Proceedings of the Seventeenth University of Western Australia School of Computer Science & Software Engineering Research Conference

Peter Dreisiger, Lyndon While (Eds.)

Yanchep, Western Australia

9–10 September 2009

ISBN: 978-1-74052-183-3

Preface

Yanchep 2009 is the seventeenth in a series of research conferences organised and run by the School of Computer Science & Software Engineering at The University of Western Australia. The conference is the flagship in a programme of activities which aim to foster a sense of community and common purpose among the research students and staff of the School. It provides the School's postgraduate and project students with an opportunity to prepare a paper and make a presentation in a full conference atmosphere: the conference deadlines stimulate the completion of research projects, and the proceedings provide a partial snap-shot of the current state-of-research in the School. In its last accreditation of CS&SE, the Australian Computer Society highly commended the School on its research conferences, offering the opinion that UWA leads Australia in this practice.

I would like to thank Dr Paul Nicholson for agreeing to be the keynote speaker at *Yanchep 2009*. Paul was the first ever PhD graduate from the School in 1984, and he has enjoyed a very varied international career since then. Paul will inform and entertain us with stories and lessons from his twenty-five years in the software industry.

The *Yanchep 2009* proceedings were produced almost single-handedly by Peter Dreisiger, whom I thank profusely. The proceedings feature a large number of contributions: fourteen full research papers and three extended abstracts. All students' papers were reviewed by their supervisors, whom I thank for their valuable contribution. In addition, all papers were read and edited by Peter. Authors are strongly encouraged to use the feedback they receive from the conference to review their papers for submission to fully-refereed international journals and conferences.

I would also like to thank Luigi Barone for his contribution to the conference, and Nicola Hallsworth for her administrative support.

Lyndon While
CS&SE Conference Chair
September 2009

Table of Contents

1 Keynote Address

2 Contributed Papers

3 Abstracts

Keynote Address

Tales from the Front Line

Dr Paul Nicholson, BSc(1H) PhD MAICD

1. ABSTRACT

A Computer Science or Software Engineering degree gives you a solid range of technical skills, but does little to prepare you for the realities of complex IT project environments. This address will range over a number of different project scenarios covering some interesting technical, commercial, contractual and political situations. The intention is to provoke thought on the range of skills developed in undergraduate and postgraduate IT students and to highlight technical areas in IT development that are still not adequately solved by the techniques and tools we currently employ.

2. BIOGRAPHY

Paul Nicholson was educated at UWA between 1975 and 1984 and was the first PhD candidate to graduate from the Computer Science Department in 1984. At the completion of his postgraduate studies he moved to Cambridge in England and joined a start up company, Cambridge Control, which was a child of the Cambridge University Information Engineering Department and the Cambridge CAD Centre.

During his career, he has been Senior Software Engineer, Project Team Leader, Project Manager, IT Development Manager, Technical Director, Operations Director, Managing Director and CEO of IT project companies ranging in size from 20 to 300. These companies all trace their lineage back to Cambridge Control. He has travelled through a sequence of company acquisitions, disposals, mergers and transfers, including a transfer back to Australia in 1997.

Paul has been involved in a very wide range of IT development projects over the past 24 years across a broad range of industries. Some projects were relatively small but highly technical, others very large and in some cases highly political. On average most have been large, multi-discipline, real-time and mission-critical. A number have been safety critical, involving the safe carriage of the general public.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

Contributed Papers

Sitting around Talking Gets the Job Done Better

Sabrina Ahmad
School of Computer Science and Software Engineering
The University of Western Australia
35 Stirling Highway
Crawley, WA 6009
Australia

sabrina@csse.uwa.edu.au

ABSTRACT
The process of identifying software requirements is complicated as it involved variety of people and it affects the quality of the end product. This paper is studying negotiation to improve the quality of requirements. The essence of requirements quality in the early stage of requirements engineering which valuing prevention approach is elaborated here. Also, five requirements quality characteristics which are likely to be improved through negotiation are identified. This paper is also reporting on a pilot study of an experiment to show the improvement in requirements quality through negotiation effort.

Keywords
Requirements engineering, negotiation, software quality

1. INTRODUCTION
The quality of requirements is vital to the success of system development in terms of fulfilling the various stakeholders' needs, developing the system in a scheduled time frame and within a budget. Good quality requirements are likely to save the project team from unnecessary cost to fix requirements defects. Most importantly, good quality requirements are leading to a good product. Results from the Standish Group [1] verify this theory.

The Standish Group's CHAOS report that covers the finding from study of 8380 IT projects illustrates that 31.1% of projects are cancelled before they are completed. The results indicate 52.7% of project cost is 189% more than the original estimates, and still deliver fewer features and functionalities than originally specified. Only 16.2% of software projects are completed on time and on budget. The top three reasons for project failure are lack of user input (12.8%), incomplete requirements and specifications (12.3%), and changing requirements and specifications (11.8%). Finally, the major reason for projects cancellation is reported as incomplete requirements (13.1%).

It is seldom technical problems which inhibit productivity and quality [2, 3]. Instead the vast majority of requirements problems are related to human interactions, process and communications. In 1987, Brooks [4] stated that the requirements engineering phase was one of the most important but difficult phases of software engineering. This is supported by empirical evidence [5-7] that proved an inadequately performed requirements engineering process is associated with software system failure. The failure is basically from developing systems that do not meet the customer's needs and expectations. Therefore, in the short run for the development organization this weakness means time consuming activities such as error correction, performance enhancements and adding functionalities. In the long run it means a damaged reputation, lost orders and reduced profits. Thus, the need for good requirements and the consequences of a lack of it are apparent in software systems.

Negotiation is useful to handle contradictory requirements and to resolve disagreement between stakeholders. Contradictory requirements or conflicts are common since project stakeholders frequently pursue mismatching goals. According to Grunbacher[8] negotiation leads to benefits such as understanding project constraints, adapting to change, fostering team learning, surfacing tacit knowledge, managing complexity, dealing with uncertainty and finding better solutions. Furthermore, the benefits of negotiation are obvious and many researchers have pointed out its usefulness for requirements engineering [9-15]. These benefits are believed to improve the stakeholders' agreement level, to save the unnecessary project cost and to improve the requirements quality. This paper focuses on the improvement of requirements quality through negotiation. Hence, this paper is written to discuss the essence of requirements quality in a very early stage of system development where the negotiation is introduced, the measurement of the quality and to discuss the improvement in requirements quality through negotiation revealed in the experiment done recently.

This paper is organized as follows. Following the introduction, section 2 is discussing the essence of requirements quality and the measurement. Section 3 discussed the experiment's aim and assumption, experimental design and the device. This is followed by section 4 which provides the results and the analysis for the experiments. Section 5 is the conclusion.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9-10 September 2009.

2. THE REQUIREMENTS QUALITY

Defining quality is challenging as it is a complex concept, dependent on organizational viewpoints and context characteristics [16, 17]. For example, do fewer errors in line of codes equivalent to a high quality system? What if one of these errors causes the loss of life? Quality has a very different meaning in different situations. In a bank operation, different quality criteria are important than in an electronic control unit of an air plane.

With requirements, this becomes even more difficult, as the notion of quality often depends on the opinions of various stakeholders. If one stakeholder need is not understood correctly, there is a possibility of developing a system considered not in a good quality as it might not support the user in fulfilling certain tasks. In an influential paper examining views of quality, David Garvin [18] suggests that quality can be described from five different perspectives. Table 1 below briefly explains the different perspectives.

Table 1: Quality Perspectives

Perspectives	**Meaning**
Transcendental view	Quality is considered as something that we always strive for as an ideal but we will never be able to implement this ideal
User view	Quality is the fitness of purpose to fulfill user needs.
Manufacturing view	Quality is focuses on the implementation of the processes.
Product view	Quality is the totality of product characteristics.
Value-based view	Quality is the amount of value given to the customer.

Mapping these views on the requirements reveals that different perspectives influence the definition and measurement of requirements quality. However, there is a need to be able to measure the quality to establish baselines, to predict likely quality and to monitor improvement.

The inherently human based nature of requirements engineering and the necessity to consider not only technical but also social aspects when eliciting, negotiating and specifying requirements makes the definition of quality characteristics for requirements even harder. Standards are usually referred to when defining the quality of requirements and requirements specifications. As for this research, IEEE standards [19-21] are referred. On top of that, the quality of requirements here will also referring to the extended version of quality criteria [17, 22, 23] to address value-based view and user based view on requirements quality. Therefore, the quality aspects discussed here consider both technical and human related aspects, which are both relevant for the overall quality of the requirements.

Referring to excerpt of quality assurance approaches in Figure 1 [17], it specifies which elements are important to consider when talking about quality assurance in the requirements engineering phase. Also, it helps to identify which quality characteristics or attributes should be measured in different phases.

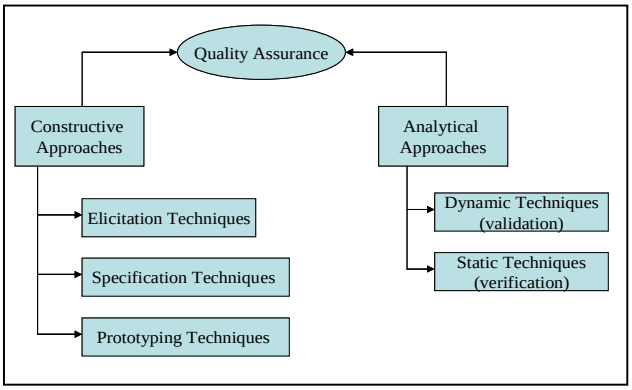


Figure 1: Excerpt of quality assurance approaches

Taken from (Denger et al [17])

The quality assurance approaches are divided into constructive approaches and analytical approaches. Constructive approaches suggest prevention to ensure that mistakes are minimized during the creation of requirements. These approaches are called constructive as they are applied while developing the requirements. The constructive approaches can be done in three techniques which are elicitation techniques, specification techniques and prototyping techniques. Analytical approaches are performed on the completed artifact with the aim to detect defects. This approach is then divided into two, which are dynamic techniques and static techniques. The difference between the two techniques is that the dynamic techniques require executable code such as testing while static techniques can be performed without executable code such as inspection.

This excerpt gives us an important view of distinguishing quality assurance in the requirements analysis phase and the requirements validation phase. In the analysis phase, requirements defects are prevented from being introduced with the help of constructive approaches. While in analytical approaches, the validation process of requirements is based on a requirements document and tries to resolve defects within the documents.

In line with the research objective to ensure quality during the creation of requirements, the quality of requirements discussed here is obviously in requirements analysis phase. In order to derive an appropriate set of requirements from various stakeholders, five characteristics are selected to represent quality at this elicitation stage. They are comprehensibility, consistency, completeness, feasibility and correctness. In this research, unambiguous characteristic is embedded with comprehensibility characteristic as it provides better understanding towards the requirements.

The characteristics of requirements quality may apply to individual requirements or to an aggregate of requirements. Aggregate requirements are a set of requirements for a system or element that specify its characteristics totally. This experiment is concern about aggregate requirements and therefore Table 2 will define quality characteristics in terms of aggregate requirements.

Table 2: Quality attributes for requirements

(Adapted from [17, 19, 20, 23, 24])

Quality Characteristics	Definition
Comprehensibility	The set of requirements is specified and phrased in a way that is understood by both classes of readers; customer and developer.
Consistency	The set of requirements is consistent if no subset of individual requirements in conflict.
Completeness	The set of requirements is complete if all of the functionality that the stakeholders want is stated
Feasibility	The set of requirements is feasible if it is technically achievable and can be implemented within the project constraints; available human resources, schedule and budget.
Correctness	The set of requirements is correct if and only if every requirement stated is the description of what the stakeholders wanted.

Comprehensibility ensures that the requirements are specified and phrased in a way that is understood easily. This is sometimes referred to as clear or as understandable [17, 24, 25]. Natural language or any formalism that includes natural language (e.g., structured English) has much inherent ambiguity. Thus, it is crucial to use less ambiguous expression and to ensure that the same term is used for the same requirement. This is important because **unambiguous** terms used by all the stakeholders help the comprehensibility [23, 24, 26].

Completeness is achieved if all of the functionality that the stakeholders want is stated and all responses of the software in all situations are included. The complete set of requirements is believed to provide sufficient capability to be developed into a system [17, 24, 26].

Consistency is to verify that no subset of individual requirements are in conflict [24, 26]. This research is looking into internal consistency to ensure that there is no subset of individual requirements stated conflict within a set of requirements. External consistency is usually looking at conflicts with any other project documentation which is outside the scope of this research. Therefore, consistency here is referring to internal consistency only.

Feasibility is important to ensure that the set of requirements are technically achievable and can be implemented within project constraints. [24, 26, 27].

Correctness is essential to ensure that the requirements reflect what the users or customers wanted. Hence, the set of requirements is correct if every requirement stated is the description of what the stakeholders wanted.

In the early stage of requirements engineering, valuing prevention approach, and based on the essence of the quality of requirements, the quality of requirement is defined as below:

"The degree to which the set of high level functional requirements is comprehensible, complete, consistent, feasible and correct."

2.1 The Measurement

Negotiation process is believed to improve the quality of requirements. Five quality characteristics of requirements are likely to be improved at the very early stage of requirements elicitation whenever negotiation is applied. They are comprehensibility, consistency, completeness, feasibility and correctness. The absence of these quality will caused defects of incomprehensible, inconsistent, incomplete, infeasible and incorrect respectively. These five quality characteristics are responding to what the stakeholders' wanted, stakeholders' understanding, project constraints and technicality as summarized in Table 3.

Table 3: Quality Characteristics Responsibility

Quality Characteristics	1	2	3	4
Comprehensibility		✓		
Consistency				✓
Completeness	✓			✓
Feasibility			✓	✓
Correctness	✓			

Legend:
1 – Stakeholders' wanted
2 – Stakeholders' understanding
3 – Project constraints
4 – Technicality

2.1.1 Comprehensibility

Comprehensibility is the degree to which the set of requirements is specified and phrased in a way that is understood by both classes of readers; customer and developer.

Comprehensibility can be discovered by inspecting the requirements list after the negotiation and identifying incomprehensible requirements statements. Initially, incomprehensible requirements statements are seeded among a number of comprehensible requirements given to the participants.

The measurement for comprehensibility is:

$$Q_{Comprehensibility} = \frac{n_{cm}}{n_h}$$

Where n_{cm} is the number of comprehensible requirements and n_h is the number of agree-to-have requirements. This ranges 0 (100% incomprehensible) to 1(100% comprehensible).

2.1.2 Consistency

The set of requirements is consistent if there is no subset of individual requirements in conflict. Davis et al. [24] stated that measuring internal consistency is easier if we think of the set of requirements defining a function that maps inputs and states into outputs and states. A consistent set of requirements is one that can be described as a deterministic flow of event. Any nondeterministic flow is inconsistent which, implies the set of requirements define two different system responses or a next state in identical situations.

In order to identify consistency, a review by inspecting the set of requirements to identify inconsistent requirements will be done. There might be two or more inconsistent requirements at one time and one or more requirement will be dropped to make a consistent set of requirements. Thus, the maximum number of requirements that can be consistent will be identified.

A measure of overall consistency is the percentage of maximum requirements that is consistent:

$$Q_{Consistency} = \frac{n_{ct}}{n_h}$$

Where n_h is the total number of agree-to-have requirements and n_{ct} is the maximum number of requirements that can be consistent. Values range from 0 (100% internally inconsistent) to 1 (100% internally consistent).

2.1.3 Completeness

The set of requirements is complete if all of the functionality that the stakeholders want is stated.

In order to claim completeness, a traceability from the set of agreed requirements to the set of what every stakeholder stated they wanted after the negotiation process will be done.

A measure of completeness is:

$$Q_{Completeness} = \frac{n_h}{n_w}$$

Where n_h is the number of agree-to-have requirements and n_w is the total number of requirements every stakeholder's representative stated they wanted, representing the voices of their group. Values ranges from 0 (100% incomplete) to 1(100% complete).

2.1.4 Feasibility

The set of requirements is feasible if it is technically achievable and can be implemented within the project constraints; available human resources, schedule and budget. There are three distinguished feasibility aspects here.

1. Resource feasible – the subset of requirements can be built within time and cost constraints.
2. Dependencies feasible – All requirements in the subset have all dependencies included in the subset.
3. Technically feasible – For the purpose of this research, all the requirements have been checked and assumed to be technically feasible.

However, the measurement of feasibility we will use here is based on 1 and 2 as mentioned above. They are resource feasible and dependencies feasible.

1. Resource feasible will be discovered by calculating the total effort needed to be spent to develop the set of requirements against available resources. Based on the constraints given and the effort needed to develop every requirement, the total effort will be calculated. If the effort needed is less or equivalent to available resources, feasibility is achieved.
2. Dependencies feasible will be discovered by tracing the requirements (node) and their dependencies (route) in the dependencies diagram. The absence of any node which is required by others in the subset indicates infeasibility.

These two types of feasibility will be measured here:

1. It is a measure of the existence of a single system and has a value between 0 and 1; a set of requirements are either achievable or given acceptable development resources they are not. However, the distance to the feasibility will be measured here based on the given project constraints. The project constraint is represented by points which stand for human effort to develop the requirements. Effort needed to develop individual requirements is 6 points (difficult), 4 points (moderate) or 2 points (easy). If the total points is equivalent or below the project constraint points ($p_t \leq p_c$), $Q_{FeasibilityA}$ is 1. If the total points are more than project constraint points ($p_t \geq p_c$), $Q_{FeasibilityA}$ shows the distance to the feasibility.

$$Q_{FeasibilityA} = \begin{cases} 1 & p_t \leq p_c \\ \dfrac{p_c}{p_t} & p_t \geq p_c \end{cases}$$

Where p_t is total points and p_c is constraint points. Value ranges from 0 (infeasible) to 1(feasible).

2. It is a measure of the number of requirements which are feasible based on the existence of requirements they depend upon in the subset.

$$Q_{FeasibilityB} = \frac{n_f}{n_h}$$

Where n_f is the number of feasible requirements and n_h is the number of agree-to-have requirements. Value ranges from 0 (infeasible) to 1(feasible).

2.1.5 Correctness

The set of requirements is correct if every requirement stated is the description of what the stakeholders wanted.

Correctness is identified through traceability from the set of agreed requirements to what the customer or user wants; it is checked if the requirements stated correctly and reflects what they wanted. This is declared by every customer or user individually after the negotiation session.

Since the term correctness applies to an individual requirement and to an entire set of requirements, correctness is measuring the percentage of individual correct requirements.

$$Q_{Correctness} = \frac{n_c}{n_h}$$

Where n_c is the number of correct requirements and n_h is the number of agree-to-have requirements. Values range from 0 (totally incorrect) to 1(totally correct).

3. THE EXPERIMENT

In this study we conducted a role play experiment in which the stakeholders need to negotiate among themselves in order to identify the right requirements to be developed. A system which is familiar to the subject who plays the role of the stakeholders is important. It will reduce the pressure on understanding the system environment, the functionalities and the constraints. Thus, the system to be used in the experiment is Unit Registration System for students at The University of Western Australia. This is a system to enable students to register their choice of courses units. The stakeholders for the system are the representative of students, lecturers, administrators and the university finance staff. The experiment is designed to discover the improvement in requirement quality through negotiation.

3.1 Aims and Assumptions

3.1.1 Aims

Negotiation employed in this experiment is a consensus based negotiation. Consensus based strategy stresses the cooperative development of decision making with group member working together rather than bargaining against each other [10, 28-30]. Therefore, the negotiation process in the experiment is moving towards consensus by achieving a group decision on a list of requirements to be developed. However, consensus might not be achieved due to the time constraints.

3.1.2 Assumption

At this point, assumption for the research is significant to the experiments. The qualitative factors which are believed to influence the negotiation results are not applied in the experiments. Therefore, all the stakeholders involved in the experiment are assumed to have the same cultural background and the same level of knowledge, maturity and experience. To support this, efforts have been put to educate the stakeholders through formal lectures, handouts and guided briefing in advance. Also, explanation and examples were given equally to all of them.

A consensus based negotiation is deployed with an assumption that all the stakeholders have positive motivation and working together towards developing a good system. This assumption will be tested by questionnaire right after the experiment during the post mortem session. The questionnaire is designed to test the assumption and to gather feedback from the participants in order to know how far the exercise meets the objectives. However, the assumption is not tested during this pilot study. It will be available during the actual experiment which is not done yet.

3.2 Experimental Device and Protocol

Groups of five students will exercise a negotiation in requirements engineering process. The objective of this experiment is to measure the quality of the consensus requirements achieved through negotiation by its comprehensibility, consistency, completeness, feasibility and correctness.

3.2.1 The Device

The device for the experiment is a case study named Academic Unit Registration System, a list of forty requirements elicited from the case study and groups of computer science students. The students are third year, fourth year and master computer science students with software engineering knowledge background. Particularly, they are equipped with the negotiation theory and concept through formal lecture before the exercise.

3.2.2 The Protocol

The experiment consists of two stages to observe the achievement through negotiation and to distinguish the progress whenever additional time is given to negotiate.

In order to ensure the existence of negotiation, a project constraint is inserted into the exercise. As an assumption, each group has 60 points which represents $60,000 and 60 days. The total effort needed to fulfill all the requirements are 120 points. Therefore, the students' groups have to drop some of the requirements and identify the most desired requirements worth 60 points. Furthermore, requirements difficulty level is introduced here to show that in real situation, different amount of effort is needed for different requirements. Complicated requirements need more effort compared to a simple one. The forty requirements are tagged as difficult, moderate and easy. Easy requirements need 2 points, moderate requirements need 4 points and difficult requirements need 6 points.

Every stakeholder in a group will own 10 requirements (exception for the team leader) and it is tagged clearly as 'S' for students, 'L' for lecturer, 'A' for administrator and 'F' for finance staff. Time is given for the participants to read the case study and to understand their requirements.

Then, 20 minutes is given to perform the negotiation in order to achieve an agreement on which requirements to have and which requirements not to have. During the negotiation process and whenever agreement is achieved, the team leader has to record 1 (agree-to-have) or 0 (agree-not-to-have) or u (undecided)

Next, a second chance is given to renegotiate and another 20 minutes is given. This is the opportunity for the groups to consider more views from the stakeholders, to explore the requirements rigorously and to carefully make a better group decision. Again, during the negotiation process and whenever agreement is achieved, the team leader has to record 1 (agree-to-have) or 0 (agree-not-to-have) or u (undecided) in the consensus sheet.

After the negotiation is over, an individual stakeholder has an opportunity to record their own say. In the individual sheet, every stakeholder can identify the requirements which they really think they wanted.

3.3 Threats to the Experiment Validity

Whenever students are used as the subject for an experiment, a typical question will be asked if the experiment results are valid or not if compared to the real environment. Students are one of the most accessible sources of small scale project data. It has been shown that data gathered from students is generally applicable to the software industry. Host [31] observed no significant differences between students and professionals for small tasks of judgment. According to Tichy [32], using students as subjects is acceptable if students are appropriately trained and the data is used to establish a trend. These requirements are both fulfilled in this case.

A role play experiment always come with dilemma if the participants are really playing their role or incorporates their personal judgment. In order to minimize that possibility, prior to the experiment, the participants were given a formal lecture on negotiation with knowledge on the nature of a role play experiment and given ample time to explore their roles and their dedicated requirements. Observation done by the researcher throughout the experiment discovered that most of the time, the participants were playing the role given to them.

4. RESULTS AND DISCUSSION

4.1 Observation

The experiment runs smoothly and results were collected from all the groups when the time given ended.

It was observed that most of the time was spent explaining to other participants the reason why one choose to have or not to have each requirement. This effort helps elaborating tacit knowledge and therefore helps to understand the role of the requirements functionality clearly. It also reveals the benefit of the requirements to the system generally and the importance of it to the specific stakeholders. There was 'give and take' approach during the session which makes some of them agree to drop their preferred requirements if they will gain others. Besides, some of them were struggling to urge other participants to agree with their decisions. In summary, even though there was no negotiation technique introduced in the experiment, but through the experiment design, the negotiations exist and not simply free conversation happen among the participants in a group during the experiment.

4.2 Analysis

The main objective for the experiment is to measure the requirements quality whenever negotiation is applied. This experiment also showed the increment in the number of the agreed requirements out of the negotiation process.

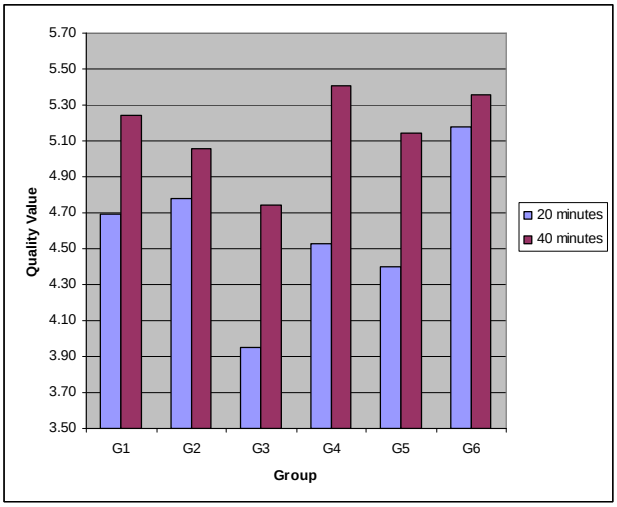


Figure 2: Quality value achieved by six groups out of a negotiation process for 40 requirements

Figure 2 shows the analysis of requirements quality gathered from a negotiation process. There are six groups with five participants each performed a negotiation for a total of 40 minutes. All the six groups need to achieve a consensus for every single requirement proposed by the stakeholders. Hence, if consensus is not achieved, the requirements will be tagged as undecided.

The quality value shown in Figure 2 is the total value of requirements quality achieved by each group within the time permitted which represents comprehensibility, consistency, completeness, feasibility and correctness as discussed in section 2.1. The blue bar represents 20 minutes negotiation effort and the red bar represents 40 minutes negotiation effort. The result suggests that more negotiation effort improves the requirements quality.

Figure 3 show the number of agreed requirements achieved by six groups out of a negotiation process. The blue bar represents 20 minutes negotiation effort and the red bar represents 40 minutes negotiation effort. There are 40 requirements altogether with 120 points effort in total to develop all the requirements. A 60 points resource constraint is introduced and need to be considered during the negotiation. This is to suggest that the decision made is not only based on the stakeholders' desired but also based on the resource constraints. The agreed requirements mean a consensus is achieved by the group for each requirement to be listed or to be dropped. Due to the time constraint, there are requirements remained undecided. Considering the resource constraint, there is a possibility that the number of agreed requirements decrease and this is shown by G4 and G6 in Figure 3. In general, the results suggest that more negotiation effort will increased the number of agreed requirements. This is also suggesting that the level of agreement among various stakeholders improved.

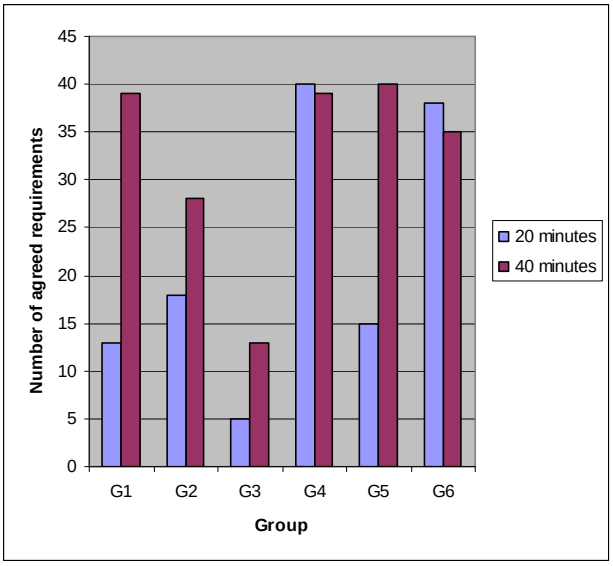


Figure 3: Number of agreed requirements achieved by six groups out of a negotiation process for 40 requirements

5. CONCLUSION

A set of complete and reliable requirements is vital to the software project success. The conflicts which lead to defects and the tacit knowledge which is hidden in an individual stakeholder may cause error in the requirements. The experiment done suggests that negotiation improved the quality of requirements and improved the number of agreed requirements among various stakeholders.

This pilot study will be used to strengthen the actual experiment which will be held in a near future. The next experiment is considering three stages experiment which consists of no negotiation stage, negotiation stage and more negotiation stage. This is to observe the improvement in the requirements quality in different stages especially the difference between the absence and the existence of negotiation. Another experiment is planned to have a different number of participants in a group. This is to observe if the amount of negotiation effort can be substitute by the number of participants negotiating. All the experiments are seen as empirical evidence to the theory which suggests that negotiation is beneficial to the software project.

