Simon Dobson

Systems Research Group
School of Computer Science and Informatics
UCD Dublin, Belfield, Dublin 4, Ireland

http://www.ucd.ie/csi

simon.dobson@ucd.ie

Leveraging the subtleties of location

- or -

How to know where you are even when you don't know your location

Overview

There are many location systems, but it's still hard to build robust location-based services

- Hard to get reliable, long-term, full-spectrum location
- Hard to express the behaviour we want to exhibit

So we need more sensors – right?…

My aim in this talk is to take a slightly contrarian position on this debate

- There *is* no one best location model; nor is the problem uniquely solvable using location technology
- One can build a location-based service with little or no location hardware
- We can fuse information, often without too heavyweight reasoning

Direct solutions

Location systems open up some exciting areas

- Device-centric: GPS, PlaceLab, Crickets → Device knows where it is
- Infrastructure: RFID, GSM, cameras, Ubisense

Building or other infrastructure knows where device is

But most are far from being ideal platforms

- Expressibility: have to match the low-level model of the sensors, not the high-level models in which applications are usually phrased
- Availability: expose applications to the noise and drop-outs
- Accuracy: observations are evidence of fact, *not* facts themselves

Shooting yourself in both feet and then adding more legs

Leveraging *all* you know

Any context-aware system retains a body of knowledge about its users and their activities

Context: information about the operating environment, understood symbolically

- Diaries, task lists
- Default observed behaviours , patterns

How do you answer the question “Where’s Waldo?”

- At least 18 recognisably different answers (in English, anyway)
- Known, approximate, unknown; discrete or continuous; based on observation or inference; based on other ontologies
- Can be weak, but stays around when the sensors fail
- Another source of evidence of fact to throw into the mix

The full taxonomy is in the paper: here I only extract the highlights

Taxonomy – 1

Co-ordinates and named spaces
- "At 55deg3minN, 3deg45minW"
- "In A1.15"

Surprisingly enough no-one's ever told me their location this way…

Temporal
- "At 1000 he will be…"
- "At 0800 he was…"

Functional spaces
- "In a conference room"
- "In his office"
- "In Willard's office"
- "In his car"

Spatial
- "Within 250m of…"
- "Between … and …"
- Either at … or … or …"

Relative
- "With Willard"

…and you have 17 other ways to locate Willard

Other location models may reduce this uncertainty

Taxonomy – 2

Proxy
- "His badge was last seen at …"

This is actually usually the case with device-based location systems

Located task
- "Meeting Willard"

By negation
- "Not …"

Surprisingly many applications can work with negated information

Default
- "At this time he is often/usually at …"

…and don't under-estimate how predictable and regular many people are…

Non-located task
- "Out/on holiday"

Unknown
- "No idea"

What does this tell us?

Contextual layering

- Describe different aspects of the world independently
- Often possible to infer extra information in one layer (location) using inference from others (identity, schedule, …)
- Often quite structured, so only quite lightweight reasoning

Given a rich model the “right” adaptive behaviour often emerges from the context

- Changes may look arbitrary when seen as GPS
- …but be perfectly logical in terms of being “with Willard”
- …and so this is the location model this particular application s hould use to express its behaviour
- Changes occur when (and only when) the place changes, for some suitable definition of “place”

Dobson and Nixon. More principled design of pervasive computing systems. LNCS 3425. 2004.

The precision trap

One criticism of this argument is that one cannot trust information coming from inference over very noisy information

- Very few people keep their diaries up-to-date

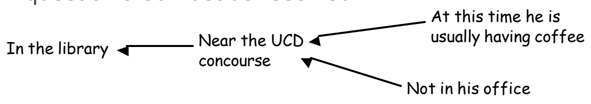
But many applications don’t actually *need* precise information, and sometimes very little will do

- Work with where someone *isn’t*, regardless of where they *are*
- Perfectly acceptable for information to age, in a controlled way
- Current location may not be significant – looking for restaurants when on a train is a good example
- Some references are unstable in space and/or time

Implications for programming

There *is* an ideal location system – but it doesn't come just from improving hardware

- …although that's obviously good too
- Work on fusing uncertain information; multiple answers to any question that must be resolved



Programming with this uncertainty

- Keep the fuzziness around, use for decisions (and un-making bad decisions)
- Learn which information is reliable? No false certainty
- Express conditions along the "best" model, convert from what's available

Conclusions

Five things to take away

1. Leverage all the location information we have – don't focus solely on "dedicated" location systems
2. Different views are structurally related in interesting ways, that often allow behaviour to "emerge" from context
3. Knowledge is in one piece, location information is not something that only appears in one ontology
4. Information richness can be used to compensate for any local deficiencies
5. We must program with the uncertainties explicitly

This is what pervasive reasoning *means*, and we should make sure we use all the richness of the contextual information we maintain