

A BDI APPROACH TO AGENT-BASED MODELLING OF PEDESTRIANS

Nicole Ronald and Leon Sterling
Department of Computer Science and Software Engineering
The University of Melbourne, Victoria 3010, Australia
Email: {naron,leon}@cs.mu.oz.au

KEYWORDS

Transport modelling, agent-based simulation, BDI architecture

ABSTRACT

The modelling of pedestrian behaviour in a real-world environment is a complex problem, mainly due to the unpredictable nature of human decision making. Agent-oriented simulation moves away from the traditional all-knowing and “controlling” simulations and towards reality, where pedestrians exhibit different behaviours depending on their knowledge of the environment and other personal characteristics. We look the behaviours that pedestrians may exhibit and the different techniques used for pedestrian modelling. We then explore whether the belief-desire-intention (BDI) architecture is appropriate for this domain. The simulation requirements and constraints are presented, followed by a high-level conceptual design using Prometheus, an agent-oriented design methodology, and a discussion of the implementation using JACK, an agent-oriented programming language based on the BDI architecture. Although the BDI architecture is useful for high-level decision making, further work is required in representing and updating the environment.

INTRODUCTION

There is a need to model pedestrian behaviour for a range of applications including event planning, resource usage, and urban planning. For example, the organisers of a large sporting event require information on what areas are likely to be congested so that management strategies can be developed and tested before the event. Similarly, the designers of a shopping mall might be interested in how people move around their intended design so they can place shop entrances and seating in useful locations.

Most of the models developed so far fall in one of two categories: large-scale models producing aggregate results and smaller, disaggregate models. Less work has been done for pedestrian modelling in between these two extremes. It is useful to have an understanding of how people will behave in certain situations, such as at a large sporting event or after a change to their environment, during the planning process.

Agent-based simulation appears to have potential

for pedestrian modelling. Each pedestrian could be modelled as an autonomous agent with its own knowledge and goals. This representation is closer to reality than traditional simulation methods as it requires less abstraction. The purpose of our research is to compare different agent-based approaches to modelling pedestrian behaviour, in particular the benefits to disaggregate modelling. The approach we discuss in this paper is based upon belief-desire-intention agents.

This paper is divided into several sections. Firstly, we examine at the need for pedestrian modelling and the properties of the pedestrian system. We then review some existing models and techniques. The design and implementation of our model is covered, followed by a discussion.

BACKGROUND

We discuss some of the interesting properties of pedestrian systems and the history and need for transport modelling, followed by a survey of pedestrian modelling approaches.

System properties

The Australian Pedestrian Council defines a pedestrian as “any person wishing to travel by foot, wheelchair or electric scooter, throughout the community” (Australian Pedestrian Council 2004). There are many reasons for walking, and the manner in which we walk changes depending on the purpose.

“Commuters scurry; shoppers meander; bush-walkers trek; power-walkers stride; lovers stroll; tourists promenade; protesters march ... But we all walk.” (Australian Pedestrian Council 2004)

Transport systems are constrained, sometimes weakly. For instance, people cannot cross the road whenever they feel like it - they should find a suitable place (such as an intersection) and wait until it is safe. They also should travel on the pedestrian network (eg. designated paths) at all times, however if it becomes too congested, pedestrians may overflow onto the road or surrounding parkland. A stricter constraint is that pedestrians cannot walk through solid objects or on water.

Most transport modelling techniques have focused

on the modelling of cars and vehicles on the road network, as congestion and environmental effects of car travel are pressing problems for cities. The recent interest in environmentally sustainable transport modes, however, has led to an interest in providing better infrastructure and facilities for cyclists and pedestrians and therefore a need for improved methods of modelling their behaviour.

Pedestrian behaviour is usually individual-based and autonomous. In most cases, we decide where we want to go and how to get there without being told explicitly. This behaviour lends itself nicely to the agent-oriented paradigm in principle.