6. ACKNOWLEDGMENTS

Thank you to both my supervisors, Professor Mark Reynolds and Associate Professor Terry Woodings for their endless support, stimulating suggestions and encouragement throughout my research journey.

7. REFERENCES

[1] "Standish Group's CHAOS Report", in http://www.standishgroup.com/: Standish Group, 2004.

[2] A. Al-Rawas and S. Easterbrook, "Communication Problem in Requirements Engineering: A Field Study Analyzing Requirements Engineering Problems", presented at First Westminster Conference on Professional Awareness in Software Engineering, Royal Society, London, 1996.

[3] D. E. H. Damian, "Challenges in Requirements Engineering", The University of Calgary, Calgary 4 January 2000.

[4] F. D. Brooks, *The Mythical Man-Month: Essays on Software Engineering*. U.S: Addison-Wesley, 1995.

[5] D. Damian and J. Chisan, "An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes that Lead to Payoffs in Productivity, Quality, and Risk Management", *Transactions on Software Engineering*, vol. 32, pp. 433-453, 2006.

[6] V. R. Basili and B. T. Perricone, "Software errors and complexity: An empirical investigation", *Communication ACM*, vol. 27, pp. 42-52 1984.

[7] K. El Emam and N. H. Madhavji, "A field study of requirements engineering practices in information systems development", presented at Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on, 1995.

[8] P. Grunbacher and N. Syeff, "Requirements Negotiation", in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Eds. Berlin: Springer-Verlag, 2005, pp. 143-158.

[9] W. Al-Karaghouli, S. AlShawi, and G. Fitzgerald, "Negotiating and Understanding Information Systems Requirements: The Use of Set Diagrams", *Requirements Engineering*, vol. 5, pp. 93-102, 2000.

[10] B. Boehm and A. Egyed, "Software Requirements Negotiation: Some Lessons Learned", presented at 20th International Conference on Software Engineering, Kyoto, Japan, 1998.

[11] D. E. H. Damian and D. Zowghi, "An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations", presented at 36th Annual Hawaii on International Conference, Big Island, Hawaii, 2003.

[12] A. M. Davis, *Just Enough Requirements Management*. New York: Dorset House, 2005.

[13] P. Grunbacher and R. O. Briggs, "Surfacing Tacit Knowledge in Requirements Negotiation: Experiences using EasyWinWin", presented at 34th Hawaii International Conference on System Science, Hawaii, 2001.

[14] K. Mohan and B. Ramesh, "Traceability-based knowledge integration in group decision and negotiation activities", *Decision Support Systems*, vol. 43, pp. 968-989, 2007.

[15] B. Nuseibeh and S. Easterbrook, "Requirements engineering: A Roadmap", in *Conference on The Future of Software Engineering* Limerick, Ireland ACM Press, 2000, pp. 35 - 46

[16] B. Kitchenham and S. L. Pfleeger, "Software quality: the elusive target [special issues section]", *Software, IEEE*, vol. 13, pp. 12-21, 1996.

[17] C. Denger and T. Olsson, "Quality Assurance in Requirements Engineering", in *Engineering and*

Managing Software Requirements, A. Aurum and C. Wohlin, Eds. Berlin: Springer-Verlag, 2005.

[18] D. Garvin, "What does "Product Quality" Really Mean?", in *MIT Sloan Management Review*, 1984, pp. 25-45.

[19] "IEEE Recommended Practice for Software Requirements Specifications. IEEE Standard 830-1998", IEEE Computer Society, 1998.

[20] "IEEE Guide for Developing System Requirements Specification. IEEE Standard 1233-1998", IEEE Computer Society, 1998.

[21] "Software Engineering - Product Quality Part 3: Internal Metrics", Standards Australia, 2007.

[22] C. Denger, V. Kerkow, A. Knethen, and B. Paech, "A comprehensive approach for creating high-quality requirements and specifications in automotive projects", presented at International Conference of Software and Systems Engineering and their Applications, Paris, France, 2003.

[23] N. Yilmazturk, ""Good Quality" Requirements in Unified Process", in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Eds. Berlin: Springer-Verlag, 2005, pp. 373-401.

[24] A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, P. Sitaram, A. Ta, and M. Theofano, "Identifying and Measuring Quality in a Software Requirements Specification", presented at Proceedings of 1st International Software Metrics Symposium, Baltimore, Maryland, United States, 1993.

[25] L. Rosenberg, L. Hyatt, T. Hammer, L. Huffman, and W. Wilson, "Testing metrics for requirements quality", presented at 2nd International Software Quality Week, Brussels, Belgium, 1998.

[26] P. Kar and M. Bailey, "Characteristics of good requirements", presented at Proceedings of 6th international Symposium of the NCOSE, Boston, Massachussetts, USA, 1996.

[27] I. Hooks, "Writing Good Requirements", presented at Proceedings of the Third International Symposium of the NCOSE, Arlington, Virginia, United States, 1993.

[28] J. Price and J. Cybulski, "Consensus Making in Requirements Negotiation: The Communication Perspective", *Australasian Journal of Information Systems*, vol. 13, pp. 209-224, 2005.

[29] H. In and S. Roy, "Visualization issues for software requirements negotiation", presented at Computer Software and Applications Conference, 2001. COMPSAC 2001. 25th Annual International, 2001.

[30] B. Boehm, P. Bose, E. Horowitz, and L. Ming-June, "Software requirements as negotiated win conditions", presented at Requirements Engineering, 1994., Proceedings of the First International Conference on, 1994.

[31] M. Host, B. Regnell, and C. Wohlin, "Using Students as Subjects—A Comparative Study ofStudents and Professionals in Lead-Time Impact Assessment", *Empirical Software Engineering*, vol. 5, pp. 201 - 214 2000.

[32] W. Tichy, "Hints for reviewing empirical work in software engineering", *Empirical Software Engineering*, vol. 5, pp. 309-312, 2001.

[33] J.-S. T. Aahad M. Osman-Gani, "Influence of culture on negotiation styles of Asian managers: An empirical study of major cultural/ethnic groups in Singapore", *Thunderbird International Business Review*, vol. 44, pp. 819-839, 2002.

Toxicity Evidence Integration

Alison Anderson
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: alison@csse.uwa.edu.au

ABSTRACT

The environment is increasingly contaminated with thousands of chemical compounds the health effects of which are not well understood. The ways in which a contaminant can perturb natural biological processes are complex and evidence of toxicity arising from any one particular experiment or study is rarely conclusive. A weight-of-evidence approach that evaluates evidence from multiple studies and information repositories is required to assess public health risk. Two fundamental issues hampering this process are the heterogeneity of data sources and the fact that much of the evidence available, while relevant and informative to some degree, will not be specific to the health endpoint under investigation. This project investigates how a structured metadata standard, the Resource Description Framework (RDF), can be applied to facilitate the aggregation of information from disparate sources and to enhance the assessment process. The aggregation component has been implemented using Testicular dysgenesis syndrome as a health endpoint. A scoring system for the assessment process is currently under development.

Keywords: Resource Description Framework, toxicity, gene network, Weight-Of-Evidence.

CR Classifications: D.2.12 [Interoperability]: Data mapping

1. INTRODUCTION

Assessing potential risk from exposure to environmental contaminants is particularly challenging due to the complexity and range of influencing factors. To mention just a few: individuals are exposed to multiple chemicals from conception onwards making it difficult to tease out which chemical(s) may have contributed to an adverse health outcome; chemicals may interact with each other giving rise to mixture effects and an innocuous substance may become toxic only after being metabolised by the body or having reacted with another contaminant in a mixture; the timing of exposure is critical and different exposure routes, such as inhalation or absorption can give rise to different health outcomes; and the risk to any one individual will depend on their susceptibility.

Evidence of potential toxicity is derived from various science domains using a range of methods. Epidemiologists study populations while laboratory scientists use in vivo (live animals), and in vitro (cell culture) models. Epidemiology, which compares health endpoints in exposed individuals with those who have not been exposed, can identify associations between a contaminant and an adverse effect but provides no information about the underlying mechanism. Conversely laboratory based studies provide insight into the mode of action but it is difficult to extrapolate these findings to the human condition. Evidence that any particular contaminant is responsible for a specific health endpoint is rarely without doubt and consequently, a Weight-Of-Evidence (WOE) approach to risk assessment is required.

WOE approaches generally involve groups of experts assessing a range of evidence and drawing on their knowledge and experience to pass judgement. The process is highly subjective and often lacks transparency [6]. Guidelines, such as those for Carcinogen Risk Assessment [5] available online from the US Environmental Protection Authority help standardise the approach taken by assessors and ensure that all appropriate evidence is taken into consideration. In some cases they may also provide scoring and ranking strategies for quantifying evidence components but for the most part the process is qualitative. To date no canonical weight-of-evidence frameworks have emerged in the health sciences and it is desirable that frameworks be developed to minimise subjective judgement and maximise opportunity for quantitative assessment [6]. The Toxicity Evidence Integration project explores how a metadata standard can be applied to firstly integrate toxicology and genomic information relevant to a specific health endpoint and secondly how the information can be evaluated for toxicity risk. Essentially the project aims to:

- Derive evidence using information that is available in public repositories.
- Use the Resource Description Framework (RDF) to facilitate:
  - data integration;
  - data mining.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

- Focus on toxicity at a molecular level (gene /protein chemical interactions).
- Develop a scoring system to determine:
  - relevance of toxicology evidence to the health endpoint of interest;
  - potential for toxicity from the chemicals of interest.

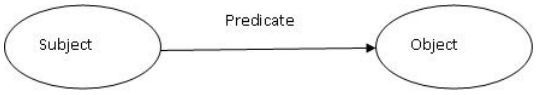
2. DATA SOURCES

Testicular dysgenesis syndrome is a health outcome that is hypothesised to result from foetal exposure to environmental contaminants [8] and as such has been selected as a case study topic to demonstrate the strategies used by this project. The specific biology of interest for this health endpoint includes germ cell development and sexual differentiation in the foetus. Accordingly genetic pathways relevant to this biological space were identified through literature searches and downloaded from publicly available resources such as the Pathway Interaction Database which is a curated collection of bio-molecular interactions created and maintained by the National Cancer Institute and the Nature Publishing Group [10]. Additional genomic information was sourced from The Genitourinary Development Molecular Anatomy Project (GUDMAP) which catalogues gene expression during foetal development in the mouse in order to better understand process and origins of congenital defects and disease in this area [7].

Toxicology information was obtained from the Comparative Toxicology Database (CTD) [9]. This resource contains information on gene/chemical interactions that have been manually curated from literature in peer -reviewed journals. Further toxicology information was downloaded from the U.S. Environmental Protection Agency (EPA) Distributed Structure-Searchable Toxicity (DSSTox) Public Database Network [4]. This resource provides toxicity data from laboratory experiments, mostly using animal models. This includes, for example, data on the ability of a suspected endocrine disruptor chemical available in the environment to bind to an estrogen receptor in a mouse model.

2.1 Data model

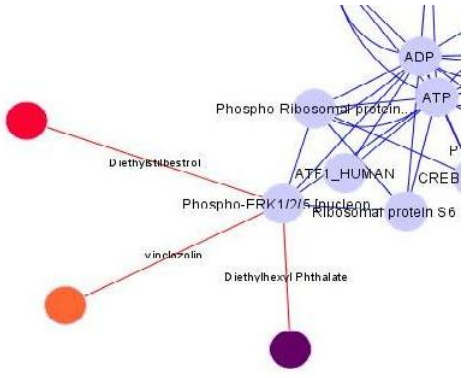
The Resource Description Framework RDF is a general-purpose data model for representing information in the Web that depicts subjects and objects as nodes that are joined together by arcs/edges representing the relationships (predicates) between them.



This simple directed graph structure comprising (subject, predicate, object) or (entity, attribute, value) is known as a *triple* and is the fundamental data structure of RDF. Several serialization formats can be used to markup information and data in accordance with the RDF model, the most popular being the Extensible Markup Language XML (RDF/XML), and Notation 3 (known as N3). The latter is a more compact and readable alternative and is currently being extended to allow greater expressiveness [1]. The gene network data collected for this project was available in RDF format. The toxicity data was converted to RDF triples using a computer program written in Java. All RDF sources were stored in a Virtuoso database. Virtuoso is a cross platform universal server providing web, file and database server functionality and is available in both commercial and open source versions. It supports semantic web functionality by providing native XML storage and was thus an ideal choice for this project.

2.2 Data mining

Storing the toxicity and genomic information in a common format facilitated federated searches. These were carried out using the W3C SPARQL RDF query specification, included in the Virtuoso software. The objective was to extract information from each resource common to specific genes. Thus toxicity information on one or more chemical contaminants could be coupled to each gene in a network generating an overview of potential perturbation for a given network. Alternatively a view of all chemical substances recorded in the literature as having changed the expression profile of one specific gene, or interacted with the protein product of that gene, could be generated. By default, output results from the SPARQL search engine are displayed in a web page table. Alternatively it can be output in triple or RDF format to a text file which can then be manipulated by a general-purpose scripting language such as PHP to display the information in a more user friendly way. Another useful approach is to copy the default table output and paste it into a text editor. This tab delimited data can then be imported into network visualisation software. There are a number of ways this can be done to produce various views. A gene network can be imported with genes as nodes and the interactions between them as edges. Chemicals that interact with a gene can then be added as additional nodes and edges. Additional information can be added as node or edge attributes. In the example below the darker nodes (red, orange and purple) represent chemicals that interact with the ERK protein in the bone morphogenetic protein (BMP) pathway.



At this point it should be noted that not all chemical gene interactions extracted in the data mining process represent potential toxic effects. In some instances the interactions may be an adaptive response only. In many cases the evidence that a gene interacts with a chemical may have been derived from an organism or development time other than that of interest for the case study and will therefore be ir-

relevant. The next stage in the process then is to weed out inappropriate interaction evidence and then use what remains to carry out a weight-of-evidence assessment of potential toxicity.

3. WOE FRAMEWORK

This section discusses the ways in which an RDF metadata framework can enrich a WOE assessment process.

3.1 Data provenance

A WOE assessment must pool evidence from a variety of sources in a range of formats from different research disciplines. It is essential that data provenance is maintained as quality of the source is critical to establishing the reliability and strength of the evidence. The subject and predicate of a triple are normally prefixed with an abbreviation that represents, via a Universal Resource Identifier (URI), the source repository. Consider the following repository mentioned previously in this paper:

The U.S. Environmental Protection Agency (EPA) Distributed Structure-Searchable Toxicity (DSSTox) Public Database Network is available at http://www.epa.gov/NCCT/dsstox/

Each chemical in the DSSTox database has a unique identifier (e.g. RID22308). The subject component of a triple can be created by appending the unique ID to the URL :

http://www.epa.gov/NCCT/dsstox#DSSTox_RID22308

This can be quite verbose and difficult to read if long address and identifiers are involved. To abbreviate it and increase readability the namespace concept is used. At the top of an N3 document the @prefix binds a prefix to a namespace as follows:

@prefix dsster: < http://www.epa.gov/NCCT/dsstox#>.

The triple subject can then be identified as

dsster:DSSTox_RID22308

URIs allows data entities (triples) to be tethered to a database or repository but essentially free of the underlying structure so that they can be extracted without loss of meaning. Pieces of information in triples are thus contextually self-contained in that the link in their subject URI tells us where in the world of information they belong. Subsets of evidence can thus be extracted from disparate resources, converted to triples if not already in RDF format, and combined to facilitate a WOE assessment without loss of provenance.

3.2 Inference capability

Evidence for a WOE assessment is normally a manually intensive process reliant on human expertise to both accumulate and assess the evidence. While the latter will always require human input the former can be automated to some extent. RDF provides machine readable metadata to facilitate federated data mining as demonstrated in this project. Additional metadata triples can be created to refine and enrich annotations and to enhance the data mining process by introducing inference capabilities.

The OWL ontology language and RDF schema vocabulary define predicates with logical semantics. The owl:SameAs predicate, for instance, can be used to state that two things are the same. If we have a resource that uses a formal name for a gene or chemical and another resource that uses a synonym we can write a rule that uses the owl:SameAs predicate to inform a semantic web application that they refer to the same thing. N3, previously discussed as a syntax option, includes semantic capabilities which are defined in the N3Logic specification [2]. N3Logic uses owl:sameAs and adds further predicates to allow rules and built-in functions.

The N3Logic specification also defines Formulae which are sets of statements or triples that can be used in rules. In the RDF data model the object component of a triple can be either a literal or another object identified by a URI. N3Logic extends this model by allowing the object value to be another triple or set of triples. This concept is also referred to as nested graphs. The triples within formulae are enclosed in braces. The following example demonstrate show this can be applied: the USA Environmental Protection Authority have supplemented estrogen receptor binding experimental results with coded categories based on a system that they have devised. Assuming a namespace for the USA EPA has been declared with the abbreviation epa, the logic can be expressed in N3 as:

epa:curator says

{ dsster:DSSTox_RID22308
dsstox:ActivityCategory “weak” } .

3.3 Capturing and accounting for uncertainty

Evidence from experiments always has an element of uncertainty and it is desirable to be able to account for this in a WOE assessment. If the uncertainty can be captured in a structured format a scoring system can then be applied to provide a quantified grading. The W3C Uncertainty Reasoning for the World Wide Web Incubator Group (URW3-XG) has defined an Ontology of Uncertainty as a meta-model to represent the semantics of uncertainty in different contexts. The ontology defines the various ways in which evidence can be derived:

- EXP: Inferred from Experiment
- IDA: Inferred from Direct Assay
- IPI: Inferred from Physical Interaction
- IMP: Inferred from Mutant Phenotype
- IGI: Inferred from Genetic Interaction
- IEP: Inferred from Expression Pattern

A computer programme could be written to automatically classify and annotate evidence data with an evidence code (according to its source) and a relevance score. An assessor would then have the opportunity to extract evidence that is equal to or greater than a specified score.

The ability to make statements about statements, also known as quoting is another useful construct to quantify levels of

uncertainty. In general quoting allows users to distinguish between what they believe to be true and what someone else on the Web states or believes to be true. For example the following claims that the expression in braces is false - that there is no-one identified as Assessor203 who wrote anything titled "Toxic soap".

{ [x:id "Assessor203"] dc:wrote [dc:title "Toxic soap"] } a n3:falsehood .

This can be extended to include a type of uncertainty, a level of uncertainty and a score.

3.4 Enhanced data mining

An assessor can obtain evidence through a literature review of the relevant health endpoint and possibly specific repositories of toxicity information known to them, as was the approach for this project. Automated searches could potentially cast a wider net and rules encapsulated in triples can assist automated evidence gathering. For example triples could be created to define universal quantified formulae. In predicate logic universal quantification is a formalisation of the notion that something is true for everything or every relevant thing. In the case of the Comparative Toxicology Database we can consider that every interaction chemical is a potential toxic entity. The N3 specification defines a log:implies rule in which the subject is the antecedent and the object is the consequent. We can thus define the CTD data entities as a member of a new class in a toxicity ontology (with the prefix eto) as follows:

@forAll X. { X a ctd:actor_type_chem } log:implies { X a eto:potential_toxic_entity }.

In a slightly more complex example, not all chemicals from the estrogen receptor binding resource(DSSTox_NCTRER) are potential toxic entities. Only those who have an ActivityOutcome predicate with a value of *active* may be considered to have potential toxicity. This can be stated in a rule as follows:

@forAll X. { X a dsster:DSSTox_NCTRER ;
dsster:ActivityOutcome active .}
log:implies { X a eto:potential_toxic_entity }.

Current work on the Toxicity Evidence Integration project is exploring how these concepts can be applied to markup information relevant to toxicity investigations.

4. SCORING SYSTEM

The complex nuances surrounding environmental contaminant toxicity necessitate a comprehensive approach to toxicity assessment. To improve the transparency and accuracy of the WOE approach it is essential to capture, cohesively link (to the biology of interest) and appropriately score and weight as many influencing factors as possible. Some of the issues arising and strategies for tackling them are discussed below.

4.1 Relevance

A publication reporting that a chemical has been shown to modify the function of a particular gene provides valuable insight into the toxicity mode of action. However, its inclusion in a WOE assessment must be dependent on the relevance it has to the biological space under investigation. Outcomes can vary across organisms, within species of an organism, and, through natural variability, in the same species. The scoring system under development defines evidence with a higher score to be more relevant and thus more indicative of potential toxicity. Under this paradigm the variance at an organism level could be coded as follows:

Biology organism v Evidence organism	**Score**
Same organism, same species	5
Same organism, different species	4
Different organism but orthologous gene(s) (gene from different species derived from a common ancestor)	3
Different species, no evidence of orthology but similar family (both vertebrates)	2
Unrelated species (human vs plant)	1
Data unavailable	0

Species information is available within the gene/chemical interaction files downloaded from the CTD resource described earlier. Information on gene orthology can also be obtained from public resources. It is therefore plausible that this scoring mechanism could be automated and the scoring implemented using semantic rules.

A second example concerns the issue of time points. A specific gene network may play a role in multiple biological systems and these systems may differ depending on the development stage of an organism (embryo, child, adult). Exposure to a contaminant may therefore cause more or less disruption depending on the time the exposure occurred. This is particularly important during foetal development when a biological system, say sexual differentiation, occurs over a short time frame marked in days and parts of a day. Exposure at this critical time could be profound while exposure some weeks later may have no biologically significant effect. To control for this, the scoring system must weight evidence according to its closeness to the biological space under investigation and might be applied as follows:

Biology time point v Evidence time point	**Score**
Same or very close to time point of interest	10
Close	5
Different	1

How *close* and *very close* are defined will vary according to the biological space and should be determined by the assessor(s). Again this process could be automated, with closeness being defined as parameters that can be used by semantic rules. There will also be some biological systems that, rather than being specific to a time point, are ubiquitous and disruption at any time will be profound. It may be useful to develop a scoring system to rate biological systems from this viewpoint. The scores could be stored in metadata and then applied during WOE assessments.

4.2 Contradictions

A scoring system needs to take into account the fact that studies can have opposing results. If one study shows an increase in expression of a particular gene after exposure to a specific contaminant and another shows a decrease un-

der similar exposre the results should not cancel each other out or be excluded as confusion or noise. As mentioned above the discordance could arise from experiments conducted in different organisms, tissue types or developmental time points. The type of experiment and methods adopted will also be influencing the results. The expression change observed under exposure to the contaminant must therefore be assessed in the context of the experiment and the underlying biology and not purely on the expression level data.

4.3 Missing data

The scoring system will also need to take into account biases that may arise due to lack of information. The majority of chemicals of concern have little or no toxicity data and it is rare that complete human health effect information exists for any one chemical [3]. Chemicals that have been tested will have more infomration available for scoring. Consider for example two chemicals A (tested) and B (not tested). Evidence involving chemical A might gain points because it is listed as being capable of binding to an estrogen receptor. Evidence involving chemical B gains no points as it is not listed, yet may in fact have the same toxicity potential as chemical A. The scoring system must clearly distinguish between *tested* and *not tested*. Semantically, this information might be incorporated into the WOE system by defining all chemicals on the estrogen binding list as belonging to a class of type tested and all others to a class of type not tested at time of assessment.

5. FUTURE WORK

The first stage of the Evidence Toxicity Integration project has focussed on the identification and integration of evidence, the use of semantic technology for data mining and the visualisation of output. The next stage is the development of scoring and ranking methods that will, in the first instance, weed out inappropriate evidence and secondly provide a more quantitative and transparent framework for accurate assessment. As with stage one, every effort will be made to comply with any existing practices and evolving standards to ensure future interoperability with other tools.

A secondary and integral aspect of the evidence integration process is that it assists us to build holistic views of dynamic biological systems and their vulnerability to toxic insult. Each source of evidence contributes unique insight. The ability of a contaminant to bind to a receptor or higher rates of exposures in those with a disease compared to those without are two entirely different evidence modalities but their combination can help elucidate toxicity modes of action. Through this aggregation weak pieces of evidence may find support and contradictory results might be explained. It also envisaged that output from this system can be used in mathematical and computational approaches to understanding toxicity and this work will assist in the identification of biomarkers for toxicity related disease.

6. REFERENCES

[1] Tim Berners-Lee and D Connolly. Notation3 (n3): A readable rdf syntax. *http://www.w3.org/TeamSubmission/n3/*, 2008.

[2] Tim Berners-Lee, D Connolly, L Kagal, and Y Scharf. N3logic: A logical framework for the world wide web. *Theory and Practice of Logic Programming.*, 8:249–269, 2008.

[3] E Demchuk, P Ruiz, J D Wilson, F Scinicariello, H R Pohl, M E Fay, M M Mumtaz, H Hansen, and C T De Rosa. Computational toxicology methods in public health practice. *Toxicology Mechanisms and Methods*, 18:119–135, 2008.

[4] U.S. EPA. Distributed structure-searchable toxicity (dsstox) public database network. *http://www.epa.gov/NCCT/dsstox/*, 2009.

[5] U.S. EPA. Guidelines for carcinogen risk assessment. *http://cfpub.epa.gov/ncea/cfm/*, 2009.

[6] S Krimsky. The weight of scientific evidence in policy and law. *American Journal of Public Health*, 95(s1):129–136, 2005.

[7] M H Little, J Brennan, K Georgas, J A Davies, D R Davidson, R A Baldock, A Beverdam, J F Bertram, B Capel, H S Chiu, D Clements, L Cullen-McEwen, J Fleming, T Gilbert, D Herzlinger, D Houghton, M H Kaufman, E Kleymenova, P A Koopman, A G Lewis, A P McMahon, C L Mendelsohn, E K Mitchell, B A Rumballe, D E Sweeney, M T Valerius, G Yamada, Y Yang, and J Yu. A high-resolution anatomical ontology of the developing murine genitourinary tract. *American Journal of Public Health*, 7(6):680–699, 2007.

[8] O V Martin, T Shialis, J N Lester, M D Scrimshaw, A R Boobis, and N Voulvoulis. Testicular dysgenesis syndrome and the estrogen hypothesis: a quantitative meta-analysis. *Environmental Health Perspectives*, 116:149–157, 2005.

[9] C J Mattingly, M C Rosenstein, G T Colby, J N Forrest Jr, and J L Boyer. The comparative toxicogenomics database (ctd): a resource for comparative toxicological studies. *Journal of experimental zoology*, 305(9):689–692, 2006.

[10] C Schaefer, K Anthony, S Krupa, J Buchoff, M Day, T Hannay, and K Buetow. The nature pathway inteaction database. *Nucleic Acids Research*, 37:674–679, 2009.

Learning Classifier Systems for Fraud Detection: a Preliminary Study

Mohammad Behdad, Tim French and Luigi Barone
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {behdad,tim,luigi}@csse.uwa.edu.au

ABSTRACT

Fraud is a serious problem that costs the worldwide economy trillions of dollars annually. However, fraud detection is difficult as perpetrators actively attempt to masquerade their actions among (typically overwhelming large volumes of) legitimate activity. In this paper, we characterise the main features of fraudulent behaviour and examine the applicability of using a learning classifier system (LCS) for fraud detection. In particular, we explore the behaviour of different LCS variants on data exhibiting features typical of such domains. While it is often assumed in the LCS literature that accuracy-based LCS performs better than strength-based LCS, we show that this may not be the case for fraud detection, and that care must be taken when applying an LCS to this problem.

1. INTRODUCTION

Fraud is the general term used to describe actions undertaken by one party to obtain an unjust advantage over another. Typically, the perpetrator, using false or misleading representations, attempts to disguise their activities in an effort to maximise the effects of their fraudulent behaviour. It was estimated that the cost of personal fraud for Australians in 2007 was $980 million [1]. This research deals with the growing problem of electronic fraud (e-fraud) — fraud perpetrated using electronic technologies; the most common forms including online credit card fraud, telecommunications fraud, email spam, and network intrusion.

Fraud detection is the task of attempting to detect and recognise fraudulent activities as they occur, typically by flagging suspicious behaviour for review and further action by the appropriate authorities. However, fraud detection is difficult to approach using automated methods since most perpetrators actively try to undermine any attempt to classify their activities as fraud. Fraud detection hence necessarily corresponds to an adversarial environment.

Previous work on automated fraud detection has focussed on specific domains such as spam detection [11], network intrusion [16] and phishing [6]. In this paper, we seek to identify some general characteristics of electronic fraud detection and examine the performance of learning classifier systems (LCSs) on abstract problems that share these characteristics. We emulate simple fraud detection problems using random Boolean functions, and compare the effectiveness of automated methods for learning these functions.

