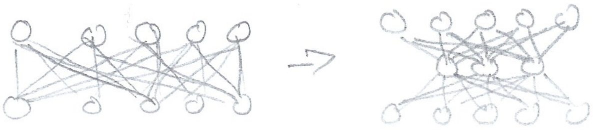
Need to consider:
↳ Memory
↳ Energy
↳ Latency

To improve resource tradeoffs → ① Prune – remove excess parameters → set random parameters to 0, use magnitude as criterion
→ Prune, then retrain, etc
② Quantize – decrease parameter precision *
③ Fully connected layers → weight factorization - low rank weight approximation
④ Convolutional Layers → convolutional separation - low rank approx

Binary Weight Networks (BWNs)
↳ weights set to $\{-\alpha, \alpha\}$ set based on original layer values

SVD Weight Approximation

Add a new layer to minimise connections



Memory + Compute saving $k < \frac{m \times n}{m+n}$

Convolution Separation: Aim: Find kernel approximation that is:
↳ (1) More computationaly efficient
↳ (2) Faithful to original kernel

$$K_n \in \mathbb{R}^{d \times d \times C}$$

$$K_n \simeq \hat{K}_n$$

$$\hat{K}_n^c = \sum_{i=1}^{K} \underset{\cdots}{H_n^n} (\overline{V_K^c})^T$$

→ vertical filter

horizontal filter

Get computation gain when $K < \frac{dCN}{C+N}$

Other efficiency parts: ① Commodity processors and accelerators
② System-level Solutions
③ Cross Models Optimizations
④ Low-Resource Architectures - MobileNet

Mobile Privacy : data generated by mobile phone and apps
↳ permissions handled differently on Android and iOS
~~[illegible]~~ user's need to be careful about app permissions ←
↳ Apple does vetting

↳ **Privacy Data Breach Detection** - Data Flow Analysis (DFA) - looks for routes between data sources and sinks. Any route without consent is a leak
- sources: Location Data, Unique identifiers, Auth Data, Contacts Calendars, Card states
- sinks: SMS, calls, file output, network

↳ Capability Leak - malicious app manages to hijack permissions granted to other trusted apps.

Dynamic Analysis

TaintDroid: Modification of Android OS allows for dynamic tracking of sensitive data movements from app to other apps and sinks
↳ automatically labels data from privacy sensitive sources ~~ap~~ and applies labels as propogates through program variables, files and interprocess messages. Logs labels as data goes to sink

Static Analysis

Cover all paths from sources to sinks - LeakMiner

Cellular Network Leaks : Mobile devices ~~and~~ roam and register with BTS's - can be identified from these records and pre-existing location profiles - one paper identifies 80% of uses. GSM itself allows listening attacks / leaks

WiFi Based Leaks: WLAN fingerprints used to infer social relations between user. Devices broadcast WiFi information that contains MAC address, hence possible to track. Can analyse traffic with WiFi hotspots. → unique ID
↳ Can also get last scanned list of WiFi hotspots - use to geolocate users and configured WiFi networks

⑧

Mobile Sensing Leaks

- Accelerometer can be used as a device signature - or identify different users of same device (and touch sensor usage)
- Even in large anonymised datasets, there's a risk that user can be identified using correlation

Location Data Leaks: Location aware apps collect data sometimes provide to third parties
- can be used to get personal information - profiles
- Data can be cross-correlated among different sources
- Location history can be used to identify the future movements but spatio-temporal prediction is hard - can know where but maybe not when
  - Predict using: ① Previous history, time spent at locations etc
    ② Correlated movement - analyse mobility patterns of users (mutual information) → leads to network of movements

POSITION AND LOCATION

→ main signal for context with a number of actual applications with it as well. (1) Safety, (2) Energy usage, (3) Space usage, (4) Security, (5) Navigation, (6) Collaboration, (7) Resource Routing, (8) Retail, (9) Health

Proximity Location:① mark area with measurable signal with limited range. When mobile device measures that signal, we know where it is → eg cellular localisation uses cellular mobile signals from each transmitter tower
- in 1990s, Infrared Active Badges
- iBeacons using BLE (Bluetooth Low Energy)
- Phillips flashing lighting - encodes ID flashing too fast to be perceived ⇒ detect using careful rolling shutter

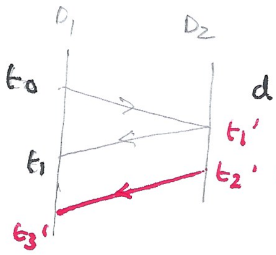
(2) **Time of Flight**: measure time for signal to propogate between two devices given known speed of signal. Can find location using trilateration or multilateration