Previous work

There are many approaches to modelling pedestrian behaviour, which can be divided into two schools. The first school is the “civil engineering” approach. This is concerned with forecasting demand so that decisions can be made about provision of new infrastructure. The main outputs of these models are numbers of people travelling along various routes and the algorithms used are frequently based on traditional vehicle modelling algorithms. They are generally macroscopic or aggregate models, where the smallest detail of a pedestrian’s movement is the locations they visited and the paths they used to get there.

The second school is the “architecture/urban geography” approach. This group is interested in how people move around areas, in particular how design and location of certain attractions influence their movements. These models are usually microscopic, in that they model a pedestrian’s path in more detail, usually in terms of steps or small grid squares. They are usually developed for small areas only, although some have been expanded to cover entire cities. Some models combine both approaches and as a result are very flexible regarding the type of areas they can model.

Mathematical models, such as regression models and Markov models, have also been used to model pedestrian behaviour at aggregate levels (Harney 2002). Regression models are of limited use as their only output is pedestrian volume at a particular time and place. Markov models are also of limited use due to their complexity.

Physical models have also been used to model pedestrians. Helbing (Helbing et al. 2001) used the notion of attraction and repulsion to model microscopic behaviour and has developed complex equations to model a range of pedestrian behaviours, commonly referred to as the “social force” model. Hoogendoorn and Bovy (Hoogendoorn and Bovy 2003; Hoogendoorn and Bovy 2004) used the same starting point of basic mechanics formula and developed a three-layered model encompassing

activity choice, wayfinding, and walking. This model attempts to minimise the cost of walking and was applied to a multi-modal transfer station.

A similar approach is the use of cellular automata (CA), where pedestrians occupy cells on a grid and move according to some simple rules (Dijkstra et al. 2001; Henein and White 2004; Narimatsu et al. 2004). Most of these models used the Schreckenberg-Nagel approach to modelling vehicle traffic using CA as a starting point (Nagel and Schreckenberg 1992). This has been shown to be useful for disaggregate models with minimal activity choice. AlpSim (Gloor et al. 2004) combines a cellular automata approach with a graph-based representation of the environment to take advantage of the benefits of both map representations, specifically higher-level planning which is very complex using a grid.

Traditional time-based simulation has also been used in industry. PAXPORT, developed by the consulting firm Halcrow, has been used to model pedestrian movements in airports, train stations, and sporting venues. It provides aggregate measures of flow and level-of-service in a graph-based environment. It was recently used to model behaviour in the Sports and Entertainment Precinct, Melbourne in order to select a design for a new bridge to be built for the 2006 Commonwealth Games (Ronald 2004).

MODELLING ASPECTS

Overall Architecture

Transport systems can be broken down into three main concepts: user, vehicle and environment. The user has a perception of attributes of the environment and their vehicle, and needs to guide their vehicle through the environment. The vehicle interacts with and changes the environment. The environment is constantly updated with the new locations of vehicles and provides perceptions to vehicles and users.

For a driver, the user is the driver in the car, the vehicle is the car, and the environment is the road network and the other vehicles. For pedestrians however, the user and vehicle are essentially the same object: a human. However, most of our walking is done subconsciously and therefore it is permissible to separate these two concepts. We can define the user as the human’s brain and the vehicle as the human’s legs.

Given the high-level decision making role of the user, the belief-desire-intention architecture would be useful for modelling user behaviour in transport systems.

The BDI Architecture

The philosophical component of BDI is based upon

practical reasoning. Practical reasoning is defined as reasoning toward actions, as opposed to theoretical reasoning, which is reasoning about beliefs. Practical reasoning can be broken down further into two activities: deliberation (deciding what goals to achieve) and means-end reasoning (how to achieve a goal) (Wooldridge 2000). Another nice feature of the BDI architecture is the ability to act in both a reactive and proactive manner, however there is a danger of being too reactive or too proactive.

The key concepts in the BDI architecture are:

- beliefs: what I know or don't know about the world;
- desires: what I want to do;
- intentions: how I plan to do what I want to do.