In the next section, we examine the abstract properties of fraudulent behaviour, defining characteristic properties we typically expect to observe in such domains. In Section 3 we present background information about LCSs and discuss how they might be used for fraud detection in dynamic environments. In Section 4 we present experiments that explore the performance of different LCS variants on representative problems that captures the defining characteristic properties introduced in Section 2. Finally, Section 5 concludes this work.

2. PROPERTIES OF FRAUDULENT ACTIVITY

Fraud detection is an extremely difficult task. As is evidenced by the ongoing large financial losses of organisations and individuals due to fraudulent behavior [1], no general off-the-shelf solution exists for this problem. The reason for this failure is the fact that fraud detection must deal with some uniquely challenging properties. These properties are described below:

1. **Experience Imbalance**: Perhaps the most defining and important characteristic of fraudulent activity is the imbalance in positive and negative experiences presented during learning; i.e. the proportion of fraudulent records to non-fraudulent records is vastly skewed. This "needle-in-a-haystack" effect typically leads to unacceptably high false positive rates in the data [12]. Indeed, these "rare" instances can have significant negative impacts on performance in data mining problems [18] and hence any fraud detection system must learn to overcome the problems associated with the rarity of positive examples in its learning process.
2. **Large Volumes of Data**: Usually fraud detection must deal with overwhelming large volumes of data, typically mostly negative (non-fraudulent) experiences. Indeed, the sheer volume of data the system must process adds to the overall complexity of the problem. One consequence of this property is that labeling data for supervised training purposes becomes infeasible, and hence finding relevant real-world training data is virtually impossible.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

3. **Online Learning**: Since complete data is not available from the onset (it becomes available gradually over time), we typically have only partial information about the problem at hand. This complicates the learning process as there is a lack of sufficient data for training purposes. Indeed, the detection system may need to approximate or generalise from the limited experiences it has been presented, but must also be able to quickly adapt as new (potentially conflicting) experiences are observed.

4. **Adaptive Adversaries**: Another difficult aspect of fraud detection is the perpetual "arms race" that occurs when dealing with intelligent adaptive adversaries who vary their techniques over time. As the detection techniques mature, fraud perpetrators also become increasingly sophisticated in both their methods of attack and detection evasion. This means that classifiers may be undermined by their own success; overspecialisation is a risk.

5. **Misclassification Costs**: In fraud detection, misclassification costs (false positive and false negative error costs) are unequal, uncertain, can differ from example to example, and can change over time. In some domains, a false negative error is more costly than a false positive (e.g. intrusion detection), whereas in other domains (e.g. spam detection) the situation is reversed [12]. This adds to the complexity of the learning problem. Another consequence of this property is the absolute need for high accuracy.

6. **Concept Drift**: In machine learning, "concept drift" means that the statistical properties of the target variable that the model is trying to predict change over time in unforeseen ways. This makes the predictions less accurate as time passes. Fraud detection suffers from a continuously changing concept definition [15]. Therefore, the approaches used for this purpose should be able to adapt themselves in the presence of drifting concepts over time.

7. **Fast Processing**: In some applications such as network intrusion detection and email spam detection, fraud detection must be done in near real-time. In these cases, the speed of processing is critical, thus not affording the fraud detection software the luxury of extended processing times.

3. LEARNING CLASSIFIER SYSTEMS

Many researchers have explored different machine learning and computational intelligence techniques for specific applications of fraud detection. For example, email spam detection has been examined using naïve Bayes [14], support vector machines [13], visualization [20], artificial immune systems [11], evolutionary algorithms [5], and artificial neural networks [21]. Similar examples exist for other domains like network intrusion detection.

However, precious few examples exist in applying a learning classifier system (LCS) to fraud detection; the most notable being the recent work of Shafi et al. on using an LCS for network intrusion detection [16]. This apparent underutilisation is somewhat surprising, as LCSs have been successfully used in a number of related machine learning and data mining tasks [3] and, due to their online learning capabilities in dynamic environments [2] and straightforward knowledge representation, seem readily applicable to this kind of domain.

3.1 LCS

First proposed by John Holland [8] in 1976, a learning classifier system (LCS) is a machine learning technique that combines reinforcement learning, evolutionary computing, and other heuristics to produce an adaptive system capable of learning in dynamic environments. It uses a population of condition-action-prediction rules called classifiers to encode appropriate actions for different input states; rules are of the If-Then type — if a condition is true, then the corresponding action will be performed. The decision of which rule to use is based on the reward (the prediction in the classifier triple) the system expects to receive from the environment as a result of performing the chosen action.

The three major components of an LCS are rule discovery, credit assignment, and the production system. Rule discovery is the task of creating new rules and is generally performed using a genetic algorithm (GA). The GA operates on the population of classifiers, evolving them through recombination and selection. The fitness function used depends on the type of LCS; a strength-based LCS uses the predicted reward of a rule as its fitness, whereas an accuracy-based LCS uses the inverse of the prediction error as the fitness value of a rule.

As an LCS should adapt to changes over time, it therefore must somehow learn from its environment and improve its rules over time. Credit assignment is the process of updating the reward prediction value of the rules based on the feedback received from the environment after performing an action.

Sometimes when deciding on which rule to use in a particular input state, more than one rule whose condition evaluates to true are found. In this situation, it is the responsibility of the production system to determine which rule to use. The production system usually uses the strength (reward prediction) of the matching rules in its decision making process. However, other mechanisms such as roulette wheel selection also accompany this process to create a balance between exploitation and exploration.

3.2 XCS

For many years, LCS algorithms used the classifier strength parameter (the estimated reward) both as a predictor of reward and as the classifier fitness in the GA. However, in 1995 Stewart Wilson investigated this practice, deciding instead to use two separate parameters for the two different processes [19]. Creating a new LCS variant called XCS (which has now become the most applied and studied LCS over the last several years [9]), the new parameter he introduced was accuracy. Under his scheme, it is not the classifiers which produce the greatest rewards that have the best chance of propagating their genes into subsequent generations, but rather the classifiers which predict the reward more accurately regardless of the magnitude of the reward. As a result, XCS is said to cover the state space more efficiently [17].

When a state is perceived by an LCS, the rule-base is scanned and any rule whose condition matches the current state is added as a member of the current "match set" M. In XCS, once M has been formed, a rule is chosen based on its accuracy. All members of M that propose the same action as the chosen rule then form the "action set" A. The accuracy of all members of the action set are then updated when the reward from the environment is determined.

Another significant difference of XCS over its predecessors is that the GA is applied to A (or niches) rather than the whole classifier population. This narrows the competition to be between classifiers which can be applied in similar scenarios rather than all the classifiers in the LCS [17].

The other important concept in XCS is that of the macro-classifier. During rule discovery, it is possible for classifiers identical to those already in existence to be generated. Instead of keeping several identical classifiers in the population, XCS associates with each classifier a numerosity corresponding to the number of examples in the population, thus allowing the LCS to explore other potential rules while maintaining recognition of the currently dominant classifiers [17].

4. EXPERIMENTS

In this section, we present experiments using abstract test problems that exhibit the characteristic properties of fraudulent activity defined in Section 2. We compare the performance of accuracy-based and strength-based LCSs on three experiments, separately examining the performance of these two LCS variants in the presence of incomplete information, noise, and experience imbalance.

4.1 Experimental Framework

We first define two benchmark problems that can be readily modified to exhibit the different properties of interest. We then vary the defining properties of these problems and report LCS performance on these modified problems below.

Two LCS implementations were developed in C#: an accuracy-based LCS (XCS) and a strength-based LCS (SB-XCS). Our implementations follow the "off-the-shelf" algorithms commonly used in the literature; specific details are provided below. Values for the parameters of these algorithms were set using the recommendations provided by Butz and Wilson [4].

4.1.1 Boolean Multiplexer

The multiplexer problem is the most commonly used benchmark problem in the LCS literature. We use a 6-bit multiplexer in which the input is a 6-bit binary string, the first 2 bits forming the address and the next 4 bits forming the payload. For example, in the string 100010, the 2-bit address (10 in binary) corresponds to the value of the 2nd bit (counting from 0) in the payload — a 1 in this case.

4.1.2 Random Boolean Function

A Boolean function of n Boolean variables, $p_1, p_2, ..., p_n$, is a function of $f : B^n \rightarrow B$ such that $f(p_1, p_2, ..., p_n)$ is a Boolean expression [7]. In this work, we set $n = 6$ and create a random Boolean function by generating randomly a $2^n = 64$ bit binary string called the action string which represents the correct action for every possible state (6-bit binary string). By calculating the decimal value of a 6-bit binary string, the index of the correct action in the action string is found. For example, if the first 10 bits of the action string are 0110010110, then the correct action for state 000101 is the value of the 5th bit (counting from 0) in the action string — a 1 in this example.

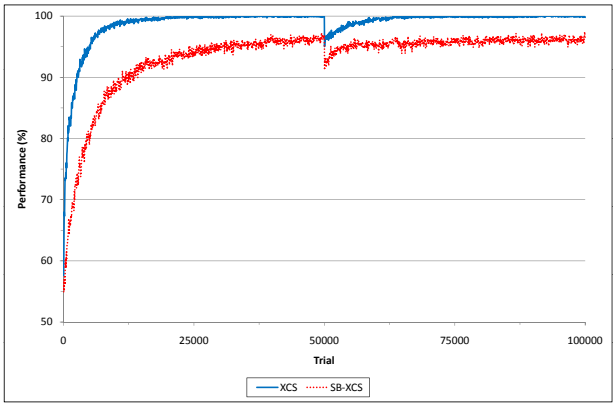


Figure 1: The effect of unseen experiences for the random Boolean function test problem

4.1.3 XCS

Our accuracy-based LCS (XCS) was based on Martin Butz's implementation of XCS [4]. To minimise confounding effects and ensure a fair comparison with the strength-based LCS, tournament selection is used in its discovery component instead of roulette wheel selection.

4.1.4 SB-XCS

Our strength-based XCS (SB-XCS) was based on the strength-based XCS proposed by Tim Kovacs [10]. In order to prevent selection bias towards over-general classifiers, tournament selection was used in the discovery component. Note that both types of XCS subsumption (GA subsumption and action set subsumption) are disabled in strength-based LCSs.

4.2 Experiment 1: Adaptability

Our first experiment compares the performance of the two LCS variants in the presence of incomplete information simulating the requirement for online learning in a fraud detection system. To achieve this, a part of the environment (a percentage of the input states) is kept hidden and only after some predetermined time (the midway point of the experiment) are these hidden states made visible.

In our first set of experiments, we hide 10% of possible states from the learning algorithms; the 10% being representative of some fraudulent activity occurring within the system. The remaining 90% of states are then randomly presented for 50000 trials, after which time the hidden states are added back to the set of states the algorithms are exposed to. A further 50000 trials are then conducted. Fig. 1 reports the performance (as a percentage of the correct predictions) of the two LCS variants on the random Boolean function problem with 10% of states hidden. Experiments were repeated 50 times and the results averaged.

We see from Fig. 1 that while XCS is able to obtain per-

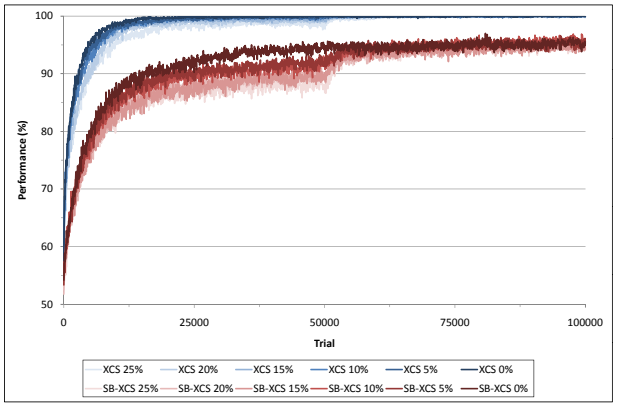


Figure 2: The effect of noise for the random Boolean function test problem

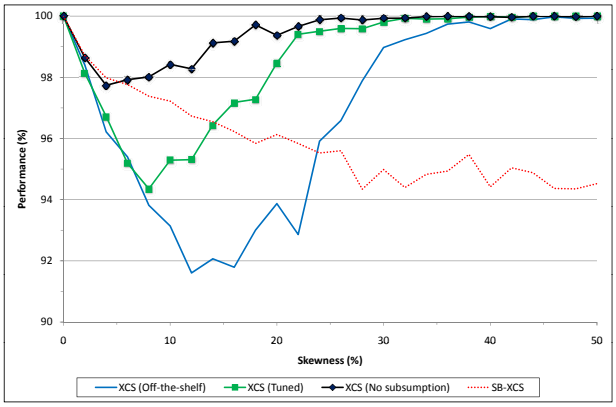


Figure 3: The effect of skewed experiences for the random Boolean function test problem

fect performance before the revelation of the hidden states, SB-XCS can not learn this environment fully, obtaining a maximum performance of around 95%. The learning rate for XCS is also significantly faster than that of SB-XCS; the enhancements of XCS (the use of accuracy as a fitness measure) offering an advantage over SB-XCS in this case. We also see from Fig. 1 that neither LCS is able to learn generalised rules that cover the unseen states and hence, when the hidden states are revealed at the 50000 midway point, both algorithms suffer an immediate performance loss. XCS does seem to handle the addition of the hidden states better than SB-XCS, returning to 100% performance within 13000 additional trials, while SB-XCS takes significantly longer to return to its maximum performance of 95%.

Similar experiments with the multiplexer problem shows that both LCS variants are able to obtain 100% performance well before the revelation point. Hiding 10% of states in this environment has little effect on the performance of either LCS variant — the performance of SB-XCS drops ever so slightly at the midway revelation point but quickly returns to 100% while XCS is seemingly unperturbed by the addition of the hidden states. This result is not overly surprising given the relatively simple nature of the multiplexer problem — both LCS variants learning generalised rules that capture the correct behaviour for these hidden states in a reasonably short period of time.

4.3 Experiment 2: Robustness to Noise

Our second experiment compares the performance of the two LCS variants in the presence of noise. In this experiment, both LCS variants were tested on five levels of noise (5%, 10%, 15%, 20% and 25%), where noise corresponds to the environment erroneously returning the wrong reward for a chosen action. For example, for 5% noise, in 5% of cases the reward is randomly assigned a value between 0 and the maximum payoff (1000).

As with the first experiment, the noise is only present for the first half of the experiment; after the midway point, the effects of the noise are removed and all reward assessments are evaluated correctly. Experiments were repeated 50 times and the results averaged. Experiments were undertaken on both test problems, but for brevity, only the results of the random Boolean function is presented here. Fig. 2 reports the results.

Fig. 2 shows that both LCS variants are reasonably tolerant to noise; the noise seemingly having only a small impact on the performance of both algorithms, although the degradation in performance is more pronounced in SB-XCS than XCS. When the noise is removed at the 50000 midway point, the performance of both algorithms rapidly converges to their maximum performance levels reported in the first experiment. As before, the performance of XCS is superior to that of SB-XCS on this problem.

4.4 Experiment 3: Robustness to Experience Imbalance

Our final experiment compares the performance of the two LCS variants for different levels of experience imbalance. For this experiment, only the random Boolean function test problem is used; the experiment varies the proportion of 1s and 0s in the action string of the random Boolean function, providing a means of varying the imbalance in experiences offered to the LCS. In this context, we define the skewness of the action string as the percentage of 1s in the action string. A balanced (unskewed) environment hence has a skewness of 50%, while a completely skewed environment has a skewness of 0% (or 100%).

In an unskewed environment, XCS obtains a performance of 100% after approximately 13000 trials. SB-XCS however does not obtain perfect performance, instead obtaining a maximum performance of about 95% after 30000 trials with no significant improvement beyond this point. As with the previous experiments, the learning rate of SB-XCS is also slower than XCS in this environment.

In order to investigate the effects of imbalanced experiences, the skewness was systematically varied between 0%-50% and the performance of both LCS variants determined for each skewness value. Fig. 3 summarises the results, reporting the maximum performance obtained by each algorithm after its performance had stabilised (defined as the average performance for 20000 trials after trial 50000). Reported results are the average of 50 independent runs.

The red broken line in Fig. 3 plots the average performance of SB-XCS while the blue undecorated solid line plots the average performance of XCS. Note that the performance of SB-XCS generally increases as the skewness is varied from 50%

down to 0%, while the performance of the XCS is maximal at the extremes and minimal at a skewness value of approximately 12%. Comparing these two lines, we observe that for highly skewed environments (skewness in the range 2%-25%), SB-XCS clearly outperforms XCS, obtaining a higher maximum performance in these cases. The opposite is true for the less skewed cases — here the performance of XCS is superior to SB-XCS.

Surprised by the the relative poor performance of XCS on highly skewed environments, we varied the algorithm's control parameters in order to "tune" the algorithm for the problem at hand. We found that by raising the threshold before subsumption can occur and making it more difficult for a classifier to be judged as perfectly accurate, the performance of XCS was significantly improved. The green solid line decorated with squares in Fig. 3 plots the performance of this "tuned" XCS variant; results indicating the performance of the tuned algorithm outperforms or matches the performance of the "vanilla" XCS in all cases.

We also investigated the performance of XCS without (either forms of) subsumption. The black solid line decorated with triangles in Fig. 3 plots the performance of this variant. We observe from Fig. 3 that by disabling subsumption, the performance of XCS is further improved, surpassing or matching the performance of the vanilla and tuned variants of XCS with subsumption, and SB-XCS too. The reason for this is that by removing subsumption, the number of specialised classifiers in the population increases, hence allowing a relatively larger knowledge base for decision making.

Overall, Fig. 3 shows that vanilla XCS does not perform as well as SB-XCS on environments with highly skewed experiences. In these cases, either the algorithm needs to be tuned to the problem at hand or replaced with its strength-based counterpart. In uncertain domains where the level of skewness is unknown, a priori parameter tuning may not be possible and some automated mechanism is needed instead. We plan to investigate this idea in future work.

5. CONCLUSIONS AND FUTURE WORK

Fraud detection is a challenging problem due to the adversarial nature of perpetrators and the rarity of fraudulent events among the magnitude of legitimate activity. In this research, we detailed seven general characteristics of fraud and motivated the use of LCSs for learning in such environments. Through the use of abstract problems that exhibit these characteristics, we examined the behavior of accuracy-based and strength-based variants in three experiments that capture these important properties. Results showed that for incomplete and noisy data, the accuracy-based variant outperformed the strength-based variant as expected.

We noted however that in environments with imbalanced experiences, the strength-based LCS outperformed the accuracy-based LCS. This could be partially remedied by tuning parameters or disabling subsumption, but still both variants fail to achieve complete accuracy in this setting. To be an effective method for automated fraud detection, LCSs must be accurate in a highly skewed environments, and future work will involve investigating methods to improve performance in this setting. Other future work will include: improving the emulation of fraud detection problem using real-valued functions and assessing the correlation of these functions to real-world data; examining the complexity of functions as well as their skewness; and investigating methods to automatically tune learning parameters on the fly.

Acknowledgments

The first author would like to acknowledge the financial support provided by the Robert and Maude Gledden Scholarship and ARC Discovery Project DP0771294.

6. REFERENCES

[1] Australian Bureau of Statistics. Personal fraud, 2007. Technical report, 2008.

[2] Jaume Bacardit, Bernadó-Mansilla Ester, and Martin V. Butz. Learning classifier systems: looking back and glimpsing ahead. In *10th International Workshop on Learning Classifier Systems*, pages 1–21. Springer, 2008.

[3] Larry Bull. *Applications of Learning Classifier Systems*. Springer, 2004.

[4] Martin V. Butz and Stewart W. Wilson. An algorithmic description of XCS. Technical Report 2000017, Illinois Genetic Algorithms Laboratory, 2000.

[5] James Dudley, Luigi Barone, and Lyndon While. Multi-objective spam filtering using an evolutionary algorithm. In *Congress on Evolutionary Computation*, pages 123–130, 2008.

[6] Ian Fette, Norman Sadeh, and Anthony Tomasic. Learning to detect phishing emails. In *16th International Conference on World Wide Web*, pages 649–656. ACM Publishers, 2007.

[7] Rod Haggarty. *Discrete Mathematics for Computing*. Addison Wesley, 2001.

[8] John H. Holland. Adaptation. *Progress in Theoretical Biology*, 4:263–293, 1976.

[9] Tim Kovacs. Learning classifier systems resources. *Journal of Soft Computing*, 6(3–4):240–243, 2002.

[10] Tim Kovacs. *Strength or Accuracy: Credit Assignment in Learning Classifier Systems*. Springer, 2004.

[11] Terri Oda and Tony White. Immunity from spam: an analysis of an artificial immune system. In *4th International Conference on Artificial Immune Systems*, pages 276–289. Springer, 2005.

[12] Clifton Phua, Vincent Lee, Kate Smith-Miles, and Ross Gayler. A comprehensive survey of data mining-based fraud detection research. Technical report, Monash University, 2005.

[13] D. Sculley and Gabriel M. Wachman. Relaxed online SVMs for spam filtering. In *30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 415–422. ACM Publishers, 2007.

[14] Alexander K. Seewald. An evaluation of naïve Bayes variants in content-based learning for spam filtering. *Intelligent Data Analysis*, 11(5):497–524, 2007.

[15] Kamran Shafi and Hussein A. Abbass. Biologically-inspired complex adaptive systems approaches to network intrusion detection. *Information Security Technical Report*, 12(4):209–217, 2007.

[16] Kamran Shafi, Tim Kovacs, Hussein A. Abbass, and Weiping Zhu. Intrusion detection with evolutionary learning classifier systems. *Natural Computing*, 8(1):3–27, 2009.

[17] Olivier Sigaud and Stewart W. Wilson. Learning classifier systems: a survey. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 11(11):1065–1078, 2007.

[18] Gary M. Weiss. Mining with rarity: a unifying framework. *ACM SIGKDD Explorations Newsletter*, 6(1):7–19, 2004.

[19] Stewart W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

[20] Ching-Tung Wu, Kwang-Ting Cheng, Qiang Zhu, and Yi-Leh Wu. Using visual features for anti-spam filtering. In *International Conference on Image Processing*, volume 3, pages 509–512. IEEE Publishers, 2005.

[21] Yue Yang and Sherif A. Elfayoumy. Anti-spam filtering using neural networks and Baysian classifiers. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 272–278. IEEE Publishers, 2007.

Situation Recognition in Smart Spaces: A Multi-Agent Approach Using Commitment Machines

Anthony Blond
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: anthonyb@csse.uwa.edu.au

ABSTRACT

This paper briefly describes proposed research on the use of commitment machines for detecting high-level situations in smart spaces. Our approach allows for individual nodes in the system to optimise the detection and transmission of situation components by only processing them if requested to do so, thus reducing computation and communication costs.

Keywords: Agents, Commitments, Composite Events, Situations, Situation Recognition

1. BACKGROUND

Advances in smart technologies and the increasing affordability of more powerful sensing devices is bringing the future of smart spaces closer [5]. This will have a significant impact in many fields, most notably at-home aged and disabled care, where remote health monitoring, emergency response and automated living assistance can decrease the burden on carers and allow people to remain in their homes for longer [4].

The defining feature of smart spaces are nodes with attached sensors and actuators, capable of on-board computation and communication with one another. Ubiquitous computing sees a large number of devices becoming interconnected, thus allowing computational tasks to be distributed across a network of nodes. An important ability of smart spaces is the collection of knowledge of the environment in order to allow for sensible actions to be taken. This can be done either by continuously collecting data from sensors, or by triggering the observations with real-world events. One reason the latter is preferred, as stated by Römer and Mattern [8], is that event-based approaches "provide an implicit data compression" because only state-changes in the real world will lead to communication in the smart space.

To use events as an abstraction of changes in the environment, it is necessary to be able to compose a collection of primitive events into more complex, high-level ones. These constructs are often referred to as *composite events* [7, 6]. This research makes use of a further level of abstraction called *situations*. A situation can be defined as an event sequence constrained by conditions and a set of trigger events.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

An example of a situation in a smart house may be "Resident is preparing dinner", with a possible definition being "kitchen light is on, followed by motion detected and the fridge door opening, with the total time difference less than three minutes". For a list of situation examples of varying complexity refer to [3].

Using situations as a means of representing data from the environment presents two key research challenges. Namely, how can situations be defined in a consistent manner and how can situations in a smart space consisting of a large number of nodes be detected? This research will focus on answering these questions, and there two key works that will be influential.

A brief description of influential works is given in Section 2, followed by a description of commitment machines ending with research goals in Section 3. Section 4 describes the proposed detection mechanism and Section 5 concludes with summarising statements.

2. INFLUENTIAL WORKS

RuleCaster [2] is a high-level rule based *macroprogramming* system developed for smart homes and wireless sensor networks, which approaches the problem of application development by considering the network as a whole. The RuleCaster Application Language (RCAL) makes use of a *space* data type, an abstraction of a distributed computer consisting of a collection of nodes, which may be shared with other spaces. A separate network model defines the services and properties of each node. Each space includes a set of interfaces, pre-states and post-states along with rules for transitions between them. A compiler, run on a central machine, maps services to physical nodes and deploys the spaces as distributed finite state machines (FSMs) across the network (see [1] for more details). The compilation process allows the application programmer to separate functionality from physical implementation, but the rule based system does not allow easy definition of situations and has no support for temporal constraints [3].

Mansouri-Samani and Sloman present GEM [6], an "interpreted generalised event monitoring language" that defines high-level composite events by the combination of smaller, low-level events. Primitive events are assumed to occur instantaneously, and have an associated time stamp, location and type and are combined using a set of five operators defining conjunction ('&'), disjunction ('||'), temporal de-

lay ('+'), temporal ordering (';') and temporal interleaving ('$\{e_1$; $e_2\}$! e_3'). Optional *guards* are available as boolean expressions involving event attributes, which are only evaluated when their corresponding event occurs. The example in Section 1 could be defined (ignoring location for simplicity) as:

```
x:lightOn ; z:(y:motion & fridgeOpen)
    when (|@z - @x <= [3*min]) && (y.value >= 4)
```

Instances of certain event types are specified using event variables with the ':' operator, which are referred to in the guard. The '@' and '|@' operators retrieve the time stamp of a single event and the time stamp of the final event in a sequence respectively. The described detection mechanism requires all primitive events to be collected at a single location and evaluation of a composite event takes place using a binary tree approach, where the leaves are events and the nodes are operators. A bottom-up approach is used for detecting the composite events, with event histories for an entire tree being stored for the same amount of time. This approach is not amenable to being distributed across multiple sensing nodes, resulting in increased communication costs.

3. COMMITMENT MACHINES

Both RuleCaster and GEM have features which are desirable for situation recognition in a smart space, but are by themselves limited. GEM lacks a means of distributed detection, while RuleCaster's distributed compilation solution uses FSMs which result in overly restrictive situation definitions and detection rules. Yolum and Singh [10] propose a solution called *commitment machines* for a similar problem of overly rigid communication protocols in multi-agent systems. In this model, an event calculus is used to formally define rules and reason about commitments between agents. A later work provides more details on the event calculus [11], and the concept is further extended by Winikoff et al. [9].

Cardell-Oliver and Liu [3] extend the concept of commitment machines to allow for the detection of situations. The key feature is the delegation of detection using situation recognisers (SRs) and sub-situation recognisers (SSRs), implemented as agents on different nodes. Continuing with the example from Section 1, an SR would detect a trigger such as the `lightOn` event, and then spawn SSRs on the relevant nodes to detect the occurrence of `motion` or `fridgeOpen` events. Commitments would be established between the SR and SSRs requiring them to announce when they detect the relevant event. Importantly, the SSRs would only be created when the original trigger event occurs.

This research proposes to extend the concept of commitment machines as situation recognisers, using GEM as a language and a compiler inspired by RuleCaster to distribute the detection tasks across multiple nodes in an efficient manner. The aim of this project is therefore twofold:

- to investigate the effectiveness of situations as a high-level representation of real world data in smart spaces;
- to assess the usefulness of commitment machines as means of detecting situations in a distributed framework.

4. METHODOLOGY

The language used for defining situations will be based almost entirely on GEM's notation, with the addition of an iterative operator from the Cambridge event language [7]. Additionally, location tags will be attached to each event by a '$' symbol, for example '`motion$kitchen`'. This tight coupling is required to ensure the correct *distributed* deployment of SSRs. Unlike GEM, which provides a single temporal window (the length of time event histories are stored) for each situation, our system will use separate temporal windows for each component of a situation, established via commitments. For example, the '&' operator will imply a temporal constraint between two corresponding events, which can be automatically chosen or defined by the programmer. By using separate windows, there can be multiple trigger events, and they need not be the first in a sequence.