→ But: (1) Need to sync the devices accurately, → 1ps delay = 300m error for radio (2) Direct signal, cannot be multipath needs to be line of sight

overestimate the distance if we have multipath. → multipath

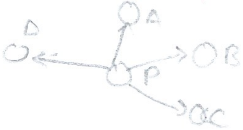
- can sync by speed differential if we transmit with two signals of very different speeds. Consider faster signal as instantaneous, using error of ratio of speeds × d.
- Bat system works in 50Hz and 433MHz - achieves 3m accuracy 95% of the time

RTT Sync:



$d(A,B) = c(t_1 - t_0)/2$

$d(A,B) = c((t_3' - t_0) - (t_2' - t_1'))/2$, time to respond



Time Difference of Arrival: Sync only base stations:

each equation gives a hyperbola → $c(t_A - t_p) = d_{AP}$

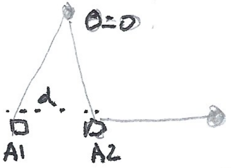
$c(t_C - t_p) = d_{CP}$

→ intersection gives the position

$c(t_A - t_C) = d_{AP} - d_{CP}$ → this does not involve P's unsynced clock

①

Angle of Arrival : estimation position of transmitter if you can get bearings to the signal frommultiple sites.



↳ Path difference means delayed version of signal is recorded :- phase difference changes with θ

$\theta = \frac{\pi}{2}$, path diff of d,
phasediff of $2\pi/\lambda \times (d \% \lambda)$

↳ best to set d as half the wavelength - if longer, multiple θ for certain phase differences. Can make d shorter, but need to more accurately measure the phase difference

↳ This is included as part of Bluetooth 5.1

UWB and Ubisense : Radio signals, while passing through walls, also suffer from multipath reflections. Leads to reducing pulses, each with finite width. Therefore hard to get the first pulse. Higher bandwidth - sharper pulses. Therefore use ultrawideband therefore easier to identify first pulse. Need to deal with noise by setting trans power above noise floor to cover the underlay (noise). Ubisense made of UWB receives are placed in the environment and wired ~~otamg~~ so that they are time synced.

↳ TDoA based on UWB ranging

↳ AoA based on receiver antenna array

Hotstart: < 4hrs since fix, need no new info

>4 hours since fix

Warm Start: Receives cache rough location, time, almanac, just need ephemeris. *

Cold start: search for all satellites + download whole almanac - 15 mins

GLOBAL NAVIGATION SATELLITE SYSTEMS (GNSS)

(1) NAVSTAR GPS, (2) GALILEO (EU), (3) Beidou (China), (4) Glonass (Russia)

↳ mostly considering GPS here

9 reserves, 2 being tested, 30 retired

Made of : (1) Space segment, (2) Control segment, (3) User segment :

31 satellites 12,500 miles away, need 24 satellites at one time

↳ 30 W signal into antenna. After antenna gain and travel to earth, get 10^{-16} W ⇒ use 15MHz and just 50 bps

Capacity from Signal to noise ratio:

↳ Shannon - Hartley : $C = B\log_2(1+S/N)$

↳ set of ground stations :

Tasks include :

① Monitor health of satellites
② Compute ephemeris corrections
③ Upload corrections
④ Upload clock corrections

↳ gps receiver

① Download almanac : list of satellites and approx orbital information and their codes *

② Download ephemeris information for observable satellites

③ Compute pseudorange and if 4+ satellites, GPS fix

↳ apparent range between satellite and receiver.

each satellite has its own signal; satellite s_1 transmits at t_{s_1}. Mobile records when it receives signal t_m.

$d_{pseudo,s_1} = c(t_m - t_{s_1})$

$d_{real,s_1} = d_{pseudo,s_1} + ct_{offset,s_1}$ → try to find this

Assume $t_{offset,s_1} = t_{offset,s_2} \ldots$

∴ need to solve for x, y, z and t_{offset}, need to see four satellites

↳ Need to know accurate position of each satellite by downloading almanac ~~contains~~ and the ephemeris.

for rough location

↳ valid for 180 days

↳ valid for ≈ 5 hours

* Cold start also has untrusted time

A-GPS: get almanac + ephemeris over network data chanel - allows for hot-start.

Information Theory tells us the Shannon-Hartley theorem: $C = B\log_2(1 + S/N)$. Increase B by applying signal spreading by XORing with a spreading code and XOR back to get original signal. Can be used ~~to~~ deal with a little bit of noise.