Previous work, as described earlier, has used cellular automata and other methods to develop their agent-based models of pedestrian behaviour. BDI fits our problem well, in that:

- People have beliefs about the environment that affect their decisions (eg. "The main street is always crowded at lunchtime - I will take another route.");
- People have desires to do something or to visit somewhere. This is more obvious with vehicle travel as people do not drive around the city just for something to do, whereas people will sometimes walk somewhere just because it is there. If people are wandering "just because", then that is still a desire;
- People have plans or procedures of deciding where to go first, how to get there, and how to create a path to follow;
- If a route is blocked due to congestion or temporary infrastructure, a new plan can be formulated and a new path taken to reach a location.

Designing with Prometheus

Prometheus is a methodology developed for specifying agent-oriented software systems. Although there are several design methodologies that could be used for this system (Bergenti et al. 2004), Prometheus was chosen because of its maturity (a book was recently published (Padgham and Winikoff 2004)) and because the concepts used in the methodology tie in with the concepts used in JACK Intelligent Agents, our chosen implementation language. Prometheus consists of three design phases:

- system specification phase: identify functionalities, inputs, outputs and shared data sources;
- architectural design phase: determine agents required and their interaction; and
- detailed design phase: internal design of agents.

The resulting design is a combination of forms and diagrams, which clearly describe the percepts, action, environment, agents, capabilities and plans in the system. Although not tied to any implementation, it fits nicely with JACK Intelligent Agents. It would not be unfamiliar to those familiar with UML for object-oriented design (Rumbaugh et al. 2004).

System specification

The system specification involves identifying system goals and functionalities, developing the interface between system and environment, and developing use case scenarios.

Firstly, the system goals and possible subgoals need to be established. An example of this is shown in Figure 1, which shows three goals (visit attractions, arrive at the stadium at a reasonable time, move through the environment) and their subgoals using an oval shape. One of the goals is to move through the environment, with the subgoals of satisfying network constraints and taking a reasonable path.

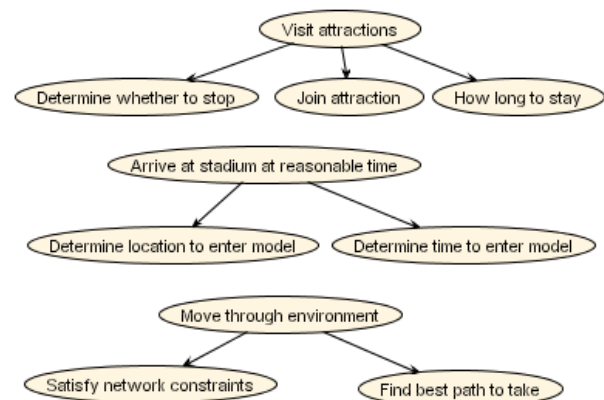


Figure 1: Displaying system goals using Prometheus.

Goals can then be grouped together to create functionalities. Scenarios can also be developed. These consist of steps such as percepts, messages, goals and actions. The system interface involves determining the actions and percepts of the interface to the environment.

Architectural design phase

The architectural design phase involves grouping the functionalities into similar areas, developing agents to control each area and specifying the interactions between agents.

The functionalities can be grouped into areas by investigating the data that is present in the system and which data is required by each functionality. This is then used to determine the agents required and what functionalities they should have. Figure 2 shows an example of a coupling diagram. Each data source is represented by a yellow cylinder and the functionalities by rectangles. The arrows signify whether the functionalities read

(arrowhead at the functionality end) or write (arrowhead at the data end) data.

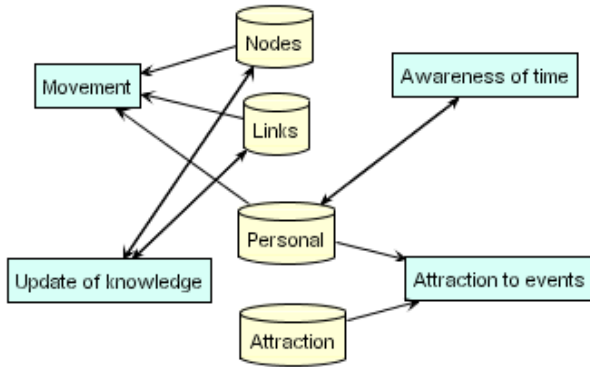


Figure 2: A coupling diagram in Prometheus.