The *scope* of each event variable will define where the corresponding constraints will be evaluated. The compiler breaks the situation specification down into a tree structure, as shown in Figure 1. The basic deployment strategy is that the branches from each operator node are sent to separate agents that are able to provide the required detection.

In this commitment machine paradigm, three types of commitment exist:

1. *Deployment triggered commitments* (DTCs) are permanent commitments from one agent to another to provide notifications of an event occurring. DTCs are required for the detection of trigger events;
2. *Standby detection commitments* (SDCs) do not require agents to provide notifications but are instead an obligation to keep track of the relevant event history for an agreed upon amount of time;
3. *Event triggered commitments* (ETCs) are established between two agents only after a required event has occurred.

The detection algorithm is best demonstrated with an extension of the existing example (location will again be omitted). Take a network with three agents for example; A, B and C. A is the top level SR with the full situation definition. B is able to provide the detection of `motion` while C is able to detect both `lightOn` and `fridgeOpen`. Initially A will use an existing service discovery method, which is outside the scope of this research, to find which agents have the required detection capabilities. After this, it will establish DTCs for the trigger events. Using the notation from [3] and [11]:

```
CC(B,A,y:motion where y.value >= 4, notify(A,1))
CC(C,A,fridgeOpen,notify(A,2))
```

Here CC means conditional commitment, that is B is committed to notify A about the sub-situation with an ID of 1

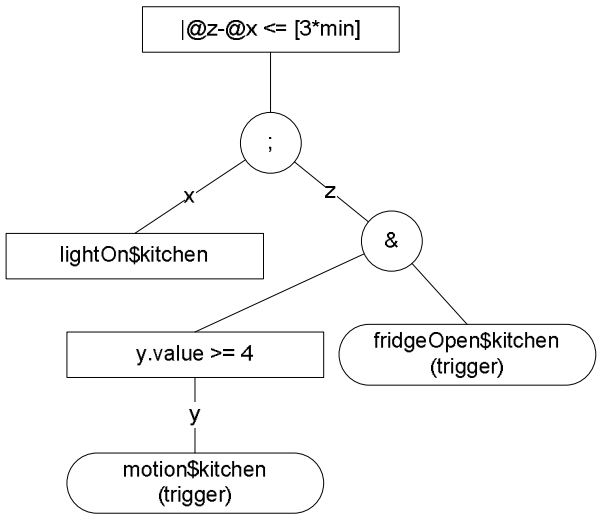


Figure 1: Compilation breakdown

on the condition that the corresponding event and boolean constraints are satisfied. The next step is for A to establish SDCs for the remaining events. This would be represented as:

```
SC(C,A,lightOn for [3*min])
```

The time value would be automatically chosen based on the fact that `|@z-@x<=[3*min]` is the longest temporal constraint. Once the above commitments have been established, A will simply need to wait to be notified by either B or C via the DTCs. Once this has occurred, it will establish an ETC with C to notify it about a potential `lightOn` event. Since the corresponding SDC has already been established, C will be able to check a buffer and notify A if the event occurred in the past, thus allowing the trigger events to occur after other events in a situation, without maintaining a complete history of all events.

This detection strategy will allow for complex situations to be detected across a large number of nodes with a minimal amount of communication and computation on the part of the nodes.

5. SUMMARY

Using the extended GEM language for defining situations we hope to allow situations with complex temporal constraints to be written in a straight-forward manner. Commitment machines will form the basis of a flexible, robust and scalable detection mechanism for these situations. This system could then be implemented in an environment such as a smart home to assist with the collection of high-level event data and home-automation.

6. REFERENCES

[1] Urs Bischoff and Gerd Kortuem. A compiler for the smart space. In *Proceedings of the European Conference on Ambient Intelligence (AmI-07)*, November 2007.

[2] Urs Bischoff, Vasughi Sundramoorthy, and Gerd Kortuem. Programming the smart home. In *Proceedings of the Third IET International Conference on Intelligent Environments*, September 2007.

[3] Rachel Cardell-Oliver and Wei Liu. Representation and recognition of situations in sensor networks. To appear in IEEE Communications Magazine, Special Issue on Situation Management, August 2009.

[4] Diane J. Cook. Health monitoring and assistance to support aging in place. *Journal of Universal Computer Science*, 12(1):15–29, 2006.

[5] Diane J. Cook and Sajal K. Das. How smart are our environments? an updated look at the state of the art. *Pervasive and Mobile Computing*, 3:53–73, 2007.

[6] Masoud Mansouri-Samani and Morris Sloman. GEM - a generalised event monitoring language for distributed systems. *Distributed Systems Engineering*, 4:96–108, 1997.

[7] Peter R. Pietzuch, Brian Shand, and Jean Bacon. Composite event detection as a generic middleware extension. *Network, IEEE*, 18(1):44–55, 2004.

[8] Kay Römer and Friedemann Mattern. Event-based systems for detecting real-world states with sensor networks: A critical analysis. In *Proceedings of the 2004 Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 389–395, December 2004.

[9] Michael Winikoff, Wei Liu, and James Harland. Enhancing commitment machines. In *Proceedings of DALT Workshop, LNAI*, pages 198–220. Springer-Verlag, 2005.

[10] Pinar Yolum and Munindar P. Singh. Commitment machines. In *Proceedings of the 8th International Workshop on Agent Theories, Architectures, and Languages (ATAL-01)*, pages 235–247. Springer, 2000.

[11] Pinar Yolum and Munindar P. Singh. Reasoning about commitments in the event calculus: An approach for specifying and executing protocols. *Annals of Mathematics and Artificial Intelligence*, 42(1-3):227–253, 2004.

Automated Feedback for Improving Software Quality in Software Engineering Education

Rachel Cardell-Oliver, You Hai Lim, Zhang Lu, Rieky Barady and Asad Naveed
School of Computer Science & Software Engineering
M002, The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: rachel@csse.uwa.edu.au

ABSTRACT

This paper addresses the problem of providing timely, formative feedback to improve the quality of software produced by software engineering students. We present an approach that utilizes open source, industrial-strength, software engineering tools. Programming assignments are designed and the tools configured, to provide feedback to students on the quality of their program code. Our preliminary results demonstrate that the feedback both enables and motivates students to improve significantly the quality of their submitted programming assignments.

Keywords: Programming Assessment, Automatic Assessment, Formative Assessment.
CR Classifications: K.3.2 [**Computers and Education**]: Computer and Information Science Education; D.2.5 [**Software Engineering**]: Testing and Debugging

1. INTRODUCTION

The practical component of most Computer Science and Software Engineering courses requires students to write computer programs and submit them for assessment. Student programs can themselves be analysed by programs that evaluate the construction and behaviour of their submitted code. The output of analysis tools can be used to provide formative feedback to students on ways to improve their work.

Computer Science educators have developed many different tools for assessment in programming courses [2]. Findings on the effectiveness of such tools range from Dehnai and Bornat's view in "The Camel has Two Humps" that programming ability is innate and that some students simply cannot learn how to program [6], to Edwards' studies of the effectiveness of tool support for teaching students to write both programs and test suites to demonstrate the quality of their programs [8]. Automated assessment tools have been widely studied for their perceived benefits of providing immediate feedback to students, with feedback available at any time, and the ability of tools to provide consistent and objective assessment. Students need feedback about programming misunderstandings immediately when they are working on programming assignments, rather than later, or they are unlikely to act on that feedback.

On the other hand, the use of automatic assessment tools in university courses has also been criticized for three major weaknesses. First, only what gets measured gets improved. Since not all learning objectives relating to programming can be assessed automatically, students are likely to focus only on those aspects of their programs that are measured by the assessment tools at the expense of most other qualities [2]. Second, students using automatic assessment tools tend to develop working strategies that rely on the instructor's feedback via the tools rather than taking responsibility for the quality of their own programs, and so they do not transfer the new skills learnt to later courses or to working life [8, 2]. Third, different students have different learning styles, and learn at different rates. Automated assessment systems need to be sufficiently flexible for students to be able to reach the standard required at different times [10].

This paper addresses the problem of providing timely, formative feedback to improve the quality of software produced by software engineering students. We present an approach that utilizes industrial-strength software engineering tools. We design programming assignments and configure the tools to provide feedback to students on program quality, enabling and motivating them to improve significantly the quality of their submitted assignments. This paper provides an overview of our assessment approach and tools and presents preliminary results on the effectiveness of this teaching and learning strategy. We conclude that students can indeed be supported in learning to develop high quality programs, measured across a range of attributes, and that they do so in a way that the skills learnt will be transferred to their later working lives.

2. APPROACH

Our focus is a software engineering (SE) course for advanced novices, that is students who have already completed a first programming course (CS1). Learning to program is not easy because students need to master (at the same time) many different skills [11]:

1. General orientation: what programs are for;
2. Notation: syntax and semantics of a programming language;
3. Notional Machine: model of the computer as it relates to executing programs;
4. Structures: schemas and plans for problem solving;
5. Pragmatics: skills of planning, developing, testing, debugging.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

In our work we assume that students have grasped the basics of program syntax and semantics, and have a preliminary understanding of synthesis, planning and pragmatics.

In software engineering courses educators aim to equip students with skills they will need on graduation to develop large programs as part of an engineering team. That is, the ability to deliver high quality, tested programs, to write readable, well documented programs, and the knowledge to use a programming language such as Java effectively. These goals shape our requirements in designing a system to provide feedback and automatic assessment in SE labs. Students perform exercises using existing software engineering tools, which we have configured to provide appropriate feedback. The selection of tools for this purpose is guided by the following requirements:

1. The tools should be widely available, preferably open source, and used in industry practice;
2. The tools should be able to be configured for different levels of programmer skill;
3. The tools should be able to provide meaningful feedback to help students improve their work, not just a mark;
4. In aggregate, the tools should assess as many different aspects of software quality as possible.

Five tools were chosen for the project: Eclipse, Checkstyle, JUnit, Abbot-Costello and Clover. In this section we summarize each tool and explain how we have configured and utilized these tools to provide timely, formative feedback for advanced novices studying software engineering.

2.1 Eclipse

Eclipse is an industrial-strength, integrated development environment (IDE) for software developers [7]. Eclipse is an open source project, and freely available versions are easy to install in university laboratories and on students' home computers. Eclipse provides support for file management, compilation, and execution of Java programs. Plug-ins have been developed for many additional features including code style checking, testing, configuration management, and UML documentation.

Eclipse is used by students in programming exercises to provide a uniform platform for different software quality tools, to provide context dependent on-line help for Java and its libraries, and for managing programming project files. We have configured Eclipse with a selection of plug-ins for the purposes of our exercises. For example, as well as Checkstyle and JUnit, we include a Violet-UML plug-in for drawing UML class diagrams from Java source code.

2.2 Checkstyle

Checkstyle is a software tool for testing that program code conforms to a set of style rules. Checkstyle rule sets include rules for code formatting, notational conventions, and coding style for Java programs [3]. Checkstyle rule sets are configurable, enabling instructors to select different rule sets or create unique rules for different stages of student learning, and different educational purposes. For example, the amount of Javadoc documentation students are required to write may be varied in different rule sets. Checkstyle is available as a plug-in for many popular IDEs including Eclipse. The plug-in tool highlights all non-conforming code, provides an explanatory error message, and suggests possible fixes wherever a Checkstyle test is not passed. Checkstyle can also be used as a a stand-alone application allowing more experienced users or instructors to use it in marking scripts or as a command-line interface.

In our assignments we use the Checkstyle plug-in for Eclipse along with two different rule sets, Beginner and Advanced. Both rules sets are a subset of the full Sun Java conventions. The students start by using Beginner rule set designed to ensure that basic formatting and Javadoc rules are satisfied, but that students are not overwhelmed with rules. In initial exercises, students are provided with outline code, so that they can focus on correcting the style errors in existing code, and making sure the new method code they add in selected places follows the style rules. Later, they develop new code from scratch, but are encouraged to refactor earlier exercises where suitable. The Advanced rule set is introduced in later stage when the students have familiarized themselves with the Beginner rule set.

2.3 JUnit

JUnit supports the creation and execution of unit tests for Java classes [9]. The Eclipse JUnit4 plug-in provides support for automatically generating a test suite for a given Java class, and for running and re-running tests with a single click. A "green bar" indicates that all test cases have been passed. Students can view a list of test cases passed or failed, and also can view the test code or follow the feedback messages produced as tests are executed. JUnit test suites are easily extendable with additional student and instructor test cases.

A set of instructor test cases is designed for each programming exercise to uncover common errors observed in students' programs. Common errors can be identified from the literature on novice programmers [11]. We have also determined likely errors from analysis of a database of student code submissions from earlier years and in other units. Each test includes a feedback message that is shown if the test fails and that guides the student on how to correct their code. For example,

```
@Test
public void testchargeInterestNormal() {
  b1.withdraw(2000); //now balance is -1000
  assertEquals(
      "Should debit account by $50 interest charged.",
      -1050,b1.getBalance());
  assertEquals(
      "Charging interest can also change minBalance.",
       -1050,b1.getMinbalance());
}
```

2.4 Abbot-Costello

Abbot is a Java library for testing Swing graphical user interfaces (GUIs) [1]. The tool set comprises Abbot, which

allows the programmer to drive user interface components, and the script editor Costello [5] for creating test suites for GUI programs, and to launch, explore and control an application. Most importantly, it supports both recording of user inputs and developer programmed test scripts [1]. A script editor is provided that can record user actions and facilitate script construction and maintenance. It also provides insights into the hierarchy of the application under test. The Costello editor provides recording functionality for testing GUI programs. Recorded user interactions are saved in XML scripts. JUnit is used as the controlling harness for running tests and suites of tests, but scripts can easily be wrapped to run in other environments. Scripts may also be used to create a demo or tutorial for an application.

2.5 Clover

Clover [4] is an industrial strength code coverage and analysis tool. It assesses how much of the source code is being executed by a suite of JUnit test cases. Clover shows exactly what test cases hit each statement of the source code, thus enabling the programmers to continually improve their testing. Clover gives insight into any testing weaknesses or redundancies and helps the programmer verify whether each test is actually testing the code it was intended for. Clover also measures the complexity of the code. Clover generates detailed and well formatted reports in HTML, PDF and XML formats. It has a plug-in for Eclipse that gives instant feedback to the programmers by making use of code highlighting and other visual cues to show which parts of the code is covered by a test suite and those parts of code that are not covered. Clover is available under license. It is the only tool in our investigation that is not an open source project.

3. RESULTS

This section presents preliminary results from student submissions of four programming tasks. The tasks were completed over two weeks of laboratories that provided an introduction to the JUnit tool, as well as practice with basic Java programming skills. Before this, the students completed an introductory exercise to introduce the Eclipse IDE and the Checkstyle tool.

The BankAccount (BA) exercise requires students to complete method code for given signatures. The missing code requires if statements but no loops. In the Calculator (CA) exercise, students define six simple calculator operators, and for this exercise they also write their own test cases. The majority of students submitted about 9 test cases, but a few submitted up to 27. The ArrayUtilities (AU) exercise tests students' ability to manage loops and array structures. They implement methods on arrays of doubles including sum, average, max, find and insertion operations. The CustomersList (CL) exercise asks the students to produce similar functionality to the ArrayUtilities exercise, but this time using the Java collection class ArrayList. Advanced students also refactored their code using the Java HashMap collection class.

The following table shows the average lines of code (LOC) and non-comment lines of code (NLOC) for student submissions of each of the four exercises. We show both the total number of students who submitted valid code, that is code that compiles correctly, and also the number whose submissions did not compile. Compiler failures cover failure to compile the student code, or to compile the instructors tester with the student code.

Exercise	BA	CA	AU	CL
LOC (avg)	66	37	64	75
NCLOC (avg)	37	19	32	36
Compilation failed	11	4	12	7
Correctly compiled	45	46	50	34

We measured the percentage of valid student submissions that satisfied quality thresholds for each exercise: either all tests passed, or at least 75% of tests passed for both the Checkstyle and JUnit tests. It can be seen from the following table that most students are able to satisfy at least 75% of the quality tests on their first submission attempt. Although a significant group of students made incorrect submissions, most of these were quickly corrected in a second submission.

Quality Threshold	BA	CA	AU	CL
JUnit 100%	80.0%	97.8%	84.0%	94.1%
JUnit 75-99%	20.0%	2.2%	16.0%	5.9%
Checkstyle 100%	66.7%	71.7%	58.0%	79.4%
Checkstyle 75-99%	24.4%	19.6%	32.0%	20.6%

The results demonstrate that students quickly become effective users of JUnit and Checkstyle to check their code as they write it, and so to develop high quality code in a productive manner. Students are able to use the tools to assess their own work using test suites and configuration files provided by the instructor. JUnit's famous "green bar" which is shown when all JUnit tests are passed has proved motivating, and gives a clear goal and a stopping point for students. The feedback comments appear to have helped students to correct errors and so more than 80% of students were able to develop programs that pass all JUnit test cases. On the other hand, Checkstyle's highlighted rules are easier to ignore, and students are less used to these, so the rate of compliance with these rules is lower, at least in the initial stages. Although Checkstyle warnings can be elevated to stop students from compiling code that does not pass the rule set, this solution was considered counter-productive for our purposes. Configuration of both Checkstyle and JUnit is important, so that students are never presented with too many open problems to fix, but can progress from one stage to the next.

Our class has a very wide range of backgrounds and abilities, including a large group (around 40%) who have not studied Java before. Self-assessment allows different students to progress at different rates, and provides a framework for stretching the best, whilst also supporting the weakest. Allowing students to control their own learning around a set of advised deadlines appears to be effective in helping more students reach a baseline standard. Preliminary results suggest that students welcome the opportunity to correct and resubmit their work. For example, 20 students revised and resubmitted their work made within two days of being informed that their program submissions did not meet the required standard. In future work we plan to evaluate dif-

ferent student learning curves, that is how quickly different students reached the required standards.

4. CONCLUSIONS

This paper addresses the problem of providing timely, formative feedback to improve the quality of software produced by software engineering students. We present an approach that utilizes industrial-strength software engineering tools, and design programming assignments and configure the tools to provide timely, formative feedback to students on program quality. Preliminary results suggest that this approach is motivating and enables students to improve significantly the quality of their submitted assignments measured across several different attributes.

Automated feedback tools are not a replacement for student-instructor interactions. Indeed, our laboratory sessions have been busier than usual, possibly because the requirements for passing each exercise are more explicit than in previous versions of the course. Our methods do, however, free the instructor to help students with the more difficult aspects of learning to program such as plans, while style, syntax, and functional correctness rules are assessed automatically and immediately by the software tools.

Acknowledgements

The authors would like to thank the students of CITS1220 Software Engineering for their valuable feedback and suggestions. This research was supported by an Improving Student Learning grant from the UWA Centre for the Advancement of Teaching and Learning.

5. REFERENCES

[1] Abbot. Abbot. Java GUI Test Framework, 2002. http://abbot.sourceforge.net/doc/overview.shtml.

[2] K Ala-Mutka. A survey of automated assessment approaches for programming assignments. *Journal of Computer Science Education*, 15(2):83–102, 2005.

[3] Checkstyle. Checkstyle plugin for Eclipse version 5.0.0.200906281855-final. http://eclipse-cs.sourceforge.net/.

[4] Clover. Code Coverage for Java - Clover. http://www.atlassian.com/software/clover/.

[5] Costello. Getting Started with Costello. http://abbot.sourceforge.net/doc/overview.shtml.

[6] Saeed Dehnadi and Richard Bornat. The camel has two humps (working title). 2006.

[7] Eclipse. Easyeclipse desktop java version 1.3.1.1. http://www.easyeclipse.org/site/home/.

[8] SH Edwards. Rethinking computer science education from a test-first perspective. In *Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages and applications*, pages 148 – 155. ACM, 2003.

[9] JUnit4. JUnit 4. http://www.junit.org/.

[10] Raymond Lister and John Leaney. First year programming: Let all the flowers bloom. In Tony Greening and Raymond Lister, editors, *Fifth Australasian Computing Education Conference (ACE2003)*, volume 20 of *CRPIT*, pages 221–230, Adelaide, Australia, 2003. ACS.

[11] A Robin, J Rountree, and N Rountree. Learning and teaching programming: A review and discussion. *Journal of Computer Science Education*, 13(2):137–172, 2003.

Estimating Conceptual Similarities using Distributed Representations and Extended Backpropagation

Peter Dreisiger[1,2], Cara MacNish[2] and Wei Liu[2]
[1] Maritime Operations Division, Defence Science and Technology Organisation
[2] Computer Science & Software Engineering, The University of Western Australia
email: {prd,cara,wei}@csse.uwa.edu.au

ABSTRACT

The ability to perceive similarities and group entities into meaningful hierarchies is central to the processes of learning and generalisation. In artificial intelligence and data mining, the similarity of symbolic data has been estimated by techniques ranging from feature-matching and correlation analysis to *Latent Semantic Analysis (LSA)*. One set of techniques that has received very little attention are those based upon cognitive models of similarity and concept formation.

In this paper, we propose an extension to a neural network-based approach called *Forming Global Representations with Extended backPropagation (FGREP)*, and show that it can be used to form meaningful conceptual clusters from information about an entity's perceivable attributes or its usage and interactions. By examining these clusters, and their classification errors, we also show that the groupings identified by FGREP are more intuitive, and generalise better, than those formed using LSA.

Categories and Subject Descriptors:

I.2.4 [**Artificial Intelligence**]: Computing Methodologies — Knowledge Representation Formalisms and Methods

Keywords:

Subsymbolic processing, Knowledge representation, Dimensional reduction, Distributed representations, Neural networks

1. INTRODUCTION

The ability to perceive similarities, and form meaningful, or 'natural', groups underlies some of our most important mental processes. In remembering, it allows us to go beyond superficial correspondences and identify precedents based upon structural, or deeper, similarities. In problem solving, it allows us to draw upon our past observations and experiences, and to find solutions in a timely and context-appropriate manner. And in learning, these groupings determine the nature of the associations and the quality of our generalisations.

In the fields of artificial intelligence and data mining, the process of placing similar objects or observations into groups is called *cluster analysis*, and its goal is to find an arrangement that maximises both the *intra*-cluster similarities and the *inter*-cluster differences. While the most common estimates of these differences are also measures of distance, choosing the 'best' measure can be far from trivial — on the one hand, it depends upon the form and representation of our observations; on the other, it determines the make up of the clusters and the types of relationships they capture.

For symbolic data, such measures should capture salient differences in the terms' roles and features. Techniques such as feature matching and correlation analysis are commonly used to estimate the difference between feature vectors or sets of objects [2]. However, these approaches have several limitations: firstly, the vectors' high dimensionality, and their resulting sparseness, can be a problem for traditional measures of distance; secondly, they treat each variable, or dimension, as equally important — an assumption that is often incorrect; and thirdly, their focus on the presence or absence of terms makes them blind to the order of terms and, thus, their roles.

One way to provide a more realistic estimate of the differences is to manually weight each variable according to its importance, or salience. Unfortunately, this requires the weights to be known beforehand, and it assumes that they are constant across situations and contexts. Another solution is to find a transformation that better captures, or accounts for, the variations in the raw data. The most common way of finding these transforms involves the use of *dimensional reduction*, and within data mining, one of the most powerful and widely used examples of this is *Latent Semantic Analysis (LSA)*.

LSA was developed to analyse and characterise written documents [4], and it uses dimensional reduction to minimise the effects of its inputs' sparseness and biases. Not only does this produce representations that better reflect the words' average meanings, but the distances between them can be used by traditional clustering algorithms to identify groups of related terms. What it cannot do, however, is record word order or differentiate between roles.

The need to represent symbolic terms, and to capture their 'essence' in a relatively small representation space, is not confined to the fields of artificial intelligence and data mining. Within the cognitive sciences, these two tasks are largely subsumed by the study of one of *our* greatest natural abilities — concept formation; indeed, the ease with which we can recognise salient similarities, and our ability to draw in-

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

dividual instances together into a common hierarchy of concepts, are subjects of ongoing research. Theories, such as Rosch et al.'s *Basic Level of Categorisation* have attempted to account for the order in which we acquire concepts [8] , while others have even tried to explain the nature of concepts, and the process of concept *formation*, in terms of conceptual spaces and geometric representations (see, for example, Gärdenfors' [1] model).

In this paper, however, we will focus on a particularly promising model of sub-symbolic processing that was first described by Miikkulainen and Dyer [6]. Like LSA, their technique, called *Forming Global Representations with Extended back-Propagation (FGREP)*, represents symbolic data as vectors, uses a form of dimensional reduction to emphasise deeper correlations, and produces representations whose relative distances reflect underlying similarities. Unlike LSA, it develops these representations along a relatively small set of opaque dimensions, and it takes order into account.

Through a series of experiments, we investigate FGREP's ability to produce meaningful clusters given an entity's perceivable attributes, or its usage and interactions. We compare these groupings to the reference class hierarchies and those found using LSA, we examine how well the resulting clusters generalise and capture differences in the terms' roles, and we show that it can be used to provide a more intuitive estimate of similarity. We begin, in Section 2, with an overview of LSA and FGREP. In Section 3, we describe our variation of the FGREP model and the test data, and in Section 4, we present the results of two experiments that focus on perceptual data and term usage. We close, in Section 5, with a summary of our results and an outline of our future work.

2. BACKGROUND

LSA is a well established statistical technique that accepts a matrix of term–passage frequencies, and uses dimensional reduction to compute their 'average meanings'. More specifically, it uses reduced-rank singular value decomposition to produce estimates of the average frequencies that are based upon the terms' shared neighbourhood, and their distribution across passages. Not only do these estimates reflect the words' meanings more closely than the raw frequencies, but the distances between the resulting row and column vectors can be used by traditional clustering algorithms to identify groups of related terms.

LSA is fundamentally deterministic, it can scale to relatively large corpora, and studies have shown that its estimates of inter-document similarity are comparable to those produced by human subjects [4]. LSA does, however, have some limitations: it assumes that the inputs can, in fact, be divided into natural bundles of information such as paragraphs or documents, and it is blind to the ordering of words within a document [3]. While LSA is well suited to the analysis of *textual* documents, this latter fact, in particular, affects both its ability to form clusters from declarative knowledge, and its ability to arrange them into meaningful hierarchies.

FGREP is a cognitive model that was introduced by Miikkulainen and Dyer [6], and explored further by Miikkulainen [5]. It is built around a multi-layer perceptron, and the essence of this model is that: (1) every term is represented by a numerical vector, called a *distributed representation*; (2) these representations change, from initially random values, to ones that capture patterns in the terms' usage; and (3) they are developed *automatically* by propagating the error signals back through the hidden layer to the network's inputs. Put another way, this extended form of backpropagation changes both the network's weights *and* its inputs, and FGREP uses these additional degrees of freedom to further improve the network's accuracy.

Structurally, FGREP consists of three components: a distributed representation store, a 'routing' network, and a three-layer perceptron. The store keeps track of the terms' current distributed representations; the purpose of the routing network is to convert tuples of terms into input and target vectors, and vice versa. At the beginning of each training cycle, the routing network retrieves the appropriate representations from the store, concatenates their current patterns of activation, and presents the resulting vector to the neural network's input and target layers (see Figure 1a). The errors are then calculated, and once the inputs have been updated, the routing network reverses the process by extracting the terms' new representations and placing them back into the store (Figure 1b). It is this circulation of representations that allows them to develop.

While FGREP is, to the best of our knowledge, the first neural system that changes its own inputs to better reflect regularities in the training data, it is not the only one to use extended backpropagation. In particular, this rule, which was called *backpropagation-to-representation* by Rogers and McClelland [7], formed the basis of their model of human concept formation. What distinguishes FGREP from this approach is its use of a distributed store and a routing network. Even though backpropagation-to-representation develops a set of distributed representations, it does so across a hidden layer — the input and target patterns are still expressed as feature vectors and, for this reason, the number of terms it can support is completely determined by the size of its input and output layers.

In contrast, FGREP's capacity is largely independent of its network's geometry. The structure of FGREP's networks also means that it is able to form its representations from a series of declarative sentences (rather than a vector of frequencies or hand-picked features), and that it can do so *incrementally* and using an arbitrary sentence structure.

3. METHODOLOGY

In this paper, we investigate the types of clusters that FGREP can produce given either perceptual data or information about an entity's usage and interactions, we compare them to the reference hierarchies and those found using LSA, and we examine how well the resulting clusters generalise. This is in contrast to the earlier studies which used FGREP or extended backpropagation as part of a larger natural language processing system, or to study the process of human concept formation.