Gold codes: use spreading codes with strong autocorrelation but very weak cross-correlation with one another.

Output is Binary Phase Shift Keyed with carrier-wave encoding gold code ⊕ navigational data
↳ L1 is at 1575.42MHz normally
↳ carrier phase flipped on any bit transition
↳ 1 Mbps
↳ 50bps

To retrieve data, correlate received signal with gold code. If correct gold code, correlation spikes and we are locked on to that satellite.

$$R_{xy}[k] = \sum_{m=-\infty}^{\infty} x[m]\, y[m-k]$$

Accuracy
- Signal acquisition errors (3m)
- ionosphere delays (≃ 5m)
- ephemeris errors (2.5m)
- satellite clock errors (2m)
- multipath (1m +)

Improve accuracy by:

① Signal diversity: use different signals L2C (1227 MHz) and L5 (1176 MHz). Given these are affected by ionosphere ~~effects~~ differently, can remove atmosphere effects
↳ only a couple of satellite

② Differential GPS: Reference stations at precisely known locations to measure error. If close to DGPS station, get cms of accuracy, but degrades as you go further away.

③ Remove selective availability: US used to add intentional time varying error which was ~~is~~ removed.

Signal Fingerprinting: perform offline survey to measure signal properties (strength) at a range of spatial positions (to comprehensively cover space)
↳ RSSI
↳ averaged over 10 measurements

↳ In online phase, scan local signals and perform pattern matching to survey points - indoor wardriving. Therefore can find nearest measurement point ⇒ assign all unseen APs reader sensitivity ≈ -100dB to make pattern matching easier
↳ Euclidean distance
↳ This can be extended to a ~~KNN~~ kNN

↳ But lots of challenging corner cases - general ambiguity
↳ Need accurate indoor measurement for offline survey!
↳ AB ~~gg~~ generally have low density
↳ Body shadowing ⇒ humans absorb signals
↳ Multiple devices + scanning costs are high

SENSOR FUSION

Need to fuse multiple sensor measurements to get robust idea of what is happening

$$Bel(\underset{\text{state at time } t}{\underline{x_t}}) = p(x_t \mid \underbrace{z_1, \ldots, z_t}_{\text{measurements}})$$

acts as a filter as it estimates based on current and past measurements only

⑪

Markov Model

propogation model

Propogation/predict
$$\underbrace{Bel^{-}(x_t)}_{\text{Prior}} = \int \overbrace{p\,\cancel{Bt}(x_t \mid x_{t-1})\,Bel(x_{t-dt})}^{\text{propogation model}}\,dx_{t-dt}$$

evaluate over all previous states

↳Prior

↳probs updated based on some model, eg constant velocity for location tracking → prior distributions

Run in 2 possible ways:
① Model all prob. distributions using mathematical models. → Kalman filter – not always possible.
② Sample to get arbitrary distributions

Correction / update
$$\underset{\text{posterior}}{Bel(x_t)} = \underbrace{\alpha_t}_{\text{norm factor}}\,\underbrace{p(z_t \mid x_t)}_{\text{measurement model}}\,\underbrace{Bel^{-}(x_t)}_{\text{prior}}$$

Kalman Filter: Recursive Bayesian filter – requires that you can write dynamics of system using linear algebra

$$x_t = F_t x_{t-\delta t} + w_t$$
$$P_t^{-} = F_t P_{t-\delta t} F_t^{T} + Q_t$$
Propagation

covariance error

noise terms

$$z_t = H_t x_t + v_t$$
Correction

↳measurement model

Idea is: (1) Propogate the state – error (Gaussian) increases
(2) Repeat propagation with measurement (Gaussian but thinner as more accurate)
(3) Multiply the two together, getting another Gaussian – representable with 2 parameters

In inertial GPS, compute position by concatenating a series of displacements and headings (dead reckoning), until next GPS measurement.
- ↳inertial sensors for displacement
- ↳gyro for heading
  - ↳ but subject to bias errors – lead to erroneous bending. Need to add biasing ~~so~~ in the kalman filter to correct for heading as well

Particle Filter: Errors due to noise accumulate very fast. Particle Filter uses Monte-carlo technique – simulating multiple hypotheses: (1) Where a person is and what their orientation is, (2) Assign each a probability that represents our belief in the particle

easily parallelizable

- ↳ Propogate: Shift each particle forward by IMU readings + add in random noise, hence cloud spreads.
- ↳ Correct: Any particle that crosses a wall get prob of 0. If measurement estimation system exists, boost particles most consistent with it.
- ↳ Resample: Randomly resample in proportion to particle probabilities using cumulative weight distribution. – forming cumulative weight is fundamentally sequential.