As part of this phase, communication between agents can also be specified at a high level.

Detailed design phase

In the detailed design phase, the agent's capabilities, events, plans and data structures are developed in more detail. An example of a detailed design for the Pedestrian agent is shown in Figure 3. This shows the events (percepts: stars shapes; actions: arrow shapes; messages: envelope shapes), capabilities (rounded rectangles), and plans that the agent requires. The arrows again signify the incoming and outgoing nature of events.

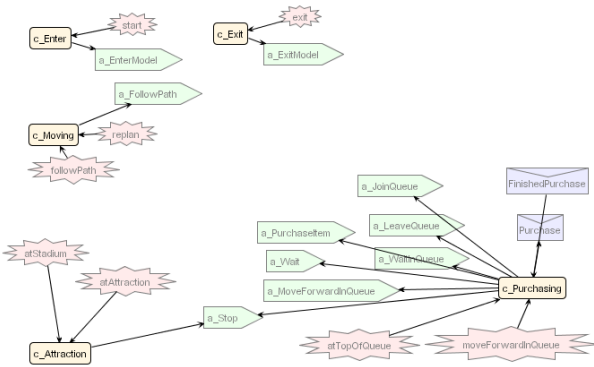


Figure 3: A detailed design for a Pedestrian agent.

Capabilities are similar to modules in that related plans, events, and data can be combined together in a coherent manner. In this model the pedestrian has a capability for each of its main activities.

Events can be actions (affecting the environment in some way), percepts (knowledge coming from the environment), and messages (to and from other agents). For each of these concepts, several parameters need to be designed including the information carried by the percept/message, the effect of the action, and what to do

in case of failure.

Descriptions of data usage are also required. Plans need to specify whether they are reading and/or writing data. Figure 4 shows two plans (p_Startup and p_UpdateEnvironment) that use three data sources (Links, Nodes, and Attractions). p_Startup has write-only access, whereas p_UpdateEnvironment has both read and write access to the data sources.

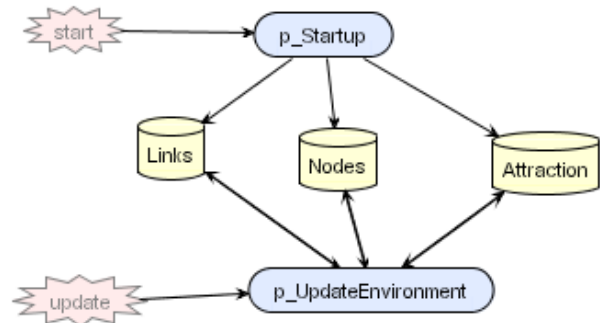


Figure 4: A diagram showing the data usage.

In this stage, plans are described at a high level, including their name, the percepts that trigger them (as shown in Figure 4 by the star shapes) and the actions that occur during the plan. They will be designed in more detail depending on the implementation platform.

The main issue with using an agent-oriented methodology is that it designs the simulation only i.e., what the agents are doing. It cannot design the core of the simulation i.e., how the clock will tick over, the graphical user interfaces required to set up the simulation, the methods to collect outputs. Therefore Prometheus needs to be combined with another methodology to design the whole of the simulation.

Implementation

We constructed a prototype model in JACK Intelligent Agents which involved agents entering a sports precinct and moving towards a stadium. Several "distractions" were located on the way to the stadium, such as food stands and street performers. In our prototype, we attempted to implement the entire model architecture (user-vehicle-environment) in JACK to avoid complex interfacing.