The system used in our experiments extends that of Miikkulainen and Dyer [6]. To focus on developing the distributed representations, and to improve their stability, we

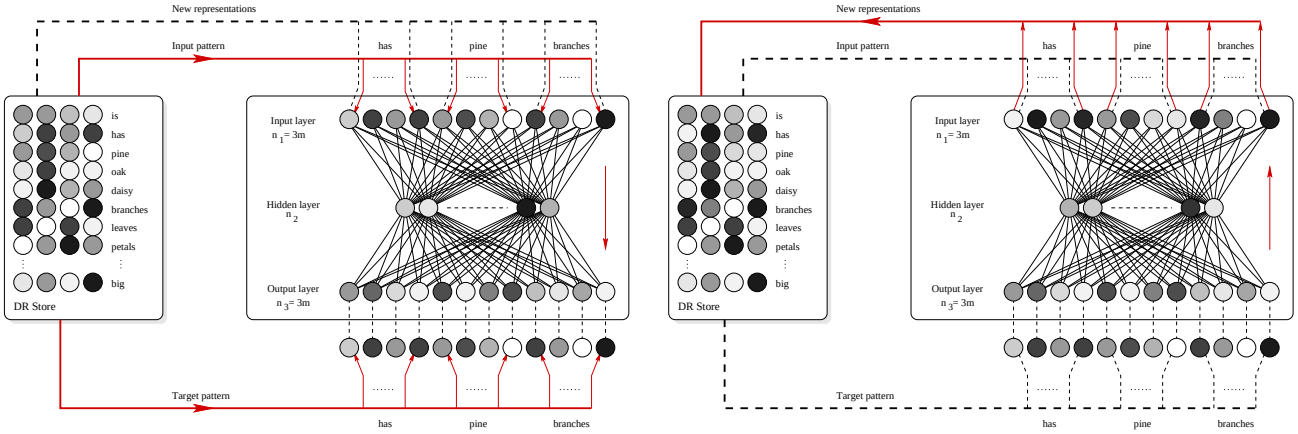


Figure 1: A generalised FGREP system, based upon Miikkulainen and Dyer [6]. The system consists of a symmetric, three-layer perceptron, a routing network, and a distributed representation store. In this example, the network is being taught to remember a three-term (or symbol) input sequence; at the same time, the representations used to construct the sequences are being updated by propagating the errors back to the input layer.

made several changes to the network's structure, input–target pairs and learning rule. Firstly, we used a *symmetric* three-layer perceptron, or auto-encoder, in place of the original model's asymmetric feed-forward network. Secondly, we taught our network to reproduce its input sentences where Miikkulainen's system was trained to identify the agents, patients, actions and modifiers given a sentence's syntactic representation. (See Figure 1 for a diagram of our generalised FGREP system.)

To improve the stability of the system, we implemented a batch learning algorithm alongside the existing sequential, or iterative, rule. (In sequential mode, the order of the training data varies from epoch to epoch, and the weights are changed after the presentation of each example; in batch mode, the weight changes are accumulated over the entire training set, and the differences are only applied at the end of each epoch.) Normally, the latter tends to yield better results and a faster rate of convergence; in our case, however, it also means that otherwise identical terms — i.e. those with the same initial position and usage — will develop divergent representations[1].

Finally, our system supports both dynamic and static representations. While the vast majority of representations continue to develop over time, this addition allows some terms, and the relations in particular, to be given a fixed representation. This use of static relations follows from Rogers and McClelland [7], and initial tests have shown that the judicious use of fixed representations can help to 'ground' the system and improve both the quality of the other representations and their rates of convergence.

For evaluation, we used two distinct data sets. The first is based on the training corpus of Rogers and McClelland [7, Appendix B2], but incorporates two changes: (1) the data was converted from a set of binary feature vectors to a sequence of `relation object attribute` sentences; and (2) information about the entities' class hierarchies was removed to produce a purely descriptive data set. The second set consists of the sentence templates and noun categories of Miikkulainen [5, Tables 5.1 and 5.2]; to generate the training set proper, we went through the list of templates and enumerated every possible instance according to the list of nouns. These sets are shown in Tables 1a and 1b, respectively.

The data sets were chosen for two main reasons: (1) they allow us to verify the behaviour of our system using existing corpora, and (2) collectively, they allow us to see how FGREP performs under a variety of conditions. The first set is relatively small, contains only perceptual information, and represents knowledge using `relation object attribute` triples. In contrast, the second set is larger (containing nearly 700 sentences versus 60), consists of relational information, and uses five-part tuples to describe its entities and the roles they play.

The experiments consisted of 25 trials that ran for 100,000 epochs. We used default values for the distributed representation and hidden-layer sizes (12 and 18 neurons, respectively[2]), and sampled the representations periodically to track their development. Each trial within an experiment used the same declarations but seeded the random number generator with a different value; this meant that each trial had a unique set of distributed representations — both initial and learned. On a 2.8GHz Pentium IV system, each trial took approximately six minutes to execute for the first set of data, and 190 minutes for the second.

[1] Our choice of learning rule, and the way in which it affects the representations' sensitivity to perturbations in their initial patterns of activation, will be the subject of a later investigation, however initial tests suggest that batch learning leads, on average, to a better set of final representations.

[2] The effects of the distributed representation and hidden layer sizes are the focus of ongoing experiments; for these trials, however, we used the default values, as recommended by Miikkulainen [5, p. 54].

	pine	oak	rose	daisy	robin	canary	sunfish	salmon
is-pretty	0	0	1	1	0	0	0	0
is-big	1	1	0	0	0	0	0	0
is-living	1	1	1	1	1	1	1	1
is-green	1	0	0	0	0	0	0	0
is-red	0	0	1	0	1	0	0	1
is-yellow	0	0	0	1	0	1	1	0
can-grow	1	1	1	1	1	1	1	1
can-move	0	0	0	0	1	1	1	1
can-swim	0	0	0	0	0	0	1	1
can-fly	0	0	0	0	1	1	0	0
can-sing	0	0	0	0	0	1	0	0
has-skin	0	0	0	0	1	1	1	1
has-roots	1	1	1	1	0	0	0	0
has-leaves	0	1	1	1	0	0	0	0
has-bark	1	1	0	0	0	0	0	0
has-branch	1	1	0	0	0	0	0	0
has-petals	0	0	1	1	0	0	0	0
has-wings	0	0	0	0	1	1	0	0
has-feathers	0	0	0	0	1	1	0	0
has-gills	0	0	0	0	0	0	1	1
has-scales	0	0	0	0	0	0	1	1

(a)

`human` ate	
`human` ate `food`	
`human` ate `food` with `food`	
`human` ate `food` with `utensil`	
`animal` ate	
`predator` ate `prey`	
`human` broke `fragileobj`	
`human` broke `fragileobj` with `breaker`	
`breaker` broke `fragileobj`	
`animal` broke `fragileobj`	
`fragileobj` broke	
`human` hit `thing`	
`human` hit `human` with `possession`	
`human` hit `thing` with `hitter`	
`hitter` hit `thing`	
`human` moved	
`human` moved `object`	
`animal` moved	
`object` moved	

`human`	boy girl man woman
`animal`	bat chicken dog lion sheep wolf
`predator`	lion wolf
`prey`	chicken sheep
`food`	carrot cheese chicken pasta
`utensil`	fork spoon
`fragileobj`	plate vase window
`hitter`	ball bat hammer hatchet paperwt rock vase
`breaker`	ball bat hammer hatchet paperwt rock
`possession`	ball bat dog doll hammer hatchet vase
`object`	ball bat carrot cheese chicken curtain desk dog doll fork hammer hatchet paperwt pasta plate spoon vase window
`thing`	`animal human object`
`action`	ate break hit move

(b)

Table 1: (a) The entity definitions, after Rogers and McClelland [7, Appendix B2]; and (b) the sentence templates and noun categories of Miikkulainen [5, Tables 5.1 and 5.2].

4. EXPERIMENTS AND RESULTS

The representations resulting from training were analysed using the open-source statistical package, R [9]. Firstly, dendrograms depicting the relationships between each of the terms were generated using hierarchical cluster analysis and the Euclidean measure of distance. The distances between each of the terms' distributed representations were averaged over all 25 trials to produce the 'typical' cluster plots shown in this paper; term–passage frequency vectors were also calculated using LSA in R, with stemming disabled, and an intermediate dimensionality that was half the number of terms in the corpus.

Secondly, the average per-term reconstruction errors squared were calculated and plotted against time to visualise the relative rates of convergence. That is, the squared difference between each distributed representation and the corresponding signal at the network's output layer was calculated, and averaged over the 25 trials; these values were further averaged to determine the per-category and overall errors.

Finally, the classification accuracies of FGREP and LSA were compared. The statistic we chose for this analysis was *category intrusion*, defined as the number of *unrelated* terms whose representations fall within the hypersphere defined by the category's centroid and its furthest member (i.e. its radius). Not only does this metric provide us with an indication of the techniques' relative accuracy, but by calculating the intrusions for the super-classes as well as the categories, we can see the extent to which they overlap, and if the clusters, themselves, form meaningful class hierarchies.

4.1 Experiment 1: Identifying Concepts by their Properties

The first experiment focuses on the nature and quality of the clusters that FGREP can form given only information about an entity's *perceivable* attributes, and its results are summarised in Figures 2 and 3. From Figure 2a, we can see that LSA tends to emphasise co-occurrences over semantic similarities; for example, it contains groups such as `(robin (sing (fly feathers wings)))`, `(swim gills scales)`, `(petals pretty)` and `(move skin)`. It is, perhaps, for this reason that there are some inconsistencies within the entities' hierarchy: for example, there is a significant distance between the `canary` and the `robin`, and the `trees` are closer to the `fish` than the `flowers`.

In contrast, the clusters formed using FGREP capture both the terms' associations *and* type (see Figures 2b–d). Continuing with the above examples, after just 10,000 epochs FGREP was able to separate the entities, parts, attributes and actions into their own clusters, each of which could be further divided into plant and animal varieties:

```
(((salmon sunfish) (canary robin)) ((oak pine) (daisy rose)))
(((feathers wings) (skin (gills scales))) ((bark branch)
                                          (petals (leaves roots))))
((big green) (yellow (red (living pretty))))
((move (fly sing)) (grow swim))
```

Furthermore, in all of the 25 trials, this process of differentiation continued over time — i.e. the longer the training continued, the more distinct these clusters became.

Figure 3a shows the per-term and per-category errors-squared, averaged over the 25 trials. From this plot we can see not only how the reconstruction errors change over time, but how well the distributed representations capture the terms' nature. On the first point, we can see that, as a rule, the errors decrease for the first 40,000–50,000 epochs; after this point, however, they seem to plateau, or even oscillate. While determining the reason for this behaviour is beyond the scope of this paper, it might be a sign that the network is trying to over-fit the representations, in turn, to each of the sentences in which they appear.

On the second point, we can see that the representations corresponding to the *entities* — that is, the plants and an-

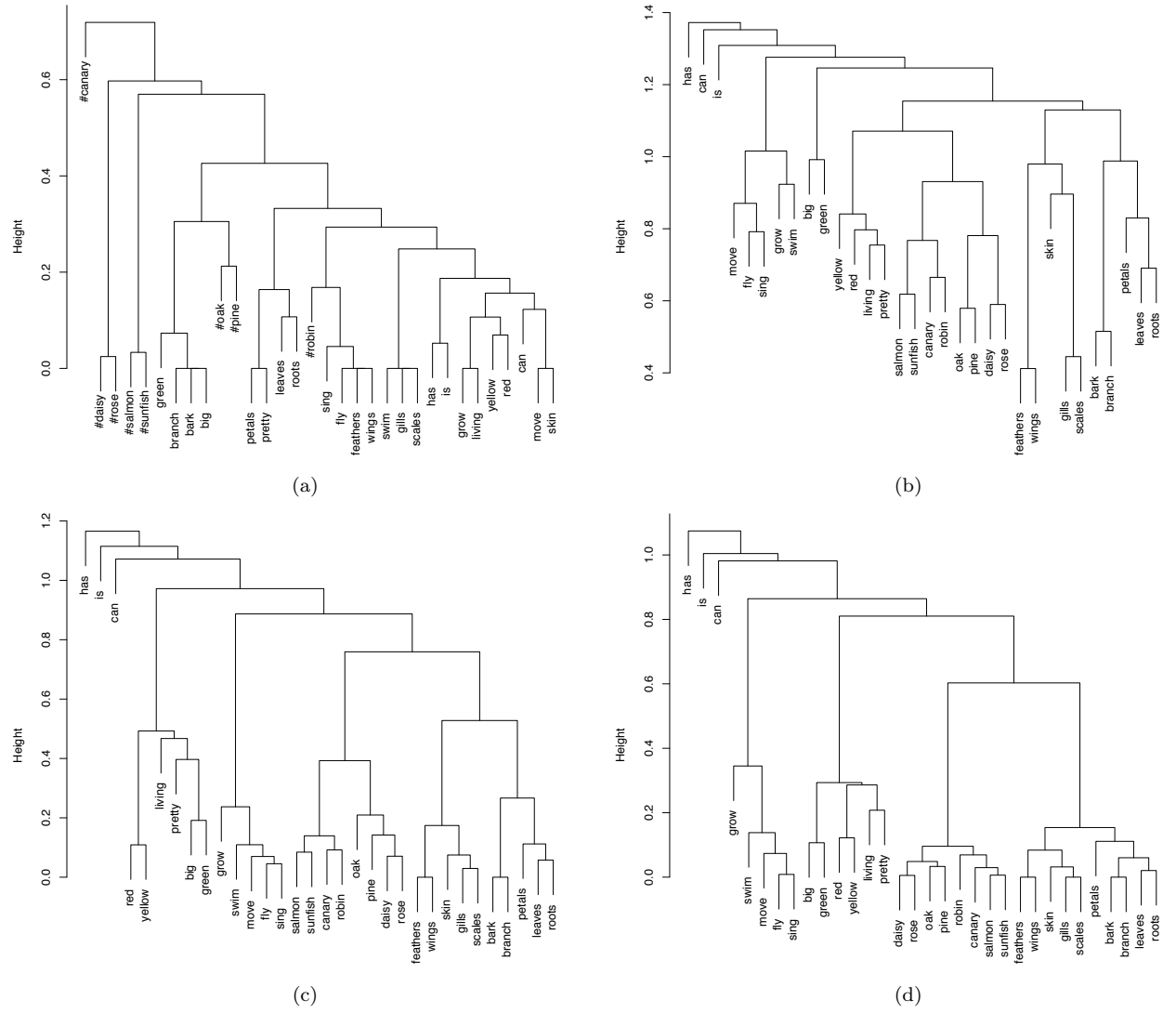
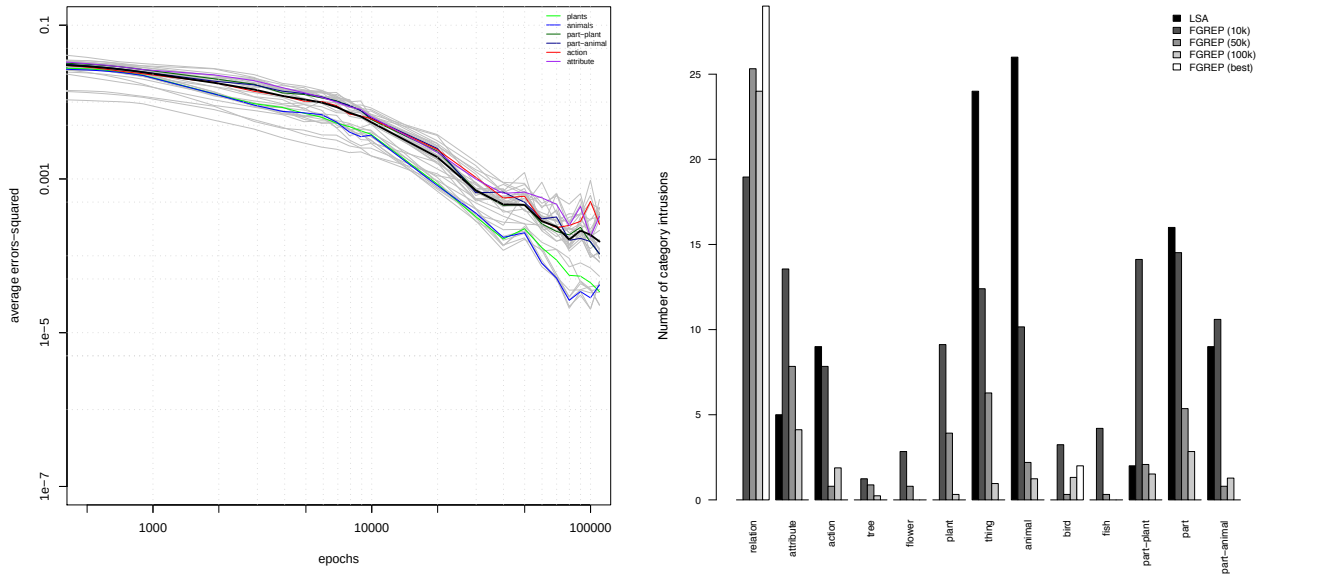


Figure 2: Hierarchical cluster plots based upon: (a) the distances as estimated by LSA; and the average distances, as estimated by FGREP, after (b) 10,000, (c) 50,000 and (d) 100,000 epochs. The average distances were calculated over all of the 25 trials, while the join heights indicate the Euclidean distance between term representations and/or clusters.

imals — had the lowest reconstruction errors while the actions and attributes (which spanned these categories) had the highest errors. In particular, there appears to be an inverse relationship between a term's reconstruction error and the information gain it offers. Take, for example, the animals and the actions (represented by the blue and red lines, respectively): given an animal, we can completely predict which other terms are about to appear; most actions, however, are associated with several plants and/or animals and, as such, they have less predictive power.

Looking at the category intrusions of Figure 3b, and the errors associated with LSA, we can make several observations: (1) while LSA is able to pair the trees, flowers, birds and fish off into their own distinct clusters, it is unable to do the same for the parts, attributes and actions; (2) even though it is able to correctly group the entities, it has difficulty *generalising* these lower-level clusters into higher-level concepts such as animals, things and parts; and (3) it groups all of the relations together, even though they are used in completely different senses.

The subsequent four sets of errors in Figure 3b were calculated from the results of the FGREP trials. The first three were based on the average inter-term distances after 10,000, 50,000 and 100,000 epochs; the last shows the results of the 'best' trial (i.e. the one with the fewest classification errors). In contrast to the LSA results, we can see that: (1) while FGREP is, on average, able to correctly group the trees, flowers, birds and fish, it does occasionally misclassify some of the entities; (2) unlike LSA, however, it *is* able to separate the parts, attributes and actions into their own clusters; (3) while the number of intrusions increases as we move up the class hierarchy, it is actually able to generalise the lower-level clusters into higher-level concepts; and (4) the relations remain relatively distinct.



(a) (b)

Figure 3: (a) The average per-term and per-category reconstruction errors-squared, calculated over the 25 trials; and (b) The category intrusions when the terms were grouped according to their LSA and FGREP measures of similarity. The first three FGREP results are the average number of category intrusions after 10,000, 50,000 and 100,000 epochs, calculated over the 25 trials; the final column shows the number of intrusions, after 100,000 epochs, for the best individual trial.

Even with a relatively small data set, this experiment demonstrated that FGREP, and thus extended backpropagation, are able to form novel conceptual clusters from perceptual data alone. Like the network of Rogers and McClelland [7], our implementation was able to correctly identify the relationships between the birds, fish, trees and flowers. Unlike their system, our adaptation of Miikkulainen's feed-forward network was also able to develop representations for the other terms — representations that captured not only the terms' basic *types*, but their finer structure as well.

When compared to latent semantic analysis, FGREP seems to form more intuitive clusters, and is better able to arrange these groupings into meaningful hierarchies; in other words, while the average number of category intrusions increases as we move up the classification tree, it does so slowly enough for the resulting generalisations to still be useful. Of course, FGREP is a stochastic process and, thus, the quality of the representations it produces are affected by their initial values; this is in contrast to LSA, which is fundamentally deterministic.

4.2 Experiment 2: Identifying Concepts by their Usage

In the first experiment, we focused on FGREP's ability to form clusters from perceivable attributes; the aim of the second experiment is to see how it performs given only information about an entity's usage and interactions, and its results are shown in Figures 4 and 5. From Figure 4a, we can see that LSA tends to favour co-occurrences and shared 'neighbourhoods' over similarities in the terms' usage. Consider, for example, the groups `(ate (pasta carrot cheese))`, `(lion sheep wolf)` and `(paperwt vase)`. The first group contains both the unambiguous `foods` and the action `ate`. The second has placed the `predators` in with a `prey`, even though they play quite different roles in the training data. The final group consists of `paperwt` and `vase`; while both of these terms belong to the `thing`, `object` and `hitter` categories, there are some *semantically* significant differences too — `vase` is also a member of the `fragileobj` and `possession` groups while `paperwt` belongs to the `breaker` category.

In contrast, from Figure 4b–d, we can see that FGREP appears to do a better job of capturing the terms' types and usage — unlike LSA, the `foods` have been separated from the term `ate`; the `predators` are now distinct from their `prey`; and the `paperwt` is now more closely associated with the `hitters` and `breakers` than it is with the `vase`. A related observation is that, while the ambiguous terms' relationships are somewhat unclear, FGREP was still able to separate them from amongst the other `animals`, `hitters`, `fragileobjs` and `possessions`. Unlike the previous experiment, the terms in this data set cannot be arranged into any meaningful hierarchies; rather the intrusions were calculated for each of the ten basic noun categories, the actions, and the catch-all category, 'thing'.

As with the first experiment, we can make two observations from the average reconstruction errors shown in Figure 5a: (1) the reductions were only consistent for the first 40,000–50,000 epochs — after that they became much more erratic; and (2) a *category's* average error seems to reflect the diver-

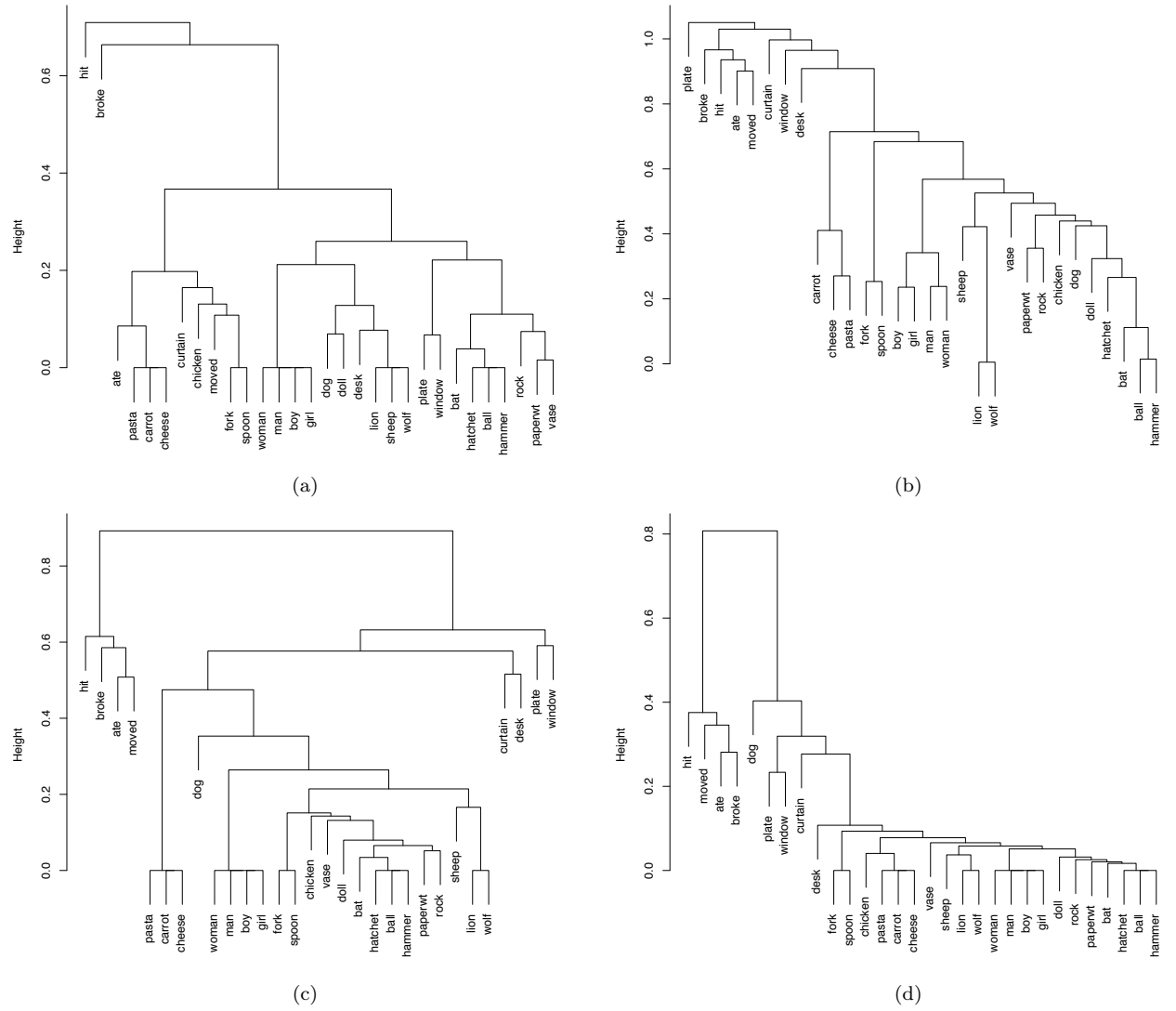


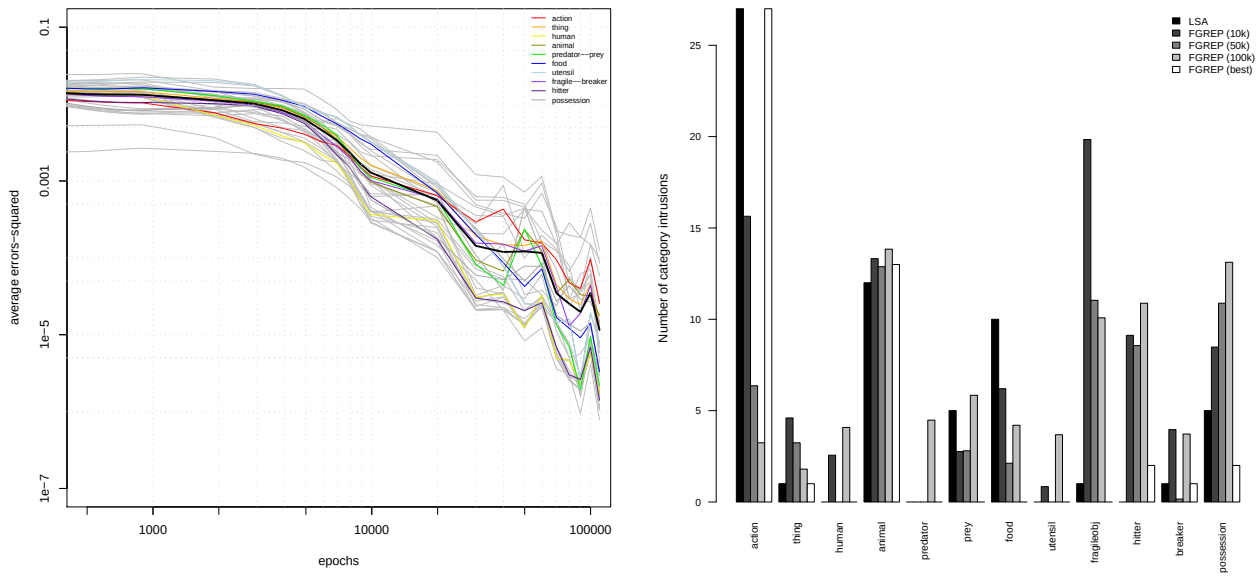
Figure 4: Hierarchical cluster plots based upon: (a) the distances as estimated by LSA; and the average distances, as estimated by FGREP, after (b) 10,000, (c) 50,000 and (d) 100,000 epochs, calculated over the 25 trials.

sity of its members' behaviour. Compare, for example, the categories `human` and `thing`: the former's members, which have amongst the lowest reconstruction errors, all exhibit the same behaviour, while the latter category consists of terms that assume ten distinct roles. The effects of these fluctuations can also be seen in Figure 5b; while the number of intrusions decreases as we go from 10,000 to 50,000 epochs, the numbers after 100,000 epochs are actually higher for nine of the twelve categories.

