Proactive Indoor Navigation using Commercial Smart-phones

Balajee Kannan, Felipe Meneguzzi, M. Bernardine Dias, Katia Sycara, Chet Gnegy, Evan Glasgow and Piotr Yordanov
Background and Outline

• Why did we build that app? “Google Core AI”@CMU

• Challenge to create **usable** AI components for an App library

• Involving **producers** and **consumers** to motivate application

• Two components produced for a Proactive Indoor Navigation App

  • Indoor Localization

  • User Prediction
Core AI Components

- User Prediction (Producer Team)
  - Felipe, Katia and Piotr
  - Decision theoretical intention recognizer
- Indoor Navigation (Consumer Team)
  - Balajee, Bernardine and Evan
- App Team
  - Felipe, Balajee and Chet
Architecture Overview

Navigation App

Indoor Localization

Path Planning

Map Management

User Prediction

RSSF Database
Robot Map
Wifi Signal
Compass
Accelerometer
Floor Map

Particles

Belief State

Waypoints

Map Annnotations

Directions

New Annotations

UI

Map Management

Destination

Annotations

Friday, 22 March 13
Indoor Localization

- Indoor localization performed with sensors in the mobile phone
- Signal strength fingerprinting (precise, high CPU usage)
- Dead reckoning (low CPU usage, error prone)
RSSI Database Construction

• Requires a map correlating APs signal in a building with precise locations
• Built using a robot equipped with accurate sensors (Rangefinder and Gyroscope)
• Tele-operated in each floor of a building
• Creates a map of empty space
• Map is shared with all mobile phones entering the building
Intention Prediction

- Based on a decision-theoretical model behaviour
  Markov Decision Process (MDP)
- An MDP is defined in terms of
  - An initial state $S_0$
  - A transition model $T(s,a,s') = P(s'|a,s)$ (Markovian)
  - A reward function $R(s)$ — sometimes expressed as $R(a,s)$
- A solution to a MDP must specify what the agent should do for any state. Such a solution is called a **policy**
Intention Prediction

• Based on a decision-theoretical model behaviour
  Markov Decision Process (MDP)

• An MDP is defined in terms of
  • An initial state S0
  • A transition model $T(s,a,s')$ — $P(s'|a,s)$ (Markovian)
  • A reward function $R(s)$ — sometimes expressed as $R(a,s)$
  • A solution to a MDP must specify what the agent should do for any state. Such a solution is called a **policy**
Intention Prediction

- Based on a decision-theoretical model behaviour
  Markov Decision Process (MDP)
- While, the solution to MDPs usually assumes a perfect decision-maker to generate a policy
  \[ \pi^*(s) = \arg \max_a Q^*(s, a) \]
- We define a stochastic policy
  \[ \pi \approx (a | s) = \frac{Q^*(s, a)}{\sum_{a'} \in A Q^*(s, a')} \]
- That yields the probability of an action being chosen, proportionally to its optimality
Generating a prediction

- Given a probability estimate of the current user-position (Belief state)
- Generate a tree of future paths using the stochastic MDP policy, such that:
  - Actions used to create successor states have a minimum probability
    \( \pi(a|s) \geq thr \)
  - All possible successor states to such actions are added to the tree
  - Only states along an increasing gradient towards target states are followed
Hierarchical Path Planning

• Algorithm based $D^*-lite$

• Hierarchical map representation in two levels of granularity

  • Higher-level structural graph (multiple rooms, floors, buildings)

  • Low-level grid of the free space (single floor)
Putting it all together

• Navigation App was built using three separate Android services controlled by the main App

• Communication via Android messaging

• Profiling of each component led to substantial design changes
Navigation Step-by-step

• Step 1 - Inputs
  • RSSI database
  • Floor plans for target building
  • User annotations or learned habits
Navigation Step-by-step

- Step 2 - Particle filter update
- Particles generated by the PF using the WiFi data (1 Hz)
- Particles updated by the dead-reckoning system (30 Hz)
- Particles outside known space discarded
Navigation Step-by-step

- Step 3 - Prediction update
- Particles from the Indoor Localization component are converted to a Belief-State
- Prediction tree is generated from most likely current state (beyond a certain threshold)
Navigation Step-by-step

- Step 4 - Path planning
- Most likely destination is extracted from the prediction tree
- Optimal path is generated taking into consideration obstacles along the way
- Path-planning performed for the same floor and between floors
Key Insights and Results

- Producer/consumer model for AI components interesting motivator
- Major bottlenecks
  - WiFi based localization - required adjustments on update frequency
  - MDP Policy recalculation - whenever possible done via external service
- Accuracy and runtime results
  - Variance in destination prediction when in long corridors
  - Magnetic disturbances in the building have large effect on localization
Potential for Future Work

- RSSI database acquisition
- Implement autonomous robot scanning
- Use crowd sourcing for RSSI database updates
- MDP learning and solver algorithm
- Generate a stochastic policy using policy iteration (anytime algo)
- Online learning of user habits
Questions?
Dead Reckoning

**Heading**

- Accelerometer + Magnetometer
  - Externally referenced –
    - Bounded error
    - Magnetic interference indoors
  - Gyroscope
    - Low noise and high accuracy
    - Not susceptible to interference
    - Error growth is unbounded over time

**Distance Measurement**

- Peak Detection Filter
  - Each pair corresponds to a step

- Variance Threshold
  - Calculate running standard deviation
Signal Strength Fingerprinting

- Automated WiFi signal strength database generation using a pioneer robot
- 2-D dynamic robot map of the environment
- At runtime, the distance is calculated as a weighted average of the nearby calibration points to reduce noise

- Accurate, high density signal strength database in a short time
- Shape and structure of the laser map allows us to speed up our pose estimation and reduce computation
Particle Filter

Initial Distribution: Uniformly random over entire environment

• Step: Use dead reckoning model to update particles

If there are new observations, update the probability of each particle

• Step a: Use robot map to identify and remove particles that lie on walls
  
Step b: When a Wifi reading is received, update particle weights

Re-arrange the samples to be concentrated in the most important areas

• Step: Re-sample using importance resampling: a new set of $n$ particles from the old set proportional to its weight