In localisation (start), we have lots of particles, eventually reducing the possible locations. In tracking, fewer particles are needed. Particle filter issues: →
① Each particle requires computation
② Results not deterministic
③ Hard to pick the right number of particles

⑫

MOBILE HEALTH

Provide healthcare support, delivery and intervention via mobile devices. Multiple different application types: (1) Education + awareness, (2) Helpline, (3) Diagnostic and treatment support, (4) Communication and training for healthcare workers, (5) Disease and epedemic tracking, (6) Remote monitoring, (7) Remote data collection. Challenge is mostly in validation of sensing results. Easy for clinical systems - compare to gold standard. But gold standard isn't available 24/7

Mobile Health Sensing: lower fidelity sensing at higher availability
↳ ambulatory sensing

↳ ① Applying established clinical sensors and techniques directly
↳ ② Taking clinical test and use available mobile sensors to do similar
↳ ③ Completely new techniques

PPG (Photoplethysmography): Detect blood volume changes using LED and photodiode, placed around finger / earlobe etc. To get it working on a watch has a number of challenges:

Having data in situ offers a lot of new research possibilities

↳ Transmissive doesn't work ⇒ use reflective
↳ Need to minimise powerdraw to preserve watch battery and sampling rate
↳ ~~Reduce~~ Green light is good but disliked at night
↳ Wrist is one of the worst locations to measure PPG - lot of tendons and ligaments
→ worn loose
→ external light
↳ forehead and ear are much better but not devices

∴ very hard to extract data as noise means SNR is very low. Also cadence lock means seemingly good data, for example in running, but this data is just wrong, just based on the runner's cadence.

Remote Diagnostics: sensors or AI can be used to take and transmit measurements that allow a remote expert to make a diagnosis. For example: Fundoscopy (eye imaging), using smartphone camera and inexpensive lenses.
↳ SpiroSmart - measures lung function using microphone

Simplifying Access to Health Data: DeepMind Streams - simple product to collate patient information. No AI, just visualisation. Same with Apple Health Records

PREVENT ⟶ DETECT ⟶ SLOW / REVERSE ⟶ SUPPORT
very hard to do
↑ ~~Scre~~ Screening
SLOW / REVERSE ⟶ SUPPORT: requires clinical guidance - challenging

Screening ↳ test for it in someone not known to have the disease as early as possible while it is easier to treat. This offers a number of advantages: (1) Cheaper, (2) Can reach more people and remote places, (3) Reduces self-selection aspect, (4) Use always on longitudinal data.

$$\text{Sensitivity (Recall)} = \frac{\text{True Pos}}{\text{True Pos} + \text{False Neg}}$$

$$\text{Specificity} = \frac{\text{True Neg}}{\text{True Neg} + \text{False Pos}}$$

(Precision)

$$\text{Positive Predictive Value} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Need to be careful for low prevalence diseases, for example (AFib for young people has low prevalence). Apple turn on only for PPV > 60%
↳ PPV = 17% for <55
50% for 55-60
71% for 60-65

Also, need to include prevalence in training data, not balance each class!

Base rate fallacy means we can overly read into test information
↳ overrun health services and reduce trust in

⑬ **Detecting Adverse Events** : use accelerometer to find falls
↳ challenging to get ground truth data.

↳ Microphone to monitor coughs and sneezes to:
① Count number of coughs to check if have a respitory problem
② Classify type of cough
③ Track progression of cold/flu.

Participatory Sensing : Create sensing network to map some quantity at scale otherwise not possible. But requires users to donate their data. ↳ eg Pollution

PREVENTION

Interpretation of Public Health Guidelines : mobile sensing can be used to personalise guidelines - making them more relevant and likely to be followed ⇒ eg. Heart Points / Intensity Minutes. Measure type of activity using heart rate.
↳ though need to be wary of high HR not due to activity. Correlation of activity and high HR, does not mean ~~that~~ causation in both directions!

MOBILE ROBOTS FOR ROBOTIC SENSOR NETWORKS

Challenges of autonomous robots : → how to model and perceive the world
↳ how to process information and exert control
↳ how to reason and plan in the face of uncertainty

* any-comm algorithms (with graceful degradation) or any-time algorithms (gradual improvement)

Multi-Robot Systems → homogeneous vs heterogeneous.