Looking at the category intrusions of Figure 5b and the errors associated with LSA, we can see that: (1) LSA was able to form distinct and disjoint clusters for each of the homogeneous categories — that is, the `human`, `utensil` and `hitter` categories whose members' behaviours are essentially identical; and (2) the categories with five or more intrusions each contained terms that played several different roles — for example, the concept `animal` includes terms that also belong to the `hitter`, `breaker`, `possession`, `predator`, `prey` and `food` categories.

FGREP's accuracy was less impressive in this experiment than it was for the first set of data. After 50,000 epochs, FGREP produced, on average, fewer intrusions than LSA for only three of the 12 categories, it was comparable for three, and actually yielded *more* intrusions for the six other categories; after 100,000 epochs, FGREP's average accuracy was worse than LSA's for 11 of the 12 categories. However, its *best* case performance after 100,000 epochs was more encouraging, yielding fewer intrusions for four categories, and producing comparable results for five other categories.

While the results of this experiment were less conclusive than those of the first, they demonstrate that FGREP is able to identify conceptual clusters from information about the entities' usage alone; furthermore, they show that, in the best case, FGREP is able to capture most of an entity's behaviour, even when it assumes several distinct roles.



(a) (b)

Figure 5: (a) The average per-term and per-category reconstruction errors-squared, calculated over the 25 trials; and (b) The category intrusions when the terms were grouped according to their LSA and FGREP measures of similarity. The first three FGREP results are the average number of category intrusions after 10,000, 50,000 and 100,000 epochs, calculated over the 25 trials; the final column shows the number of intrusions, after 100,000 epochs, for the best individual trial.

This experiment, and the differences between the average and best-case results, also highlighted two of FGREP's less desirable qualities: (1) that the quality of its clusters can be quite sensitive to the distributed representations' initial positions; and (2) that this dependence is exacerbated for terms that exhibit multiple types of behaviour.

Where FGREP *did* perform better than LSA, it seemed to do so by capturing terms' roles as well as their co-occurrences. For example, LSA was unable to differentiate between the `predators` and the `sheep` because it cannot take term order or syntax into account; FGREP, on the other hand, was able to distinguish between these two categories because they assume different roles, and thus occupy different slots, in Miikkulainen's data (the `predators` are agents while the `prey` are patients). Similar results were also observed for the `food` and `breaker` categories.

5. CONCLUSIONS AND FURTHER WORK

In this paper, we saw that FGREP is able to place terms into meaningful and intuitive clusters given either perceptual data or information about an entity's usage and interactions. While the quality of these clusters varied across trials and experiments, it is consistently able to: (1) arrange them into distinct and intuitive class hierarchies; (2) capture relationships and hierarchies that LSA cannot; and (3) distinguish between terms based upon the roles they assume.

These abilities, and at least some of FGREP's novelty as a technique, come from the fact that its fundamental unit of knowledge is the sentence, and that the order of terms *within* these sentences is both meaningful and well defined. Interestingly, the quality of its clusters seems to depend upon the number of distinct roles its members assume — i.e. the more complicated or ambiguous a term's usage, the harder it is to represent as a single point.

In the future, we will focus on characterising FGREP's typical behaviour, studying the stability of its learning rule, providing better support for terms that assume multiple roles, and extending FGREP to support discrete episodes and causal relationships; we will also use our analyses, both statistical and dynamical, to try to improve the average quality of its representations. Specific questions to be addressed include:

- How does the *initial* distribution of representations affect their later development, can we draw any conclusions about the 'typical' behaviour of the system, and can we develop any heuristics that improve the *average* quality of the representations?
- How stable is FGREP's learning rule? Treating the network as a dynamical system and the representations as particles, we can ask how stable are the representations to perturbations in their initial positions, how do the representations interact (and do the developed ones form concept-specific 'attractors'), and how does the trajectory of an ambiguous term's representation differ from those that participate in only one kind of relationship?
- Can we improve the system's performance, particularly for the ambiguous terms, by using *more* than one

particle per term? I.e. instead of forcing an ambiguous term to lie somewhere *between* each of its natural clusters, can we use multiple particles to develop unambiguous representations for each of its possible roles or use-cases?

- How is FGREP related to the auto-encoder, on the one hand, and a single-objective, multiple-particle form of gradient descent, on the other? Are there any optimisations that we can apply from these techniques to FGREP?
- What *other* kinds of information can it learn from? Thus far we have focused on declarative knowledge and case–role descriptions — can it be applied to other types of information, such as n-gram representations or the description of paths within a graph or semantic network?
- Can FGREP be extended to capture discrete episodes, and even causal relationships? Can this be achieved through simple extensions to the input vector, or will we need to make changes to the network's structure? and
- By relating sentence weights to the levels of activation within a semantic network, can we partition the network's sentence base into a set of smaller training sets? Put another way, can we implement a local, or context-specific, form of learning that increases the number of terms we can acquire by skipping over those entities and concepts that never co-occur?

6. REFERENCES

[1] Peter Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT Press, 2000.

[2] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Elsevier, 2nd edition, 2006.

[3] Thomas K Landauer and Susan Dumais. Latent Semantic Analysis. *Scholarpedia*, 3(11):4356, 2008.

[4] Thomas K Landauer, Peter W. Foltz, and Darrell Laham. An Introduction to Latent Semantic Analysis. *Discourse Processes*, 25:259–284, 1998.

[5] Risto Miikkulainen. *Subsymbolic Natural Language Processing: An Integrated Model of Scripts, Lexicon, and Memory*. MIT Press, 1993.

[6] Risto Miikkulainen and Michael G. Dyer. Forming Global Representations with Extended Backpropagation. In *IEEE International Conference on Neural Networks*, volume 1, pages 285–292, 1988.

[7] Timothy T. Rogers and James L. McClelland. *Semantic Cognition: A Parallel Distributed Processing Approach*. MIT Press, 2004.

[8] E. Rosch, C. B. Mervis, W. Gray, D. Johnson, and P. Boyes-Braem. Basic Objects in Natural Categories. *Cognitive Psychology*, 8:382–439, 1976.

[9] W. N. Venables, D. M. Smith, and the R Development Core Team. *An Introduction to R (Version 2.9.0)*. Available from `http://cran.r-project.org/doc/manuals/R-intro.pdf`, 2009.

Modelling Dynamical Systems with Recurrent Neural Networks

Alberto Dri and Cara MacNish
School of Computer Science &
Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009,
Australia.
email: {alberto, cara}@csse.uwa.edu.au

ABSTRACT

Dynamical systems are ubiquitous in science and nature, from the interwoven trajectories of planets, to the interconnected patterns of climate change, to the biological regulatory systems that sustain life. Developing computational models of these systems allows us to better understand them — both to *explain* how current states came about, and to *predict* future behaviour. This in turn helps us with both assessment or diagnosis, and treatment or controlling future system behaviour.

The traditional approach to modelling dynamical systems is to hand-craft differential equations in an attempt to match the observed data. In highly complex systems, which may have many parameters affecting behaviour, this is a difficult and time-consuming task, and while good models may be developed, there is often little assurance that the models optimally reflect the data. Machine learning techniques, on the other hand, seek to *automatically* adapt models to best fit the data.

Recurrent neural networks (RNNs) are a learning technique that extends the universal function approximation capabilities of neural networks with an internal memory or *state*. This makes the recurrent network an adaptive dynamical system, capable (at least conceptually) of learning to model a broad range of dynamical systems. In practice there are a range of issues relating to network structure, training, and sensitivity.

This paper discusses the above ideas, which will form part of the first author's PhD proposal. The paper gives a brief introduction to dynamical systems, what they are and why we wish to model them. It also discusses recurrent neural networks, their characteristics and most common applications. It explains why RNNs are particularly suitable for modelling dynamical systems. Finally, it indicates how the authors intend to investigate the potential of RNNs to develop, in collaboration with the Biology Computational Engineering Group, an adaptive model to help understand the biological process of bone remodelling [4].

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

Keywords: Recurrent Neural Networks, Dynamical Systems, Modelling Complex Systems, Machine Learning, Backpropagation Through Time

CR Classifications: B.6.3 [Software]: Artificual Intelligence

1. INTRODUCTION

The primary goal of my PhD research is to understand, apply and advance computational intelligence (CI) techniques for modelling dynamical systems.

I am dedicating particular attention to the study of RNNs, as this type of neural networks is capable of learning, has internal memory and can represent spatial and temporal behaviour. These characteristics make RNNs particularly suitable for modelling dynamical systems.

Dynamical systems have traditionally been modelled as systems of differential equations, which describe how a set of interdependent variables change over time. Developing new scientific theories and working out their equations is no simple work. Other than scientific insight, specific domain knowledge and intuition, this process involves a lot of patience and guesswork.

Adaptive modelling takes over where mathematical modelling leaves off, potentially stretching well beyond the mere construction of predictive and diagnostic models. By helping scientists and engineers to uncover hidden relationships amongst modelled objects, it leverages the process of theory building, paving the way towards scientific innovation and discovery.

The potential applications of such RNN-based models are numerous and diverse, ranging from modelling physical and biological phenomena to the construction of complex environmental, economic, financial and medical system models, to name a few.

At the University of Western Australia, in the School of Computer Science and Software Engineering (CSSE), a promising relationship between the Adaptive Systems Group and the Engineering Computational Biology Group (ECBG) started to form in June 2009. The idea is to use CI, which is a topic studied within the Adaptive Systems Group, to con-

solidate, improve and extend existing mathematical models developed by the ECBG to understand the process of bone remodelling.

In this paper I describe what I have learned so far about RNNs and dynamical systems. And then I describe how I intend to develop an RNN-based model for the bone remodelling process.

2. DYNAMICAL SYSTEMS

A dynamical system is characterized by:

- Having a state that changes over time.
- Having a set of deterministic rules describing all possible state transitions within the state space.
- Being highly dependent on initial conditions.

The archetypical example of a dynamical system is the solar system, whose state is given by the positions and relative velocities (and possibly several other properties one might consider relevant) of the sun, the planets and their satellites, and whose deterministic transition rules are given by Newton's laws of motion and also Newton's law of universal gravitation.

The solar system is the most popular example of a dynamical system probably because it is one of the simplest ones to explain. Most dynamical systems are quite complex, and intimately related to the notions of deterministic chaos and fractals. Other examples of dynamical systems include the weather, the stock market, a lakes ecosystem, the inner workings of a cell, and several other physical and biological systems. Life itself can be considered the dynamical system of ultimate complexity.

Because of their deterministic behaviour, and the smooth state transitions that take place over time, dynamical systems have traditionally been described by systems of differential equations. It has been suggested in [5] that a general framework to resolve dynamical systems can be given by following the system of equations:

$$\frac{dx_1}{dt} = f_1(x_1, x_2, \ldots, x_n) \tag{1}$$

$$\frac{dx_2}{dt} = f_2(x_1, x_2, \ldots, x_n) \tag{2}$$

$$\vdots$$

$$\frac{dx_n}{dt} = f_n(x_1, x_2, \ldots, x_n) \tag{3}$$

Each of the individual equations in the above system may be understood as the instantaneous variation of the state variable x_i being described as a function of all other state variables in the system, including itself.

By solving the system of equations with appropriate boundary conditions, we can "run" the system forward in time when the objective is to predict future states and create forecasts. This type of application is called predictive modelling. We can also run the system backwards in time when the objective is to determine which possible past states of the system may have led to the current state. This second type of application is called diagnostic modelling.

The difficulty with developing models with systems of differential equations is that scientists are on their own on the process of building theories, describing these theories in terms of mathematical equations and, finally, validating the equations against experimental observations. This becomes increasingly difficult as the system and equations grow in complexity.

3. RECURRENT NEURAL NETWORKS

Adaptive methods have been proposed [1, 2, 7] as alternatives to the use of differential equations for modelling complex dynamical systems. These methods consist of using computational approaches rather than mathematical equations in order to look for recurrent patterns on experimental data.

Thanks to their capability to learn and to represent internal state, RNNs have been heralded as a very effective way of modelling dynamic behaviour over time. In fact, it has been suggested that, given a sufficiently large number of processing units, neural networks are able to approximate mathematical functions to an arbitrary degree of accuracy [3]. This implies that supervised learning can be used to infer the behaviour of dynamical systems, provided that sufficient experimental data is available.

The basic idea consists of the following:

1. Experimental data is collected from the system to be modelled.
2. Supervised learning is used to train the RNN, which learns to approximate the experimental data up to the desired accuracy.
3. Optionally, a Finite State Machine (FSM) is extracted from the RNN [2].
4. The trained RNN (or the extracted FSM) is used as a model of the dynamical system.
5. The RNN can be kept in permanent training. By comparing the RNN model predictions to newly observed data, the network can continue to adapt as the modelled system changes or evolves over time.

In the following sub-sections I outline the benefits and drawbacks of modelling dynamical systems using RNNs.

3.1 Benefits of Modelling Dynamical Systems with RNNs

These are the benefits I found in modelling dynamical systems with RNNs:

- RNN-based models are able to improve their accuracy over time. This is a great advantage over systems of differential equations, which make rather static models. The only requirement for this feature to be implemented is that training data must be available for the continuous training of the network.
- RNN-based models can adapt to new circumstances and, therefore, are able to handle situations unforeseen at design time. Again, this is an advantage over systems of differential equations, which, as discussed, are static and must provide a full description of the system's behaviour at design time. RNN-based models are able to change their output as new experimental data offers new insights on the actual system's behaviour. Naturally this requires continuous re-training of the system as soon as new data becomes available.
- RNN-based models can remember the state they are currently in. Consequently the same set of input parameters can generate different results and cause different changes within the system's memory. This differs from feed-forward neural networks, which have no memory and therefore will generate the same output given the same set of input parameters. This characteristic of RNN-based models is of particular relevance for the modelling of dynamical systems, because dynamical systems have their state continuously changing over time.
- RNN-based models can remember previous input values, so that delay filters can be simulated. This is an important characteristic to be taken into consideration, should the model require the simulation of causes that have lasting effects. Take for example the need to simulate the moment of inertia of a turning aircraft, which has been accelerating in a given direction. The internal feedback mechanisms of a RNN have a lasting fading effect over its state variables.
- RNN-based models do not require, at design time, any domain specific information, nor do they require any intuitive insight on how the dynamical system might behave. The RNN-based model is able to derive all this information from the training data. The power of this unique feature is that it can be used in combination with already known information about the system to be modelled. In such way we can build models that can autonomously evolve to deal with situations completely unknown to scientists at the time the system is being designed.

3.2 Drawbacks of Modelling Dynamical Systems with RNNs

These are a few weaknesses of modelling dynamical systems with RNNs:

- RNN-based models require training through supervised learning and, therefore, the availability of experimental data becomes an essential aspect to be considered. In some situations, particularly in bio-medical engineering, experimental data is obtained through intrusive, often surgical processes, which for obvious reasons are to be avoided.
- A RNN training process may be a slow and tricky process. Learning multiple trajectories on complex, multidimensional state spaces involves continuous optimizations.
- RNNs are rather specialized objects — different RNNs must be trained for different problem domains. Additionally, different types of problems may require customization of a RNN architecture, topology and connectivity.
- RNNs are a lot more complex than feed forwards neural networks, both in terms of structure and algorithms. RNN training has often been described as problematic, as learning algorithms often cannot avoid being trapped in local minima. This leads to caution about their utility.

4. MODELLING THE BONE REMODELLING PROCESS

4.1 The currently existing mathematical model

Bone remodelling is the process of bone resorption and formation that continuously takes place during a vertebrates life. During bone remodelling, different types of cells at various stages of maturation interact with each other in complicated ways. The observable results of these complex bone cell iterations are changes in bone volume, replacing old with newly formed tissue and healing micro-fractures.

At CSSE, this emergent behaviour has been one of the subjects of study of the Engineering Computational Biology Group. With the objective of better understanding the bone remodelling process, they developed a cell population model, which is based on a system of differential equations. This model was implemented in MATLAB and their experiments and results were published in [4].

As discussed in several recent seminars in CSSE, the greatest advantage of having a bone cell model is that it is much simpler to perform experiments on the computer than on real biological systems. The model can be used not only to predict how the values of certain variables change over time, but also to help on the process of theory building, for example, by changing differentiation rates and other constants that determine how the bone remodelling process works.

4.2 The Construction of a RNN-based model

The idea of building a new, adaptive version of the bone model is to attempt to improve the model by allowing the model itself to derive the mathematical equations it needs, based on recurrent patterns in the experimental data.

As previously discussed, a significant amount of data is required to train a RNN. This could be the source of a problem, as a sufficiently large volume data is not available at the present moment for the bone remodelling process.

As a consequence, a "shaping" approach was agreed upon, in which the differential equations are initially used to guide the network close to a solution:

1. I assume that the current mathematical model *is* the dynamical system to be modelled.

2. I use the current mathematical model to generate as much training data as needed.

3. I use the generated training data to train a RNN using supervised learning and back propagation through time [6].

The outcome of the above scenario is an RNN-based model, which, at least in theory, is capable of making similar predictions to the currently existing mathematical model.

4.3 Making the RNN-based model better than the original model

Once an RNN-based approximation has been obtained, the next step is to attempt to improve on the original model through further training with the observed data. This requires the availability of additional training data, potentially containing information not taken into consideration while developing the equations of the original model.

5. CONCLUSIONS AND FURTHER WORK

I have been reading a lot about dynamical systems theory and about RNNs. I believe I now understand the complexities related to training an RNN and know how to use it to model a complex dynamical system. I believe I also developed a good understanding of the complexities involved in modelling dynamical systems.

I have been studying and experimenting with the MATLAB Neural Network Toolbox, which implements several types of neural networks, including RNNs. The MATLAB Neural Network Toolbox also implements a few versions of the back propagation through time training algorithm. The MATLAB neural network components are highly customizable, so that the software developer can create ad-hoc networks that fit their particular problem domain.

The next step in my research is to learn more about the bone remodelling process and about the implementation of the current model. Soon after that I can start putting together the adaptive model I have discussed in this paper. I am aware of the difficulties I am going to face, but optimistic about my goals and the results I want to achieve.

6. ACKNOWLEDGEMENTS

I would like to thank Dr. Peter Pivonka for kindly providing the MATLAB code that implements the mathematical version of the bone remodelling model.

7. REFERENCES

[1] Mikael Boden. A guide to recurrent neural networks and backpropagation. November 2001.

[2] Mike Casey. The dynamics of discrete-time computation, with application to recurrent neural networks and finite state machine extraction. *Neural Computation*, (8):1135–1178, 1996.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mat. Control Signals Systems*, 2(2):303–314, November 1989.

[4] Peter Pivonka et al. Model structure and control of bone remodeling: A theoretical study. *ELSEVIER*, 43:249–263, April 2008.

[5] Steven H. Strogatz. *Nonlinear Dynamics and Chaos*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1994.

[6] Paul J. Werbos. Backpropagation through time: What it does and how to do it. volume 78, October 1990.

[7] Stephen Wolfran. *A New Kind of Science*. Wolfram Media Inc., Champaign, IL, 2002.

Development of Lingual Differentiation in Child Speech: Emerging Trends

Sally Hodson
Department of Computer Science and Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia
sally@csse.uwa.edu.au

Cara MacNish
Department of Computer Science and Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia
cara@csse.uwa.edu.au

ABSTRACT
This paper summarises the results so far of a study investigating the development of independent tongue tip and tongue back control in the speech of pre- school aged children. Degree of differentiation between tongue tip and tongue back movements, known as lingual differentiation, is being measured using computer analysis of the frequency characteristics of 't' and 'k' sounds produced in conversational speech. Each participant's speech is recorded at three separate sessions over a six month interval in order to investigate speech development at an individual as well as group level. This paper presents the results obtained at the completion of the second round of speech recording. Two measures, centre of gravity, and skewness of plosive bursts are reported. The early results indicate that the degree of lingual differentiation is highly variable, at both the inter- and intra- participant levels. Comparison of older and younger participant groups suggests there may be an overall trend towards improvement in lingual differentiation with age, but further statistical investigation of the correlation between age and lingual differentiation is needed. Interpretation of the early findings, and future directions for the larger study are discussed.

Keywords
Signal processing, Speech, Phonetics

1. INTRODUCTION
The physical production of human speech may be modeled as a series of gestures – overlapping movements of the tongue, lips larynx, velum and jaw that result in a degree of constriction in the vocal tract [3,8]. The various speech sounds of a given language are produced through the use of gestures involving different places and degrees of constriction. A constriction, such as closure of the lips, can be achieved using a different organs, such as the upper lip, lower lip, or jaw, and may involve different rates of acceleration and deceleration of the organ in movement. These variables are influenced by the preceding and following speech sounds.

Child speech development involves a gradual improvement in speech clarity, related in part to the number of different speech sounds that the child can produce [15]. Some sounds, such as 'b' emerge in the infant's first babbling routines while others such as 'th' and 'r' can emerge as late as the early school years. Development can be described either in terms of the sounds observed by a listener, or in terms of the development of the underlying gestures.

Early babbling, such as 'ba-ba-ba' has been characterized as simply involving continuous voicing, combined with cyclical opening and closing movements of the jaw [1]. More sophisticated babbling patterns such as 'ba-da-ba-da' follow. This sequence still involves cyclical jaw movements, but while 'ba' involves lip contact, 'da' involves elevation of the whole tongue with the jaw, resulting in tongue palate contact [1]. Later, the child learns to move the tongue tip and body independently of each other, resulting in a 't' and 'k' contrast [6]. In this way, the process of speech sound development in children can be characterized as involving increased differentiation of gesture types

One notable characteristic of child speech, when compared to that of adults, is high variability [16]. Direct measurement of articulator movement has suggested that children show more variability in range and speed of gestures, and are more sensitive to coarticulatory effects – that is, influence by the gestures occurring before and after the gesture of interest [6]. Pre- school aged children show a greater degree of variability than school-aged children or adolescents [17,18]. However, the extent to which child speech is more variable than adults and the age at which speech patterns stabilize are still being debated in the literature. It appears that studies involving more complex speech productions, at the discourse and sentence level have more variability than less complex utterances such as single syllables [5, 20].

Naturalistic discourse may better capture variability and is more reflective of real-life development. Additionally, observing speech in an everyday context may be more reflective of real world system performance. However, the degree of experimental control is greatly reduced [13]. There is a greater risk of confounding factors affecting accuracy of the data. Some confounding factors in this instance might be speech rate, surrounding speech sounds, and levels of background noise. For this study it was decided that the benefits of observing speech production in a complex and realistic setting outweighed the risk of introducing confounding factors.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9-10 September 2009.

One challenge of conversational speech measurement is that it involves a complex combination of overlapping gestures, which are difficult to measure concurrently. There are 24 consonants in English, meaning there are 24 different types of vocal tract constriction that are linguistically meaningful in the English language. To investigate all of these variations at once is beyond the scope of this study.

For this reason, the contrast between only two consonants, 't' and 'k' is investigated. These two consonants differ in terms of place of constriction. While 't' in produced through the elevation of the tongue tip to the alveolar ridge (the section of the palate directly behind the front teeth) 'k' is produced by elevating the body of the tongue to make contact with the velum, or soft palate [12]. Both sounds are produced using ballistic movements involving rapid acceleration and deceleration and complete constriction of the vocal tract [11]. Complete constriction results in a build-up of air pressure behind the point of constriction, followed by a burst of air, known as a plosive burst, when the constriction is released.

The contrast between tongue tip and tongue body elevation is of particular interest to researchers of speech sound disorders. Previous research has found that many children with speech sound disorders do not demonstrate adequate lingual differentiation, that is, adequate independent control of the tongue tip and back [4,5,9]. For example, while an adult produces a 't' sound with only the tongue tip, and a 'k' sound with only the tongue back, some children with speech sound disorders have been found to elevate the entire tongue for both 't' and 'k' sounds. This phenomenon has been measured both directly, using sensors on the palate to measure tongue-palate contact, and indirectly, using speech signal analysis to infer the point of contact from the acoustic consequences.

A developmental trajectory approach will be used in this overall research project to track changes in lingual differentiation. Traditional approaches to measurement of child speech development use discrete measures, such as the achievement of specific speech sounds. In contrast, this study will focus on a continuous measure, the degree of differentiation between anterior and posterior tongue movements. This provides data in terms of a gradient, as opposed to the traditional milestone approach.

Following a pilot study by the author [10], spectral moment analysis of the burst spectra was deemed the most suitable measure of point of constriction for use in naturalistic speech, and in a developmental trajectory approach. The frequency characteristics of a plosive burst vary, depending on the associated point of constriction [12]. For example, as 't' is further towards the front of the mouth, the resonance chamber (that is, distance from constriction to end of the vocal tract) is much shorter, resulting in a higher frequency burst. The first and third spectral moments, centre of gravity and skewness, were selected as these best differentiate 't' and 'k' productions [4, 14].

This paper discusses two main questions relating to the overall project. The first aim is to investigate how lingual differentiation changes in pre -school aged children, compared to adults. The second question addressed is whether children show more variation in degree of lingual differentiation than adults. Answers to these questions have implications for understanding motor learning in children, as well as the validity of diagnostic categories. This paper discusses the early findings of a six month longitudinal study into the development of lingual differentiation in children.

2. METHOD

2.1 Participants

The participants are 32 children aged between 2 years, 7 months and 5 years 1 month at the time of the first data collection session, who were recruited from local childcare centres. No participants have received, or are currently receiving speech pathology services. Observation by the researcher, a practicing speech pathologist, did not reveal any evidence of undiagnosed speech or language disorders. Participants have no reported physical, intellectual, or language disabilities.

As the voice of the researcher is also recorded in the speech samples, the researcher's speech is also analysed to provide an adult comparison.

2.2 Assessment Task

Each participant will participate in three data collection sessions at three-monthly intervals. This means that data can be collected on individual patterns of development for each child, while investigating trends over a broader age range. The combination of longitudinal and cross-sectional data provides the clearest indication of developmental change within the time constraints of the project.

Each data collection session lasts for ten minutes, and involves tape recording of a conversation between the participant and the adult researcher. This conversation involves common everyday questions, such as "what did you do at daycare today?" and "what toys do you like to play with", along with discussion of pictures in a book, for example, "What's Grandad doing?" "Have you ever worked in the garden like Grandad?". There were no limitations on the topic or vocabulary used by participants, as the aim was to create a natural and flowing conversation, where the participant produced multiple word utterances.

2.3 Procedure

The conversation sessions were conducted onsite at the respective childcare facilities in a quiet room away from other children. The speech samples were recorded onto a Sony ICD-P620 digital recorder. They were converted to .wav files with a sampling rate of 44.1kHz. The speech samples were then analysed in Praat [2]. The onsets of all plosive bursts associated with 't' and 'k' productions were identified and marked by viewing the spectrographic record. Once the onset of each plosive burst was identified, scripts developed by the researcher were used to extract the first 20ms of each plosive burst, in accordance with [4], and the first and third spectral moments were calculated using an inbuilt function of Praat.

While data on all four spectral moments, and the highest peak in a long term average spectrum, were collected both with and without band pass filtering, only unfiltered first and third spectral moments are discussed in this paper, as these were the two measures that appeared to best differentiate 't' and 'k' productions for the adult participant on visual inspection of data.

As data was collected from four different childcare centres, and sometimes in different rooms within the centre, the level and nature of background noise is likely to vary between

environments. To address this issue, the speech of the adult interlocutor was analysed in each different environment. On attainment of appropriate software, a multivariate ANOVA will be used to compare values obtained for the three measures in each of the different environments. In the interim, a one-way ANOVA of the first spectral moment (centre of gravity) was used to investigate the impact of background noise. For this analysis 't' productions and 'k' productions were analyzed separately as they have different frequency characteristics and thus may be impacted differently by background noise. Results were not significant for t, ($F(5,142) = 1.494$, $p = 0.195$) or for k ($F(5,142) = 1.207$, $p = 0.307$). According to analysis so far, the frequency characteristics of plosive bursts do not appear to be significantly impacted by changes to recording environment.

3. RESULTS

For each recorded speech sample, average values for centre of gravity and skewness were calculated. Summary statistics for both /t/ and /k/ are shown in Table 1.

Table 1. Means and standard deviations for the first and third spectral moments of participants' 't' and 'k' productions.

	t		k	
	Mean	**Standard deviation**	**Mean**	**Standard deviation**
Number of productions	18.32	8.79	21.32	9.72
Centre of gravity (Hz)	3813.73	491.19	2958.21	262.70
Skewness	1.91	0.26	2.65	0.25

3.1 Do children show a lesser degree of lingual differentiation than adults?

At present, the difference between the means for 't' and 'k' is the only measure being employed to measure degree of differentiation. However, this measure considers centre of gravity and skewness separately. Alternative techniques are currently being investigated by the researcher for future analysis. Summary statistics for the mean distance between average 't' and 'k' values for each participant are shown in Table 2, compared to the adult interlocutor.

