Crickets

Tutorial on using cricket location system

Cricket Goals

• Research prototype
  • build and then evaluate
• Useful mainly indoor environments
  • walls, ceilings not too far
• Recognize spaces, not just physical position
  • good boundary detection is important
    • doors, floors, etc.
• Preserve user’s privacy
  • Big-brother can be a bother
    • user has choice to reveal location
Features

- Distributed architecture
  - No wired infrastructure
  - Easy deployment (no satellites)
  - Low maintenance
- Users are not tracked
  - Listeners are passive
  - Large number of listeners w/o interference
- Integrates with a wide range of resource discovery systems

Cricket: Private location-support

No central beacon control or location database
Passive Listeners + Active Beacons
Finding the distance

• Basic formula: \( \text{distance} = \text{speed} \times \text{time} \)

• want to find the distance

• we know the speed

• How do we figure out time?

• there are several choices

• 1: Measure round-trip time (like radar)

• this violates some of our goals -- which ones?

Finding the distance

• 2: Synchronized clocks

• receiver knows exactly when transmitter sent signal

• need very accurate clocks

• send a signal to first sync clocks; then send second signal

• does this work?
Finding the distance

- 3: Use two different speed signals
  - both start at same time
    \[ d = s_1 \times t_1 \]
    \[ d = s_2 \times t_2 \]
- We measure delay: \[ m = t_1 - t_2 \]
  \[ t_2 = \frac{m \times s_1}{s_2 - s_1} \]
  \[ d = \frac{m \times s_2 \times s_1}{s_2 - s_1} \]
- Cricket does this
  - RF is really fast compared to US

Location Estimation

- Distance estimation via coupled RF and ultrasonic signals
  - Beacons send information on the RF channel with concurrent ultrasonic pulse

![Diagram](attachment:image.png)
Uncoordinated Beacons

- Multiple beacon transmissions are uncoordinated
- Different beacon transmissions can interfere
- causes inaccurate distance measurements at the listener

Multiple Beacons

- Beacon transmissions are uncoordinated
- Ultrasonic signals reflect heavily
- Ultrasonic signals are pulses (no data)
  These make the correlation problem hard and can lead to incorrect distance estimates
Solution

- Carrier-sense + randomized transmission
  - reduce chance of concurrent beacons
- Bounding stray signal interference
  - envelop all ultrasonic signals with RF
- Listener inference algorithm
  - Processing distance samples to estimate location

Bounding Stray Signal Interference

- Engineer RF range to be larger than ultrasonic range
  - Ensures that if listener can hear ultrasound, corresponding RF will also be heard
Bounding Stray Signal Interference

\[ \frac{S}{b} > \frac{r}{v} \]

(RF transmission time) (Max. RF-US separation at the listener)

- No “unaccompanied” ultrasonic signal can be valid!

Bounding stray signal interference

- Envelop ultrasound by RF
- Interfering ultrasound causes RF signals to collide
- Listener does a block parity error check
  - The reading is discarded...
Problem: Closest Beacon May Not Reflect Correct Space

Correct Beacon Placement

- Position beacons to detect the boundary
- Multiple Beacons per space are possible
Implementation

- Cricket beacon and listener

![Diagram of Cricket v1 Prototype](image)

- Cricket v1 Prototype
  - Ultrasonic sensor
  - RF module (xmit)
  - RF antenna
  - Listener
  - Beacon
  - Atmel processor
  - RS232 i/f

Cricket v1 Prototype

![Prototype image](image)
Cricket Beacon LEDs

- **Debug Switch = UP**
  - Green LED = Transmit
  - Red LED = Carried Sensed
- **Debug Switch = Down**
  - Green LED = Every 5th transmission
- **At Startup**
  - LEDs flash version number
  - Red on, Green flash count = Major #
  - Green on, Red flash count = Minor #
- **Power Switch**
  - Up = On

Cricket Beacon Antennas

- **Receive Antenna**
  - For sensing transmission of other beacons
- **Transmit Antenna**
  - Limit transmission distance
  - Should not touch ultrasound
  - Should not cover receive antenna
Listener LEDs

- Green Flash
  - Received valid RF and ultrasound
- Red Flash Once
  - Received Radio, but not ultrasound
- Red+Green Flash
  - RF Error (e.g., parity error)
- Red and Green always on
  - Listener not working correctly
- Power On
  - Both LEDs flash together once

Off  On
Power Switch
Cricket Version 2

- Each device can be configured as:
  - listener
  - beacon
  - listener & beacon
- Same H/W as Berkeley Mote
  - Runs tiny-os
  - Connector for sensors
- Lots of configurations possible
  - need special connector to configure
Software Components

- Cricket Beacon 1
- Cricket Beacon 2
- Cricket Listener
- cricketd
- Application 1
- Application 2
- Application 3