① Centralized vs Decentralized Architectures
↳ One control / estimation unit communicates with all robots to issue commands
↳ requires synchronized comms channels + single point of failure
↳ Scalable, robust to failure, often asynchronous but sub-optimal performance though can converge to optimum*
\+ comms challenge

② Communications can be implicit (has observable states with information exchanged through observation) or ~~out~~ explicit (unobservable states and needs to be communicated explicitly)

Coverage classes → ① Blanket : deploy sensors in a static arrangement to cover an area

↳ ② Barrier : deploy sensors in static arrangement to maximise probability of undetected penetration through the area

↳ ③ Sweep : move group of sensors across coverage area to achieve balance between maximising n. of detections / time and minimising missed detections/area

Tessalation
↳ Voronoi diagrams (given a set of points) partitions a plane into regions based on which seed the particular area is nearest to.
generators → points; Voronoi cells → regions
↳ Allows optimisation of configuration of robots → one robot per Voronoi cell.

Density function $\phi(x)$ describes importance of different areas in space

$$M_{V_i} = \int_{V_i} \phi(x)\,dx$$
(mass)

$$C_{V_i} = \frac{1}{M_{V_i}} \int_{V_i} x\,\phi(x)\,dx$$
(centroid)

Centroidal Voronoi Tessalation
Generator point of each Voronoi cell is also centroid, minimising the following distance:

$$H(p) = \sum^{n} H(p_i) = \frac{1}{2} \sum_{i=1}^{n} \int_{V_i} ||p_i - x||_2^2\,\phi(x)\,dx$$

⑭ Voronoi Tesalation becomes a CVT when generators coincide with cell centroids.

Move generators by considering : $\frac{\partial H(p_i)}{\partial p_i} = - M_{v_i} (c_{v_i} - p_i) = 0$

Topologies

fully connected

star topology

random mesh

↳ decentralized coordination

centralised / decentralised coordination

$$u_i = \underline{\dot{p}_i} = k(\underline{c_{v_i}} - \underline{p_i})$$

What kind of controller

how to compute centroid position

↳ Lloyds Algorithm

① Construct Voronoi partitions for generators
② Compute centroids of these region
③ Move generators to centroids and start over.

Convergence is guaranteed

↳ how to compute robot positions
↳ localisation

Distributed Estimation : attempt to estimate the value of a variable by using sensor readings of multiple robots.

↳ Collaborative Localisation : use relative inter-robots observations to communicate position estimate. We fuse relative observations to go from local frame of reference to global.

↳ Can use a particle filter or Kalman filter

Range & Bearing Model

$r_{mn}^{[i]}$: range with center $x_m^{[i]}$ to x_n
$\theta_{mn}^{[i]}$: bearing from $x_m^{[i]}$ wrt x_n

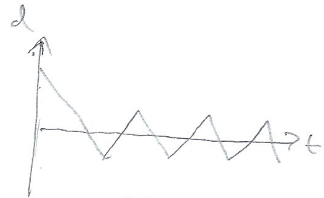
relies on the fact that robots can see each other and can see each other this detection data

detection data $\boxed{d_{mn}} = \langle r_{mn}, \theta_{mn}, x_m \rangle$

$$\underbrace{p(x_n \mid d_{mn})}_{\text{sensor model}} = \eta \sum_{\langle x_m^{[i]}, w_m^{[i]} \rangle \in X_m} \phi \left(\begin{bmatrix} r_{mn}^{[i]} \\ \theta_{mn}^{[i]} \end{bmatrix} ; \begin{bmatrix} r_{mn} \\ \theta_{mn} \end{bmatrix} , \Sigma \right) \cdot w_M^{[i]}$$

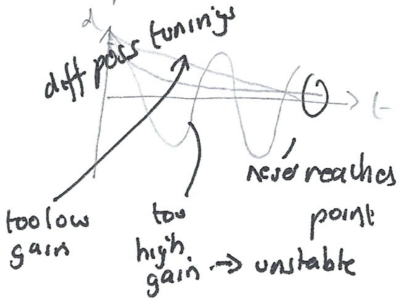
Robot Control

↳ Bang-Bang Controller : → if too far left, full left, min right
too far right, full right, min left
else, full left, full right



↳ Proportional Control → turning control = K * error
↳ distance to trajectory

$$u(t) = K_p e(t)$$



↳ Proportional Integrative Derivative Controller :

$$u(t) = K_p e(t) + \underbrace{K_i \int_0^T e(\tau) d\tau}_{\text{accumulated error}} + \underbrace{K_d \frac{de(t)}{dt}}_{\text{derivative}}$$

derivative
↓
dampener