Table 2. Summary statistics for distance between average centre of gravity and skewness values for 't' and 'k' for both the participants and the adult interlocutor.

	Participants		Adult	
	Mean	**Standard deviation**	**Mean**	**Standard deviation**
ΔCentre of gravity (Hz)	855.52	494.44	1606.28	219.36
ΔSkewness	0.74	0.32	1.47	0.19

As there are 32 participants, as opposed to one adult measured over multiple occasions, it is unsurprising that the participant group has higher standard deviation values than the adult. The adult participant has a much greater difference between average values for 't' and 'k' productions on both measures. This suggests that the adult participant does, indeed, produce 't' and 'k'sounds with more different frequency characteristics, suggesting a greater degree of lingual differentiation. The statistical significance of this observation will be investigated using a single sample t-test.

3.2 Does the degree of lingual differentiation in child speech increase over time?

To investigate this question, all collected speech samples were ranked by age of participant at recording time and then divided into two groups. This resulted in the formation of a younger group and older group of children with an equal number of speech samples in each group. The younger group includes all speech samples from children aged 3 years, 10 months and younger (mean 3 years 5 months, standard deviation 4.2months) at the time of recording, while the older group includes all samples from children aged 3 years, 11 months and older (mean 4 years 4months, standard deviation 4.3months). There are three participants for whom their first two recordings are included in the younger group, and their final included in the older group, as they turned 3 years and 11 months in between the second and third recording sessions.

A two sample t-test was used to compare the degree of difference between 't' and 'k' measures for the younger and older groups. The results are summarized in Table 3.

Table 3. Summary statistics and alpha values for 2 sample t-test comparing distance between average centre of gravity and skewness values for 't' and 'k' for the younger and older participant groups.

	Younger group		Older group		
	Mean	**Standard deviation**	**Mean**	**Standard deviation**	**p**
ΔCentre of gravity (Hz)	763.6	452.3	1001.4	498.7	0.045
ΔSkewness	0.717	0.361	0.856	0.250	0.063

The older group produced 't' and 'k' sounds that were more different from each other than those produced by children in the younger group. This difference was found to be statistically significant at the .05 level for centre of gravity, but not for skewness. The results so far suggest that older pre-school children do show a greater degree of differentiation in their /t/ and /k/ productions than younger pre-school children, but less than adults. This may be indicative of a positive developmental trend.

To investigate this trend and the possible path of the developmental trajectory, the degree of lingual differentiation was plotted against participant age at sample collection. Figure 1

shows the results for centre of gravity and Figure 2 shows the results for skewness.

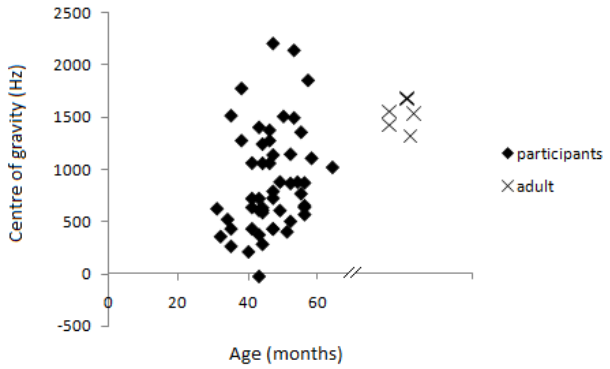


Figure 1. Difference between average centre of gravity values for 't' and 'k' for each participant as a function of age.

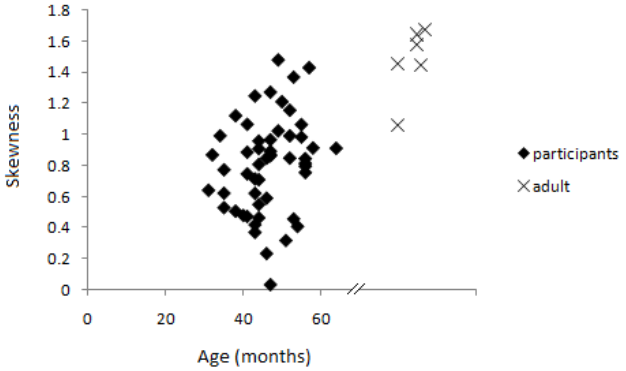


Figure 2. Difference between average skewness values for 't' and 'k' for each participant as a function of age.

While a slight positive trend may be present, a high degree of inter-participant variability is present, suggesting that lingual differentiation develops at different rates for different typically-developing children. Further statistical investigation is needed to ascertain the nature of trends and variability in this data.

Finally, the developmental trajectories of individual participants over the three sessions was also plotted. To date, only six participants have completed all three sessions with most others having completed two. The changes over the three sessions for these six participants are shown in Figures 3, and 4.

While the results of analysis of group data indicate that lingual differentiation is greater in older pre- school children and adults, individual children do not follow a smooth upward trajectory, instead appearing chaotic.

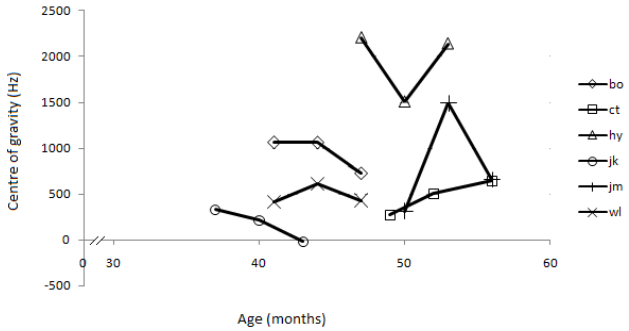


Figure 3. Changes in difference between average centre of gravity values for 't' and 'k' for individual participants over a six month period.

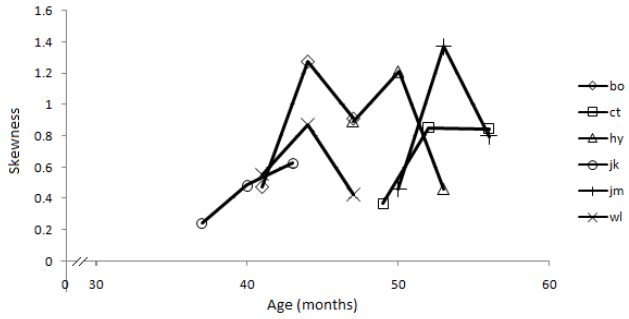


Figure 4. Changes in difference between average skewness values for 't' and 'k' for individual participants over a six month period.

4. DISCUSSION

Results so far suggest that typically developing pre- school aged children have a lesser degree of lingual differentiation than adults in their productions of 't' and 'k' in conversation. Furthermore, older pre- school aged children differentiate between the two sounds to a greater extent when compared to younger children. Visual inspection of plots of age against degree of lingual differentiation suggests that there may be a positive correlation between these two factors, although it is difficult to know for sure. If so, it suggests that lingual differentiation improves between that ages of 3 and 5 years of age. However, inter-speaker variability is high. Further statistical investigation is required to support this observation.

Additionally, it appears that some participants are achieving a degree of lingual differentiation that is equal to, or even exceeds the degree of lingual differentiation found in the adult interlocutor. This age range, from 3 to 5 years may be particularly critical in the development of lingual differentiation. Interestingly, this age is also associated with major improvements to speech clarity and to the number of different speech sounds produced correctly [15]. It is possible that lingual differentiation is correlated with phonological performance. This would make sense, as measures of lingual differentiation are measuring changes in the gestures that underlie speech production.

Although complete data sets are only currently available for six participants, it appears that individual trajectories may be chaotic, where the degree of differentiation may increase and decrease over the short term, while the long term trend is of improvement. It may be difficult to detect abnormality when the normal range is so variable.

The developmental trends and trajectories observed so far are consistent with a dynamic systems model of skill development in children [19]. Dynamic systems tend to show overall trends, in this case a general trend of increased lingual differentiation with maturity, while appearing chaotic at a micro-level. Additionally, from a dynamical systems perspective, changing from one attractor state to another, involves periods of increased variability in performance until the new state is established [19].

As the pre-school years are a time of rapid improvements in speech production, this could account for the high degree of variability observed in the underlying motor movements. It may be that variability is not just statistical 'noise', but a clinically useful measure in itself of underlying system reorganization. If so, this adds support to the use of naturalistic language sampling for assessment of clinical populations.

4.1 Future Directions

Collection of the third and final speech recordings for participants will be completed by October, 2009. Further statistical analysis will then be employed. Firstly, Linear Discriminant Analysis will be used to determine how much each of the four spectral moments contributes to the difference between 't' and 'k' classifications, and thus which measures will be useful for investigation.

Knowing exactly whether just one, or multiple, measures best capture differences between 't' and 'k' will determine the types of analysis used to provide measures of 'overlap' between 't' and 'k' burst spectra values. K-means cluster analysis or intrusion measures are being considered. Further statistical exploration of the relationship between age and lingual differentiation is also needed. Additionally, some exploration of factors influencing variability, such as coarticulation, could provide useful insights.

While data relating to children with speech sound disorders was not reported for this paper, it has been collected, and comparison of these participants to the group trends is also needed. This will involve comparisons of traditional perceptual measures of disorder with the findings of the speech signal analysis.

The results obtained so far are consistent with a speech development model in which children develop more differentiated gestures over time, and with the finding that children have highly variable speech gestures. While a comparison of younger participants, older participants and adults supports the hypothesis that lingual differentiation improves over time, inspection of the developmental changes in individual participants shows that the degree of lingual differentiation may increase and decrease over time with no observable pattern. The presence of an overall trend, but chaotic results at the individual level is consistent with dynamic systems theory.

5. REFERENCES

[1] Beckman, M. E., & Edwards, J. (2000). Lexical frequency effects on young children's imitative productions. In M. B. Broe & J. B. Pierrehumbert (Eds.), Acquisition and the lexicon (pp. 208-218). Cambridge: Cambridge University Press.

[2] Boersma, P., &Weenink, D. (2007). Praat: Doing phonetics by computer (Version 4.5.21).

[3] Browman, C. P., & Goldstein, L. (1992). Articulatory phonology: An overview. Phonetica, 49, 155-180.

[4] Edwards, J., Fourakis, M., Beckman, M. E., & Fox, R. A. (1999). Characterizing knowledge deficits in phonological disorders. Journal of Speech, Language, and Hearing Research, 42, 169.

[5] Gerosa, M., Giuliani, D., & Brugnara, F. (2007) Acoustic variability and automatic recognition of children's speech.Speech Communication, 49, 847-860.

[6] Gibbon, F. E. (1999).Undifferentiated Lingual Gestures in Children With Articulation/Phonological Disorders. Journal of Speech, Language, and Hearing Research, 42(2), 382-397.

[7] Goffman, L., & Smith, A. (1999). Development and phonetic differentiation of speech movement patterns. Journal of Experimental Psychology: Human Perception and Performance, 25, 649-660

[8] Goldstein, L. M., & Fowler, C. (2003). Articulatory phonology: a phonology for public language use. In A. S. Meyer & N. O. Schiller (Eds.), Phonetics and Phonology in Language Comprehension and Production: Differences and Similarities (pp. 159-207): Mouton de Gruyter.

[9] Hodson, S. L., & Hird, K. (2007). Gestural phonology in the assessment of paediatric speech sound disorders. Paper presented at the Speech Pathology Australia 2007 National Conference, Sydney, Australia.

[10] Hodson, S.L. (2008) Acoustic Measures of Development of Lingual Differentiation: A Pilot Study. Paper presented at the 2008 Reflecting Connections Conference. Auckland: New Zealand.

[11] Kent, R. D. (1992). The biology of phonological development. In C. A. Ferguson & L. Menn (Eds.), Phonological Development: Models, Research Implications (pp. 65-90). Timonium: York Press.

[12] Kent, R. D. (1997b). The Speech Sciences. San Diego: Singular.

[13] Nagy. N. 2006. Experimental methods for study of linguistic variation. In K. Brown, (Ed.)Encyclopedia of Language & Linguistics, (2nd ed., Vol 4). Oxford: Elsevier.

[14] Nittrouer, S. (1995). Children learn separate aspects of speech production at different rates: Evidence from spectral moments. Journal of the Acoustical Society of America, 97, 520-530.

[15] Owens, R.E. (2001). Language development: An introduction.(5th ed.). Boston: Allyn and Bacon.

[16] Sharkey, S.G.,& Folkins, J.W. (1985). Variability of lip and jaw movements in children and adults: Implications for the development of speech motor control. Journal of Speech and Hearing Research, 28, 8-15.

[17] Smith, A., (2006).Speech motor development: Integrating muscles, movements, and linguistic units. Journal of Communication Disorders, 39, 331-349.

[18] Smith, A., & Zelaznik, H. (2004). The development of functional synergies for speech motor coordination in childhood and adolescence. Developmental Psychobiology, 45, 22–33.

[19] Smith, L. B., & Thelen, E. (Eds.). (1993). A dynamic systems approach to development. Cambridge: MIT Press.

[20] Stathopoulos, E.T. (1995). Variability revisited: an acoustic, aerodynamic, and respiratory kinematic comparison of children and adults during speech. Journal of Phonetics, 23, 67-80.

An Improved Approach for Local 3D Feature Matching for Ear Recognition

Syed M. Shamsul Islam and Rowan Davies
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {shams,rowan}@csse.uwa.edu.au

ABSTRACT

Local features are significantly robust to occlusions and other visibility variations as well as minor deformations. However, human recognition via ear biometrics using local features involves some distinctive challenges. Due to the complex geometry and the high level of similarity between two sets of ears, matching local features produces large numbers of false feature matches. To increase recognition performance, we propose to eliminate some incorrect matches using a simple form of geometric consistency, and some associated similarity measures. We obtain an improvement from 81.60% to 92.77% in rank-1 ear identification on the University of Notre Dame Biometric Database, the largest publicly available profile database from the University of Notre Dame with 415 subjects.

1. INTRODUCTION

The performance of a biometric recognition system greatly depends on the representation of the underlying distinctive features and the algorithm for matching those features. Due to ease of computation various global features has been used in biometric systems. However, global features are often not robust to variations in observation conditions, including the presence of occlusion and deformations. In search of more robust features under these variations, researchers have proposed using local features.

Among 2D local features SIFT (scale invariant feature transform) [10] and its variants are used in many biometric applications [9, 13, 1, 11]. Also Guo and Xu [6] proposed using the Local Similarity Binary Pattern (LSBP) and Local Binary Pattern (LBP) for ear data representation and matching. However, 2D data has many inherent problems compared to its 3D counter part including sensitivity to the use of cosmetics, clothing and other decorations. Recently, Mian et al. [12] proposed the Local 3D feature (L3DF) inspired by 2D SIFT and found to be very effective for face representation and recognition. Islam et al. [8] also found it to be very fast and somewhat robust for ear recognition.

Among the biometric traits, ears and faces are considered to be most suitable for non-intrusive biometric recognition. However, they are also highly similar among themselves [3]. Therefore, using local features for these two biometric traits (especially for ears) produces many false or incorrect matches for a gallery-probe pair. In this paper, we find initial feature matches using L3DFs with simple metrics (such as RMS distance between features) similar to Mian et al. [12]. However, we propose using a geometric consistency check to filter out some of the incorrect feature matches in an additional round of matching. The consistency of feature match is judged from the difference between the implied distances on the probe and gallery to a particular match from the first round. This match is chosen to maximize the consistency in the first round. We also develop a similarity measure based on this consistency by computing the proportion of consistent distances in the first round. Finally, we combine this additional measure with the proportion of consistent rotations and mean distance error of the selected features computed as we previously proposed in [8, 7]. Experiments with different datasets prove the effectiveness of our technique via a considerable improvement in the recognition results.

The paper is organized as follows. The proposed approach for refining matching is described in Section 3 after describing the background and motivation in Section 2. The results obtained are reported and discussed in Section 4 and compared with other approaches in Section 5. Section 6 concludes with some future research directions.

2. BACKGROUND AND MOTIVATION

The local 3D feature that we use to demonstrate our approach can be depicted as a 3D surface constructed using data points within a sphere of radius r_1 centered at location p. Fig. 1 shows a local 3D feature surface extracted from an ear.

As described in [8], a 20×20 grid of heights is approximated using the surface points of a 3D surface feature. The grid is aligned following the axes in the Principal Component Analysis (PCA). The similarity between two features is calculated as the Root Mean Square (RMS) distance between corresponding heights on this grid. The similarity is computed for each probe feature location to all the gallery feature locations excluding those being located more than a threshold away. (As previously, this threshold is empirically chosen as 45mm in the case of ear matching since our automatic ear detection procedure does not precisely locate the ear.) The gallery feature that corresponds to the minimum distance is considered as the matching feature for that particular probe feature as previously.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9--10 September 2009.

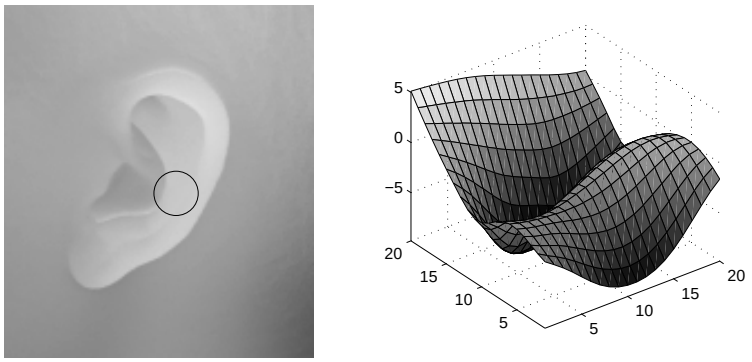


Figure 1: Example of a 3D local surface (right image) and the region from which it is extracted (left image, marked with a circle) [8].

The starting point for the current work was the observation that for a corresponding probe and gallery the set of feature matches produced by the above technique generally contains a large proportion of incorrect matches, and that the correct matches must be geometrically consistent, while the incorrect matches are unlikely to be. In contrast, for a non-corresponding probe and gallery, it is unlikely that there will be a large set of features that match well and are geometrically consistent. Here geometrically consistent means that there is a rigid transformation that maps the probe feature locations to the corresponding gallery feature locations.

While it seems possible to check full geometric consistency during the matching process, by constructing rigid transformations from sets of matched points using Random Sample Consensus (RANSAC) [5] or a variant such as Progressive Sample Consensus (PROSAC) [4], we chose instead to stick close to the original local 3D feature matching algorithm of Mian et al. [12] in order to build on its demonstrated strengths.

3. PROPOSED REFINEMENT TECHNIQUE

In this section, at first we describe how distance consistency can be used to filter out incorrect matches. Then, we describe the derivation of two similarity measures to be used in addition to the feature similarity.

3.1 Computation of Distance Consistency

In order to discard the incorrect matches, we choose to add a second round of feature matching each time a probe is compared with a gallery that uses geometric consistency based on information extracted from the feature matches generated by the first round. The first round of feature matching is done just as described in Section 2, and we use the matches generated to identify a subset that are most geometrically consistent.

For simplicity, we measure geometric consistency of a feature match by counting how many of the other feature matches from the first round yield consistent distances on the probe and gallery. More precisely, for a match with locations p_i, g_i we count how many other match locations p_j, g_j satisfy:

$$||p_i - p_j| - |g_i - g_j|| < d_{key} + \kappa\sqrt{|p_i - p_j|}$$

Here the threshold includes a term to allow for the spacing between candidate key points, as well as one that grows with the square root of the actual probe distance $|p_i - p_j|$ to account for minor deformations and measurement errors.

To exploit geometric consistency quickly and simply, we just find the match from the first round which is most "distance-consistent" according to this measure. Then, in the second round, we only allow feature matches that are distance-consistent with this match - thus for each probe feature we find the best matching gallery feature that is distance consistent with the chosen match.

We also compute the ratio of the maximum distance consistency to the total number of matches found in the first round of matching and use that as a similarity measure, proportion of consistent distances (λ).

3.2 Computation of Rotation Consistency

Similar to Islam et al. [8, 7], we also compute a similarity measure based on the consistency of the rotations implied by the feature matches. Each feature match implies a certain rotation between the probe and gallery, since we store the rotation matrix used to create the probe feature from the probe (calculated using PCA), and similarly for the gallery feature, and we assume that the match occurs because the features have been aligned in the same way and come from corresponding points. We can thus calculate the implied rotation from probe to gallery as $R_g^{-1}R_p$ (or $R_g^t R_p$) where R_p and R_g are the rotations used for the probe and gallery features.

We calculate these rotations for all feature matches, and then for each we determine the count of how many of the other rotations it is consistent with. Consistency between two rotations R_1 and R_2 is determined by finding the angle between them, i.e., the angle of the rotation $R_1^{-1}R_2$ (around the appropriate axis of rotation). When the angle is less than 10° we consider two rotations consistent. We choose the rotation that is consistent with the largest number of other matches, and then we use the proportion of matches consistent with this (α) as a similarity measure. As we shall see in Section 4, this measure becomes the strongest among the other measures used prior to applying Iterative Closest Point (ICP) algorithm in our ear recognition experiments.

Fig. 2 illustrates an example of the correspondences between

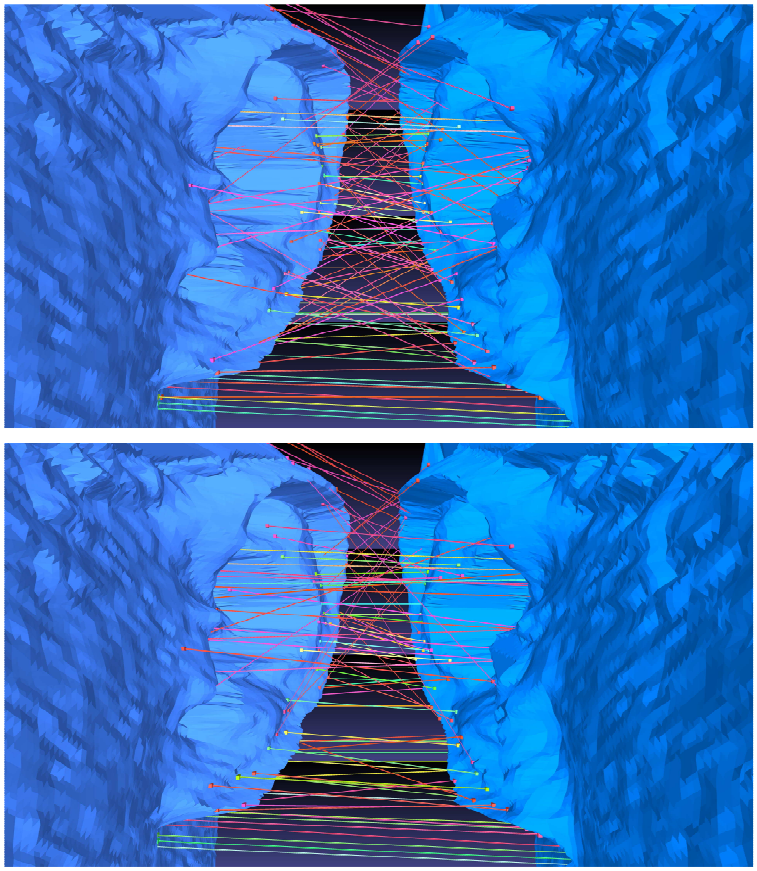


Figure 2: Feature correspondences before (top image) and after (bottom image) filtering with geometric consistency (best seen in color).

the features of a probe image and the corresponding gallery image (mirrored in the z direction) before (top image) and after (the bottom image) the geometric consistency check is performed. The green channel (brighter in the black and white) indicate the amount of rotational consistency for each match. It should be clear that a good proportion of these matches involve corresponding parts of the two ear images. We also observe that a large number of incorrect matches are eliminated after applying the geometric consistency check.

3.3 Final Similarity Measures

When the second round is complete for a particular probe and gallery, we use the mean error of the feature matches (γ) in the second round, proportion of consistent distances (λ) and the proportion of consistent rotations (α) from the second round to measure the closeness of the match. Similar to [8], we perform min-max normalization of the similarity measures and compute a weight factor (η) for each of the measures as the ratio of the difference of the minimum value from the mean to that of the second minimum value from the mean of that similarity measure. We compute the weighted sum as follows:

$$\epsilon = \eta_f \gamma + \eta_r(1-\alpha) + \eta_d(1-\lambda)$$

Based on the above score, we sort the candidate gallery images and apply coarse alignment and final ICP only on the minimal rectangular area of the best 20 candidates to make the final decision of matching.

For coarse alignment, we use the translation (t) corresponding to the maximum distance consistency and the rotation (R) corresponding to the largest cluster of consistent rotations as follows:

$$d_p{}' = Rd_p + t;$$

where, d_p and $d_p{}'$ are the probe point coordinates before and after coarse alignment.

4. RESULT AND DISCUSSION

To evaluate the performance of our refined matching on ear biometrics, we have used Collection F and J from the University of Notre Dame Profile Biometrics Database [14, 15]. For different experiments, we have sub-divided these two datasets into the following two subsets. In dataset A, the earliest 302 images of the UND Collection F are used as gallery and the latest 302 images are used as probe (there is a time lapse of 17.7 weeks on average between the earliest and latest images). The entire Collection J of the UND database is considered as dataset B which includes 415 earliest images as the gallery and another 415 images as probes. However, since the ear in one of the profile images was not automatically detected, the number of probes included in recognition experiments for this dataset is 414. We also tested our algorithm on 100 profile images from 50 sub-

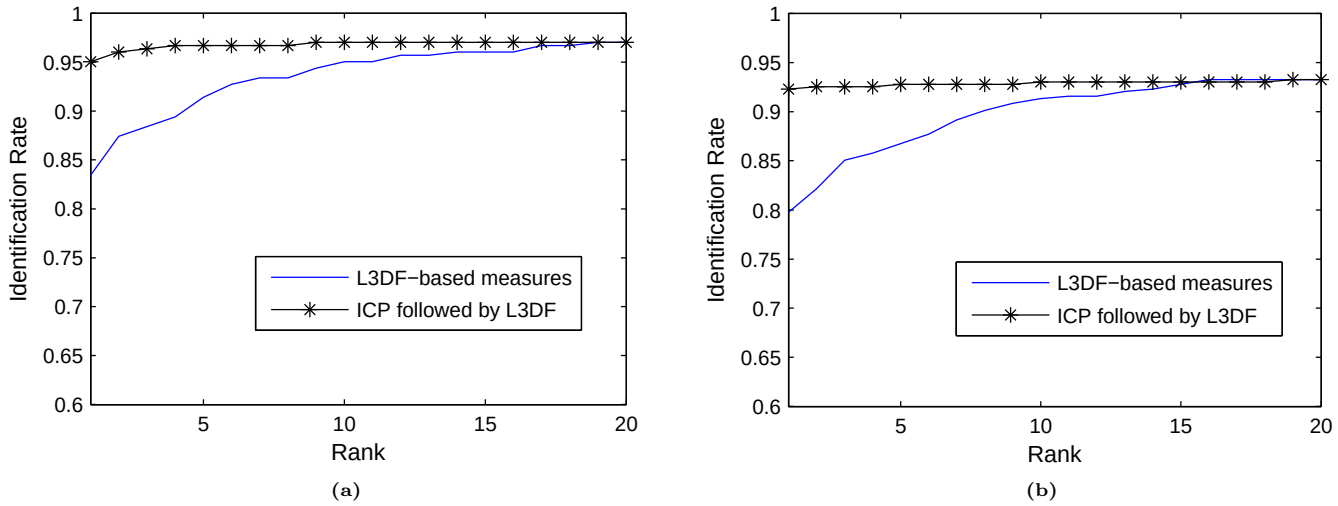


Figure 3: Indentification results using geometric consistency: (a) on dataset A (b) on dataset B.

jects each having images with and without ear-phones on (considered as dataset C). The images were collected at the University of Western Australia using a Minolta Vivid 910 range scanner in low resolution mode. There are significant data losses in the ear-pit regions of the images. Images without ear-phones are included in the gallery and others in the probe dataset.

On dataset A, we perform two experiments. In the first experiment, we use feature errors and proportion of consistent rotations for initial matching and only the rotation for coarse alignment. We then apply the final ICP on the minimal datapoints (as described in Section 3) of the gallery and the probe. This yields 93.71% rank-1 recognition for ear biometrics. In the second experiment, we use second stage of matching with distance consistency applied and use both proportion of consistent distances and rotations for initial matching as well as coarse alignment prior to application of final ICP on the best 20 gallery candidates. These changes improve the recognition rate to 95.03% which confirms the significance of using the geometric consistencies (see Fig. 3(a)).