JACK is based on the BDI architecture and was purpose-built for simulations, in particular defence simulations. The aim of the package was to develop a stable, lightweight and practical agent-based programming language that would not be superseded quickly and would facilitate further research. It is based on Java with a few syntactic extensions, and when compiled compiles to Java code (Busetta et al. 1999). The product has been used for several applications, mainly within de-

fence, however it has a strong reputation worldwide both in research and industry (Agent Oriented Software 2005).

JACK supports the concepts in the BDI architecture and Prometheus: agents, events, beliefs (data), capabilities, and plans. Appendix A shows a plan (p_Stopping.plan) written in JACK code. As the JACK files are compiled into Java before execution, normal Java statements can be embedded in JACK files.

It is straightforward to implement goal-directed behaviour, such as moving towards the stadium. The belief system, however, is similar to facts in a logic programming language such as Prolog and does not handle complex beliefs well. For example, it is difficult to represent an environment in detail using JACK belief-sets. Ideally an interface to the environment should be developed and then any environment format (eg. graph, cells, shapes) could be used behind that.

The decision-making used in BDI and in JACK cannot elegantly handle continuous events, such as stepping. It is also difficult to define the subconscious decisions behind walking. Therefore the vehicle model in our architecture would be better suited to an object representation rather than an agent one. JACK has the ability to interface with both Java and C++ code, however we are still experimenting with this interface.

An advantage of agent-based technology is that agents are capable of doing several things concurrently without trouble eg. walking and looking about. So, if you are walking to the post office and you see a shop that has a sale, you continue walking as you make a decision whether to detour or not. If you decide to keep going, you continue with your existing walking plan. However, if you decide to detour, you stop your current walking plan and construct another to get to the sale. We found JACK had problems interrupting one plan before starting another and in our model, agents sometimes found themselves in two places at once. A solution is to increase the lookahead of the pedestrian agent, so that they make decisions and construct new plans earlier.

CONCLUSION

In this paper, we explored the need for pedestrian modelling, the nature of pedestrian behaviour and techniques for modelling it. In particular, we investigated using the belief-desire-intention (BDI) architecture to model pedestrian behaviour using the design methodology Prometheus and the agent-oriented language JACK Intelligent Agents.

We found that the user-vehicle-environment architecture is an appropriate separation for transport models, and applies to pedestrian models even though the user and vehicle are physically the same. The BDI architecture is appropriate for the user model only, as that is

where the decisions are made. Prometheus is useful for designing the BDI concepts required. For the vehicle and environment model, an object approach is more suitable than using an agent language such as JACK.

The work is continuing as part of a larger project to evaluate approaches and methodologies for modelling pedestrian behaviour. The next stage is to learn from our prototype and use JACK Intelligent Agents to develop a model of the user, following the Prometheus specification, and connecting it to environment and vehicle modules written in Java.

ACKNOWLEDGEMENTS

The first author thanks Dr. Michael Kirley and Julien Fischer for their assistance.

APPENDIX A: SAMPLE JACK CODE

```
package pedestrian;
import data.*;

public plan p_Stopping extends Plan {
    #handles event m_Stopping att;

    #posts event m_Replan replan;

    #uses data b_Personal personal;
    #uses data b_Attraction attraction;

    #uses interface a_Pedestrian self;

    body() {
        logical int node, id, m, n, type,
            open, close, distract, l,
            s, swing;
        logical String name;

        attraction.get(id, att.where, n,
            type, name, open, close,
            swing);

        personal.add("move",0);
        personal.add("seenAttraction",0);
        System.out.println(self.name()
            + ": " + "stopping");
        personal.get("currentNode", n);
        personal.get("currentLink", l);
        personal.get("currSpot", s);
        @subtask(replan.post(n.as_int(),
            l.as_int(),s.as_int(),
            att.where));
        System.out.println(self.name()
            + ": " + "stopping now");
    }
}
```