Binary Data Over RF

- ASCII data over serial port 9600 baud
- ASCII data on TCP sockets (port 2947)

cricketd

- Background program (demon) that reads serial port and writes data to a socket
- Command line arguments (defaults work correctly on ipaq)
  - -T k Version 3 Listeners (with LEDs) (default)
  - -T c Version 2 Listeners (without LEDs)
  - -S <port> Socket port number (default is 2947)
  - -p <dev> Serial port device name (default “/dev/ttySA0”)
  - -s <baud> Baud rate of serial port (default is 9600)
  - -h Help
  - -D <num> Debug level
Cricket Listener Output

- Strings reported from Listeners
  - When good RF and good ultrasound pulse heard:
    - "$Cricket2,ver=3.0,space=MIT7,id=20,dist=4F,duration=1A"
  - When only good RF heard, no ultrasound heard:
    - "$Cricket2,ver=3.0,space=MIT7,id=20"
  - When RF detected, but parity error detected:
    - "$Cricket2,ver=3.0,err=rf"

Speed of Sound

- Listener reports distance and duration in 15.625 KHz counter cycles (64 microseconds each).
- Assume speed of sound is 344.49 m/s then 22.047 mm/cycle
- For 343.75 m/s = 22 mm/cycle
- Need to subtract 36 units for delay from end of RF to start of US transmission.
So where are you?

- Telnet to cricketd (on correct port)
- Get names of beacons within range
- Get distances from beacons
- Lookup beacon location in database
  - Or use beacon name (longer transmission)
  - Triangulate (compensate for temp)

So where are you?

- Beacon name may tell you room
  - That may be enough
- May want to know relative movements
  - As you walk around the room
  - No climbing on tables
- Can you do it using two beacons?
  - Can you do it without calibration?
Two beacons

- Put them along the wall
- Come very close to one of them
- Now know distance between them
- Given distances from both
  - Before and now \((d_1,d_2) & (e_1,e_2)\)
  - Can find relative movement
  - Two solutions! No problem, why?
  - Ex. Doom virtual world

Orientation

Orientation relative to \(B\) on horizontal plane

- Beacons on ceiling
- Mobile device (parallel to horizontal plane)
- Cricket listener with compass hardware
Hardware Design

• Main site for cricket stuff:
  http://nms.csail.mit.edu/cricket
  • http://nms.csail.mit.edu/cricket/fab
  • Need user & password
  • http://nms.csail.mit.edu/cricket/distrib
  • Need user & password

More accurate readings

• Readings are not accurate -- reflections, noise, etc. cause inaccurate values
• How to compute current position?
• How to compute when in motion?
When standing still

- Values still fluctuate
- Want to average over many values
  - takes long time
- Want to ignore “outliers” -- the values that do not make sense

Least Squares Method

\[
\sum_{i=1}^{n} (||\hat{\phi}_i - p_i|| - d_i)^2
\]

- Given a bunch of readings, find an estimated location that minimizes:
  - \(d\) is distance from beacon, \(p\) is beacon location, and \(\phi\) is estimated location
  - \((\phi - p)\) is estimated distance from beacon
  - \((\phi-p) - d)^2\) is square of difference
  - minimize the sum over all measurements
Kalman Filters

• Each measurement has a probability density function associated with it.
• It is specified by standard deviation (sigma) and by variance (sigma squared).
• Given two measurements, each with its own probability or conditional density.
  • compute probability density function around new location as a weighted average of both.

Kalman Filters for dynamic system (when moving)

• In addition to location measurement (and its probability function) also have
  • velocity plus noise
  • compute guess as to new position in future
    • based on time and velocity.
  • The velocity noise factor
    • spreads out the probability density function of the location in the future.
  • New real measurements combine with predicted location as before
    • measurements usually have more accuracy than predicted value: so get more weight.
When tracking moving cricket

- Least squares does not work because hear beacons at different times
- Use Kalman Filters
  - combined with Least Squares and outlier rejection

Hybrid Approach

![Flow chart for the tracking algorithm. The picture shows outlier rejection and LSQ explicitly; all the other steps are part of the extended Kalman filter. The “measurements” are the distance samples obtained as triples. The “output” at each stage is the position estimate φ.](image)

Figure 3: Flow chart for the tracking algorithm. The picture shows outlier rejection and LSQ explicitly; all the other steps are part of the extended Kalman filter. The “measurements” are the distance samples obtained as triples. The “output” at each stage is the position estimate φ.
When Kalman Filter goes bad ...

- Reject new values that look bad
  - but if too many get rejected then maybe those are the right ones
- To get back on track, use Least Squares
  - but need simultaneity
  - how to do that?
- Listener sends special signal with a nonce
  - heard by all listeners
  - all beacons then know distance for listener
  - beacons send back the distance with nonce
  - listener gets data as if all beacons chipped at same time

Figure 4: Flow chart of a mobile device using a hybrid architecture.
Alternative Method

- Detect with cricket is at rest
- Not so easy to do
  - we use tilt meters & camera
  - tilt meters are noisy
  - optical flow of camera
    - not fool-proof