REFERENCES

- Agent Oriented Software. 2005. "Agent Oriented Software (AOS) - JACK Intelligent Agents, software agent system." <http://www.agentoriented.com/shared/home/index.html>, accessed 4 April 2005.
- Australian Pedestrian Council. 2004. "The Pedestrian Council: Pedestrian Council Home". <http://www.walk.com.au>, accessed 9 December 2003.
- Bergenti, F.; M.-P. Gleizes; and F. Zambonelli. 2004. *Methodologies and Software Engineering for Agent Systems*. Springer.
- Busetta, P.; R. Rönnquist; A. Hodgson; and A. Lucas. 1999. "Light-Weight Intelligent Software Agents in Simulation". Technical Report 3, Agent Oriented Software.
- Dijkstra, J.; A. Jessurun; and H. Timmermans. 2001. "A multi-agent cellular automata model of pedestrian movement". In *Pedestrian and Evacuation Dynamics*, M. Schreckenberg and S. Sharma, (Eds.), Springer-Verlag, Berlin, 173–181.
- Gloor, C.; P. Stucki; and K. Nagel. 2004. "Hybrid techniques for pedestrian simulation". In *Cellular Automata: 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004*, P. Soot; B. Chopard; and A. Hoekstra, (Eds.), volume 3305 of *Lecture Notes in Computer Science*, Springer-Verlag, 581–590.
- Harney, D. 2002. "Pedestrian modelling: current methods and future directions". *Road & Transport Research* 11, no. 4, 2–12.
- Helbing, D.; P. Molnár; I. J. Farkas; and K. Bolay. 2001. "Self organizing pedestrian movement". *Environment and Planning B: Planning and Design* 28, 361–383.
- Henein, C. M. and T. White. 2004. "Agent-based modelling of forces in crowds". In *Multi-Agent and Multi-Agent-Based Simulation*, P. Davidsson; L. Gasser; B. Logan; and K. Takadama, (Eds.), volume 3415 of *Lecture Notes in Computer Science*, Springer-Verlag, 173–184.
- Hoogendoorn, S. and P. H. L. Bovy. 2003. "Simulation of pedestrian flows by optimal control and differential games". *Optimal Control Applications and Methods* 24, 153–172.
- Hoogendoorn, S. and P. H. L. Bovy. 2004. "Pedestrian route-choice and activity scheduling theory and models". *Transportation Research B* 38, 169–190.
- Nagel, K. and M. Schreckenberg. 1992. "A cellular automaton model for freeway traffic". *Journal de Physique I* 2, no. 12, 2221–2230.
- Narimatsu, K.; T. Shiraishi; and S. Morishita. 2004. "Acquisition of local neighbor rules in the simulation of pedestrian flow by cellular automata". In *Cellular Automata: 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004*, P. Soot; B. Chopard; and A. Hoekstra, (Eds.), volume 3305 of *Lecture Notes in Computer Science*, Springer-Verlag, 211–219.
- Padgham, L. and M. Winikoff. 2004. *Developing Intelligent Agent Systems - A Practical Guide*. John Wiley and Sons.
- Ronald, N. 2004. "The role of pedestrian modelling in planning for special events". In *Proceedings of the 27th Australasian Transport Research Forum*.
- Rumbaugh, J.; I. Jacobson; and G. Booch. 2004. *The unified modeling language reference manual*. Addison-Wesley, Boston, MA.
- Wooldridge, M. 2000. "Intelligent agents". In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, G. Weiss, (Ed.), MIT Press.

AUTHOR BIOGRAPHIES

NICOLE RONALD graduated with undergraduate degrees in computer science and civil engineering from the University of Melbourne in 2002. She is currently a postgraduate student in the Department of Computer Science and Software Engineering at Melbourne. Between December 2002 and January 2005 she worked as a graduate transport modeller for an global engineering consulting firm in Melbourne. Her webpage is www.cs.mu.oz.au/~naron and her email address is naron@cs.mu.oz.au.

LEON STERLING is Adacel Chair of Software Innovation and Engineering at the University of Melbourne. He is a co-director of the Intelligent Agent Lab (www.cs.mu.oz.au/agentlab) and his research interests include agent-oriented software engineering. His email address is leon@cs.mu.oz.au.