On dataset B, without the geometric consistency check we obtain 71.57% and 81.60% rank-1 ear identification rate before and after using ICP on the best 20 gallery candidates sorted by L3DF-based measures. However, the results improve to 79.76% and 92.77% respectively when using our geometric consistency checks and measures. The improved results are shown in Fig. 3(b).

Results on dataset C and a summary of the results on other datasets with or without using geometric consistency measures are shown in Table 1. Matching time recorded are for a Matlab implementation of our algorithms using unoptimized Matlab codes on a Core 2 Quad 9550, 2.83GHz machine. Although, using geometric consistency measures increases the computational time little bit, it significantly improves the accuracy of the system especially for the larger datasets.

The strength of the proportion of consistent rotations ($propRot$) measure over other similarity measures is illustrated in Fig. 4 for an identification scenario on dataset B. In the figure, the mean error of feature matches and the proportion of consistent distances are indicated by $errL3DF$ and $propDist$ respectively.

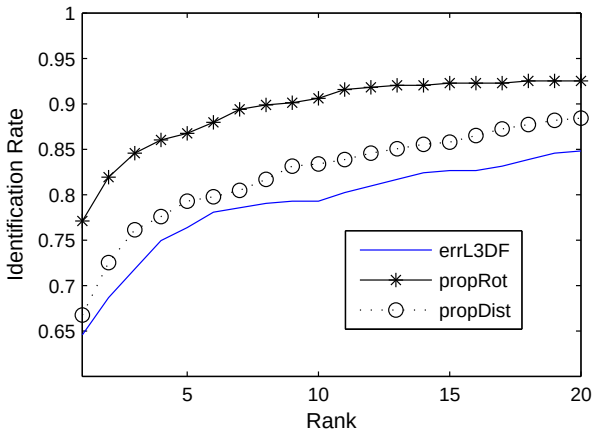


Figure 4: Comparing identification performance of different geometric consistency measures (on dataset B)

5. COMPARATIVE STUDY

Islam et al. [8] reported 84% rank-1 ear recognition result using first stage of feature matching along with the proportion of best rotations on the first 100 images of Collection F of the UND database. For the same dataset using distance consistency and proportion of consistent distances, we have obtained 92% rank-1 ear recognition.

Using distance consistency, we have obtained 9% improved identification and 8% improved verification result for ear recognition compared to that reported in [7].

Mian et al. [12] used edge error and node error on the graphs constructed from the keypoints of the matching local features. Although this roughly measures the distance consis-

Table 1: Performance variations for using geometric consistency measures.

Approach	Rank-1 Identification rate (%)				Avg. Matching time (s)
	dataset C (50-50)	dataset B (100-100)	dataset B (302-302)	dataset A (415-415)	
L3DF without geometric consistency	86	89	76.82	71.57	0.04
L3DF using geometric consistency	88	92	83.44	79.76	0.06
ICP and L3DF without geometric consistency	96	98	93.71	81.60	2.43
ICP and L3DF using geometric consistency	98	98	95.03	92.77	2.28

tency between matched features, it is less reliable when there are many bad matches, and therefore, the author did not use the approach for filtering the incorrect matches, rather as a similarity measure only. For the dataset used in [7], we obtain 69.13% recognition accuracy for ear using feature errors and graph errors. For the same dataset corresponding results with our distance consistency and proportion of rotations is 79.74%.

Chan and Bhanu [2] use geometric constraints similar to ours on their Local Feature Patches. However, they do not use an explicit second round of matching.

6. CONCLUSION AND FUTURE WORK

Our results indicate that this simple technique yields worthwhile gains. It is also seems very likely that we could exploit geometric consistency further. For example, instead of just choosing the most distance-consistent match, we could choose a whole set of them that are mutually distance consistent. Then, provided that we have at least three matches, we could calculate a rigid transformation via least square error and use this to directly map probe locations to the required gallery locations (modulo the threshold). We intend to do this in future work, and further gains seem likely.

Acknowledgment

This research is sponsored by ARC grants DP0664228 and LE0775672. The authors acknowledge the use of the UND Biometrics database with profile images and the FRGC v2 face database for ear and face recognition. They also like to thank M. Bennamoun for his valuable input, A. Mian for his 3D face normalization code and D'Errico for his surface fitting code.

7. REFERENCES

[1] J.D. Bustard and M.S. Nixon. Robust 2D Ear Registration and Recognition Based on SIFT Point Matching. *In Proc. of 2nd IEEE International Conference on Biometrics: Theory, Applications and Systems, 2008. BTAS 2008*, pages 1–6, 2008.

[2] Hui Chen and B. Bhanu. Human Ear Recognition in 3D. *IEEE Trans. PAMI*, 29(4):718–737, 2007.

[3] Hui Chen and B. Bhanu. Efficient Recognition of Highly Similar 3D Objects in Range Images. *IEEE Trans. PAMI*, 31(1):172–179, 2009.

[4] O. Chum and J. Matas. Matching with PROSAC - Progressive Sample Consensus. *In Proc. of the CVPR'05*, 1:220–226, 2004.

[5] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Comm. of the ACM*, 24:381–395, 1981.

[6] Yimo Guo and Zhengguang Xu. Ear Recognition Using a New Local Matching Approach. *In Proc. of the 15th IEEE International Conference on Image Processing, ICIP'08*, pages 289–292, 2008.

[7] S.M.S. Islam, M. Bennamoun, A. Mian, and R. Davies. Score Level Fusion of Ear and Face Local 3D Features for Fast and Expression-invariant Human Recognition. *M. Kamel and A. Campilho (Eds.): ICIAR 2009, LNCS 5627, Springer, Heidelberg*, pages 387–396, 2009.

[8] S.M.S. Islam, R. Davies, A. Mian, and M. Bennamoun. A Fast and Fully Automatic Ear Recognition Approach Based on 3D Local Surface Features. *J. Blanc-Talon et al. (Eds.): ACIVS 2008, LNCS 5259, Springer, Heidelberg*, pages 1081–1092, 2008.

[9] Yan Ke and R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. *In Proc. of the CVPR'04*, 2:506–513, 2004.

[10] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. Journal of Computer Vision*, 60:91–110, 2004.

[11] A. Majumdar and R. K. Ward. Discriminative SIFT Features for Face Recognition. *In Proc. of Canadian Conference on Electrical and Computer Engineering, 2009. CCECE '09*, pages 27–30, 2009.

[12] A.S. Mian, M. Bennamoun, and R. Owens. Keypoint Detection and Local Feature Matching for Textured 3D Face Recognition. *International Journal of Computer Vision*, 79(1):1–12, 2008.

[13] Jae-Han Park, Kyung-Wook Park, Seung-Ho Baeg, and Moon-Hong Baeg. $\pi - SIFT$: A photometric and Scale Invariant Feature Transform. *In Proc. of the 19th International Conference on Pattern Recognition, 2008. ICPR 2008*, 2:1–4, 2008.

[14] University of Notre Dame Biometrics Database. http://www.nd.edu/∼cvrl/UNDBiometricsDatabase.html, 2005.

[15] P. Yan and K. W. Bowyer. Biometric Recognition Using 3D Ear Shape. *IEEE Trans. PAMI*, 29(8):1297–1308, 2007.

Fragments of RoCTL* Obligation, Axioms and Soundness

John M^{c}Cabe-Dansted, Tim French, Mark Reynolds
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {john,mark,tim}@csse.uwa.edu.au

ABSTRACT

RoCTL* was proposed to model robustness in concurrent systems. RoCTL* extended CTL* with the addition of Obligatory and Robustly operators, which quantify over paths that are failure-free and paths with one more failure respectively. This this paper gives an axiomatisation for all the operators of RoCTL* with the exception of the Until operator; this fragment is able to express similar contrary-to-duty obligations to the full RoCTL* logic. We call these formal systems OA and NOA▲ respectively.

Keywords: Axioms, Deontic, Temporal, Modal, Logic, Full Computation Tree Logic, RoCTL*

1. INTRODUCTION

The Robust Full Computation Tree Logic (RoCTL*) [7] is an extension of CTL* introduced to represent issues relating to robustness and reliability in systems. It does this by adding an Obligatory operator and a Robustly operator. The Obligatory operator specifies how the systems should behave by quantifying over paths in which no failures occur. The Robustly operator specifies that something must be true on the current path and similar paths that "deviate" from the current path, having at most one more failure occurring. This notation allows phrases such as "even with n additional failures" to be built up by chaining n simple unary Robustly operators together.

The RoCTL* Obligatory operator is similar to the Obligatory operator in Standard Deontic Logic (SDL), although in RoCTL* the operator quantifies over paths rather than worlds. SDL has many paradoxes. Some of these, such as the "Gentle Murderer" paradox spring from the inadequacy of SDL for dealing with obligations caused by acting contrary to duty such as "If you murder, you must murder gently". Contrary-to-Duty (CtD) obligations are important for modeling a robust system, as it is often important to state that the system should achieve some goal and also that if it fails it should in some way recover from the failure. RoCTL* can represent CtD obligations by specifying that the agent must ensure that the CtD obligation is met even if a failure occurs. For further discussion of CtD obligations and motivations for RoCTL*, see [7]. The Obligatory operator, as well as some uses of the Robustly operator, are easy to translate into CTL* [12].

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

When RoCTL* was originally proposed [7], it had two accessibility relations, a success and failure transition. However we may equivalently define RoCTL* with a single accessibility relation if we add a violation atom **v** to indicate that the previous transition was a failure transition (this new definition was first used in [12]). Under this definition, the RoCTL* models are CTL models, albeit with a special violation atom not used in RoCTL* formulas.

The addition of the Robustly operator and temporal operators to Deontic logic allows RoCTL* to deal with Contrary-to-Duty obligations. SDL is able to distinguish what ought to be true from what is true, but is unable to specify obligations that come into force only when we behave incorrectly. For example, SDL is inadequate to represent the obligation "if you murder, you must murder gently" [6]. The addition of temporal operators to Deontic logic allows us to specify correct responses to failures that have occurred in the past [20]; RoCTL* follows this approach. However, this approach alone is not sufficient [20] to represent obligations such as "You must assist your neighbour, and you must warn them iff you will not assist them". In RoCTL* these obligations can be represented if the obligation to warn your neighbour is robust but the obligation to assist them is not. Contrary-to-Duty obligations have interesting effects on RoCTL* logic. For example, the statement "it is obligatory for ϕ to be true at the next step" neither implies nor is implied by the statement "At the next step it is obligatory for ϕ to be true".

Other approaches to dealing with Contrary-to-Duty obligations exist. Defeasible logic is often used [13], and logics of agency, such as STIT [3], can be useful as they can allow obligations to be conditional on the agent's ability to carry out the obligation.

The paper that introduced the RoCTL* logic [7] provided some examples of robust systems that can be effectively represented in RoCTL*. It is easy to solve the coordinated attack problem if our protocol is allowed to assume that only n messages will be lost. The logic may also be useful to represent the resilience of some economy to temporary failures to acquire or send some resource. For example, a remote mining colony may have interacting requirements for communications, food, electricity and fuel. RoCTL* may be more suitable than Resource Logics (see e.g. [5]) for representing systems where a failure may cause a resource to become temporarily unavailable.

A number of other extensions of temporal logics have been proposed to deal with Deontic or Robustness issues [4, 11, 8, 1, 19]. Each of these logics are substantially different from RoCTL*. Some of these logics are designed specifically to deal with deadlines [4, 8]. An Agent Communication Language was formed by adding Deontic and other modal operators to CTL [19]; this language does not explicitly deal with robustness or failures. Hansson and Johnsson [8] proposed an extension of CTL to deal with reliability. However, as well as being intended to deal with deadlines, their logic reasons about reliability using probabilities rather than numbers of failures, and their paper does not contain any discussion of the relationship of their logic to Deontic logics. Like our embeddings into QCTL* [7] and CTL* [14], Aldewereld et al. [1] uses a Viol atom to represent failure. However, their logic also uses probability instead of failure counts and is thus suited to a different class of problems than RoCTL*. None of these logics appear to have an operator that is substantially similar to the Robustly operator of RoCTL*.

Diagnosis problems in control theory [10, 2] also deals with failures of systems. Diagnosis is in some sense the dual of the purpose of the RoCTL* logic, as diagnosis requires that failure cause something (detection of the failure) whereas robustness involves showing that failure will *not* cause something.

It is known that every RoCTL* statement can be expressed in CTL*, however some statements can be expressed much more concisely in RoCTL* than CTL* [14]. We can reduce RoCTL* formulas to QCTL* formulas of linear length [7]; however, QCTL* is not elementary. There is a tableau for the RoCTL*-like logic RoBCTL* [15]; however, this tableau is also non-elementary in the worst case. It unlikely that RoCTL* has an elementary decision procedure, as it would be the first logic on tree-like models without an elementary translation into tree automata [14]. Although automated proof generation is likely to perform poorly in the worst case, proof systems based around axioms tend to focus on producing clean proofs of particular theorems rather than worst case performance. This gives a motivation for research into axiomatisations for RoCTL*. Additionally, by giving axioms for the RoCTL* operators, this gives further understanding as to how RoCTL* relates to other modal logics.

In this paper we have not considered the Until operator. The Next operator is enough to model interactions between time and the Obligatory and Robustly operators, including the contrary-to-duty obligations mentioned above. Given that there is a full axiomatisation of CTL*[18], including the Until operator, finding a full axiomatisation of RoCTL* seems feasible. However axiomatisations of smaller fragments of RoCTL* are also interesting. In the expanded tech report [17] we have given an axiomatisation of just the Obligatory and All paths operators, as examining axiomatisations of the path-quantifiers without time provides better understanding of the interactions between those operators. Indeed, we found it easier to axiomatise the Robustly operator together with the Next operator, as we can use the Next operator to simplify the axioms for the Robustly operator. We do not currently have an axiomatisation for the Robustly operator on its own, the search for such an axiomatisation may lead to a better understanding of this operator.

2. ROCTL* LOGIC

The RoCTL* logic has two definitions, each of which have the same validities. The original definition [16] had two accessibility relations, a "success" relation and a "failure" relation. When comparing the expressivity of RoCTL* with CTL*, it is convenient to instead use a "Viol" atom and a single accessibility relation [14]. In this paper we use the original definition with two accessibility relations.

We may represent the statement "Even if n or fewer unexpected events occur" by chaining together n instances of the $\blacktriangle$ operator.

The RoCTL* Logic has a set $\mathcal{V}$ of atomic propositions that we call variables. Where p varies over $\mathcal{V}$, the formulas of RoCTL* are defined by the abstract syntax

$$\phi : \quad = \quad \top \,|\, p \,|\, \neg\phi \,|\, (\phi \wedge \phi) \,|\, (\phi U \phi) \,|\, N\phi \,|\, A\phi \,|\, O\phi \,|\, \blacktriangle\phi \ .$$

The $\top$, $\neg$, $\wedge$, N, U and A are the familiar "true", "not", "and", "next", "until" and "all paths" operators from CTL. The abbreviations $\bot$, $\vee$, F, G, W, E $\rightarrow$ and $\leftrightarrow$ are defined as in CTL* logic. As with SDL logic, we define $P \equiv \neg O \neg$. Finally, we define the abbreviation $\Delta \equiv \neg\blacktriangle\neg$. We say that ϕ is a state formula iff N and U operators do not occur outside the scope of an O or A operator.

2.1 RoCTL-Structures

Definition 1. *A valuation g is a map from a set of states A to the power set of the variables; we represent the statement "the variable p is true at state w" with $p \in g(w)$.*

Definition 2. *A RoCTL-structure M is a 4-tuple*

$$(A, R^s, R^f, g),$$

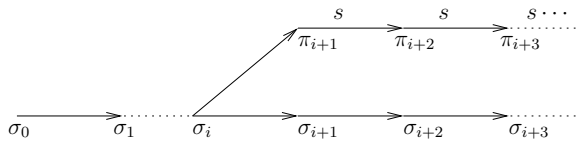
consisting of a set of states A, a serial (total) binary "success" relation R^s, a binary "failure" relation R^f and a valuation g on the set of states A. We define $\mathbb{M}$ as the class of such RoCTL-structures.

Definition 3. *We define R^{sf} to be an abbreviation of $\left(R^s \cup R^f\right)$. We call an ω-sequence $\sigma = \langle w_0, w_1, \ldots\rangle$ of states a fullpath iff for all non-negative integers i we have $w_i R^{sf} w_{i+1}$. For all i in $\mathbb{N}$ we define $\sigma_{\geq i}$ to be the fullpath $\langle w_i, w_{i+1}, \ldots\rangle$, we define σ_i to be w_i and we define $\sigma_{\leq i}$ to be the sequence $\langle w_0, w_1, \ldots, w_i\rangle$.*

Definition 4. *We say that a fullpath σ is failure-free iff for all $i \in \mathbb{N}$ we have $\sigma_i R^s \sigma_{i+1}$. We define $SF(w)$ to be the set of all fullpaths in M starting with w and $S(w)$ to be the set of all* failure-free *fullpaths in M starting with w.*

Definition 5. *For two fullpaths σ and π we say that π is an i-deviation from σ iff $\sigma_{\leq i} = \pi_{\leq i}$ and $\pi_{\geq i+1} \in S(\pi_{i+1})$. We say that π is a deviation from σ if there exists a non-negative integer i such that π is an i-deviation from σ. We define a function δ from a fullpath to a set of fullpaths such that where σ and π are fullpaths, π is a member of $\delta(\sigma)$ iff π is a deviation from σ.*

Below is an example of an i-deviation π from a fullpath σ. The arrows not labeled with s can be either R^s or R^f. The diagonal arrow represents the unexpected event, which can be a success or failure. After π diverges from σ, it avoids any failures that may have been on $\sigma_{>i}$. We require that a deviation not introduce any failures except for the deviating event itself, hence $\pi_{\geq i+1}$ is failure-free.



Note that after the deviation, all transitions must be success transitions while steps before the deviation may be either success or failure transitions. This follows from the intuition of Robustly representing "even if an additional failure occurs", as the failures that are prior to the deviation are already on the existing path. Additionally, allowing failures to occur before the deviation also allows n Robustly operators to be nested to represent the statement "even if n additional failures occur".

2.2 RoCTL* Semantics

We define truth of a RoCTL* formula ϕ on a fullpath $\sigma = \langle w_0, w_1, \ldots \rangle$ in RoCTL-structure M recursively as follows:

$$
\begin{aligned}
M, \sigma &\vDash N\phi \text{ iff } M, \sigma_{\geq 1} \vDash \phi \\
M, \sigma &\vDash \phi U \psi \text{ iff } \exists_{i \in \mathbb{N}} \text{ s.t. } M, \sigma_{\geq i} \vDash \psi \\
&\qquad \text{and } \forall_{j \in \mathbb{N}} j < i \implies M, \sigma_{\geq j} \vDash \psi \\
M, \sigma &\vDash A\phi \text{ iff } \forall_{\pi \in \mathcal{SF}(\sigma_0)} M, \pi \vDash \phi \\
M, \sigma &\vDash O\phi \text{ iff } \forall_{\pi \in S(\sigma_0)} M, \pi \vDash \phi \\
M, \sigma &\vDash \blacktriangle\phi \text{ iff } \forall_{\pi \in \delta(\sigma)} M, \pi \vDash \phi \text{ and } M, \sigma \vDash \phi\,.
\end{aligned}
$$

The definitions for $\top$, p, $\neg$ and $\wedge$ are as we would expect from classical logic. We say that a formula ϕ is valid in RoCTL* iff for all structures M in $\mathbb{M}$, for all fullpaths σ in M we have $M, \sigma \vDash \phi$. Where M is obvious from the context, we will just write $\sigma \vDash \phi$.

It is also easy to show that if R^f is empty, the O and $\blacktriangle$ operators are equivalent to the A operator. Restricting the class of structures such that $R^s \cap R^f = \emptyset$ and/or such that R^f is serial does not affect the validities of the logic.

2.3 RoBCTL*

The RoBCTL* logic is similar to the RoCTL* logic; however, path quantifiers quantify only over paths within a bundle B. We will define RoBCTL* below.

Definition 6. *A RoBCTL-structure M is a 5-tuple*

$$\left(A, R^s, R^f, g, B\right),$$

consisting of a set of states A, a serial (total) binary "success" relation R^s, a binary "failure" relation R^f a valuation g on the set of states A, and a prefix and fusion closed bundle B of paths such that for every $x \in A$ there is a path $\sigma \in B$ such that σ is failure-free.

RoBCTL* has the same syntax as RoCTL*. The semantics are also similar; however, RoBCTL* uses $\mathcal{SF}_B$, S_B, and δ_B in place of $\mathcal{SF}$, S, and δ where $\mathcal{SF}_B(\sigma_0) = \mathcal{SF}(\sigma_0) \cap B$, $S_B(\sigma_0) = S(\sigma_0) \cap B$, and $\delta_B(\sigma) = \delta(\sigma) \cap B$.

3. PROPERTIES OF ROCTL*

It is easy to show that the fragment of RoCTL* which consists only of the O operator acting on atomic propositions is the trivial logic Triv, which adds the axiom $p \leftrightarrow \Box p$ to K; see for example [9]. We can axiomatise this fragment by adding $p \rightarrow Op$, $p \rightarrow Pp$ to an axiomatisation for classical logic.

Here we will show that the fragment of RoCTL*, which has the O operator acting on LTL path formulas, is equivalent to KD45.

We define fragments of RoCTL* over path formulas as follows. We say that a formula ϕ is *opaque* iff if it is of the form $N\psi$, $\psi_1 U \psi_2$ or p. We say that a formula ϕ is valid in the fragment iff ϕ' is valid for all ϕ' that could result from opaque subformulas in ϕ being replaced with other formulas. Note that as atoms are opaque, a formula ϕ is valid in the fragment iff ϕ' is valid, where ϕ' is the formula that results where every opaque formula ψ in ϕ is replaced with an atom p_ψ.

We can define an axiomatisation KD45 for KD45 by adding the D, 4 and 5 axioms to K. We present these axioms[1], and the equivalent conditions of frames below:

Necessitation Rule: If A is a theorem of K, then so is $\Box A$.

The Axioms K, D, 4, 5 are as follows. We also here define the T axiom which is clearly not in KD45, and something that ought to be true and is believed to be true can still be false.

Name	Axiom	Condition on Frames
K	$\Box(\phi \rightarrow \psi) \rightarrow (\Box\phi \rightarrow \Box\psi)$	$\top$
D	$\Box\phi \rightarrow \Diamond\phi$	$\exists_u wRu$
4	$\Box\phi \rightarrow \Box\Box\phi$	$\forall_{w,v,u}(wRv \wedge vRu) \rightarrow wRu$
5	$\Diamond\phi \rightarrow \Box\Diamond\phi$	$\forall_{w,v,u}(wRv \wedge wRu) \rightarrow vRu$
T	$\Box\phi \rightarrow \phi$	$\forall_W wRw$

Definition 7. *The formal system* K *is the axiom K together with the Necessition Rule: If ϕ is a theorem of K, then so is $\Box\phi$. The formal system* O *is the formal system* K *with the addition of the D, 4 and 5 axioms.*

For a world w let wR be the set S of worlds that $v \in S$ iff wRv.

Lemma 8. *For all $v \in wR$, we have $vR = wR$.*

Proof. Say that there is a world u in vR that is not in wR. Then we see that $wRvRu$ but $\neg wRv$; however, R is Transitive (4). Say that there is a world u in wR that

[1]see `http://plato.stanford.edu/entries/logic-modal/`

is not in vR. Then we see that wRv and wRu but $\neg vRu$. Since R is Euclidean (5), we know by contradiction that $vR = wR$. $\square$

Say that the accessibility relation R is Serial, Transitive and Euclidean. Consider the set of worlds S reachable from some world w (which may not include w). Say that there exists a pair of worlds $u, v \in S$ such $\langle u, v\rangle \notin R$. As R is transitive, it is clear that wRu and wRv. As R is Euclidean, it must be the case that uRv. By contradiction no such u, v exist. Hence R forms the complete graph over S. As R is serial, S is non-empty.

We want to show that every validity of this fragment of RoCTL* is a validity of $\mathbb{KD}45$. RoCTL* is written in a different format to KD45. As it is O we are applying the KD45 axioms to, we will here consider O to be way of writing $\Box$, and we will treat opaque statements as atomic propositions in KD45.

Definition 9. *We will use $\vDash_{\Box}$ to represent $\vDash$ under the normal multimodal Kripke semantics where modal operators quantify over worlds. We use $\vDash_R$ to represent $\vDash$ under the semantics of RoCTL* where O, A and $\blacktriangle$ quantify over paths rather than worlds. For any formula ϕ let ϕ^N be ϕ with every atom p replaced with Np. For example $\psi = p \wedge q \iff \psi^N = Np \wedge Nq$.*

Lemma 10. *If ϕ is satisfiable in KD45, then ϕ^N is satisfiable in RoCTL*.*

This lemma is a special case of Lemma 26.

It is easy to show that axioms of KD45 are sound in this fragment (see e.g. Theorem 29), so the theorem below follows from the lemma above.

Theorem 11. *The axioms of KD45 provide a sound and complete axiomatization of the fragment of RoCTL* that has O acting over path formulas.*

4. AXIOMATIZATION FOR O AND A

It is easy to show that the A operator over path formulas has the same validities as S5, which we can axiomatise as KT5 (see e.g. [18] for an axiomatization of the operators of CTL*). As shown above O is KD45, also known as Deontic S5. The interactions between O and A are as follows:

1. $A\phi \rightarrow O\phi$
2. $O\phi \rightarrow AO\phi$

Definition 12. *Let the formal system OA be the combination of all the above axioms and the Necessitation rule.*

4.1 Completeness of OA

In this section we will prove the completeness of OA. We will first define a class of structures $\mathbb{OA}$. We show that the canonical model is in $\mathbb{OA}$, and hence that OA is complete with respect to $\mathbb{OA}$. Finally we show that every validity of $\mathbb{OA}$ is a validity of RoCTL*, which gives us completeness with respect to RoCTL*.

Below we state a number of well known properties of modal logics, proofs of these can be found in textbooks such as [9].

Definition 13. *A* canonical frame *is a structure $\mathcal{A} = (A, R)$ where the set of possible worlds A is the set of maximally consistent sets, and given any two sets of maximally consistent formulas s and t we define sRt to be true iff the condition $\Box\phi \in s \implies \phi \in t$ holds for all formulas ϕ [9]. The* canonical valuation *g is the valuation such that $g(s) = s \cap \mathcal{V}$. The 3-tuple (A, R, g) is called the* canonical model.

Definition 14. *A "maximally consistent set of formulas" is a set Ψ of formulas such that Ψ is consistent, but for any formula ϕ not in Ψ, adding ϕ to Ψ will result in an inconsistent set of formulas.*

Lemma 15. *(Basic Existence Result) For any formal system, and for every set of formulas Ψ consistent in S, there is a maximally S-consistent set of formulas s such that all formulas in Ψ are in s.*

Corollary 16. *Any formula ϕ is provable within a formal system S iff ϕ is a member of all maximally S-consistent formulas.*

Definition 17. *A formal system S is* canonical *if all valuations of the canonical frame are models of the formal system.*

Theorem 18. *Let S be a standard formal system with canonical valued structure $\mathcal{A}$ and valuation g. Say S is sound with respect to some class $\mathbb{M}$ that contains $\mathcal{A}$. Then for each formula ϕ, the conditions*
(ii) $(\mathcal{A}, g) \vDash \phi$,
(iii) $\vdash_{\mathsf{S}} \phi$,
(iv) $\mathbb{M} \vDash \phi$
are equivalent.

Definition 19. *We define $\mathbb{KD}45$ as the class of models that satisfy the conditions for D, 4 and 5, and likewise KD45 as the formal system that results when we add the axioms D, 4, and 5 to K. We define $\mathbb{S}5$ as being the models that satisfy the conditions of T and 5. We define S5 as the formal system that results when we add the axioms T and 5 to K.*

Definition 20. *We let the class $\mathbb{OA}$ of structures be the class of multidimensional structures where (A, R^O, R^A, g) is a member of $\mathbb{OA}$ iff $(A, R^O, g) \in \mathbb{KD}45$, $(A, R^A, g) \in \mathbb{S}5$, $uR^Ov \implies uR^Av$ and $(uR^Av \wedge vR^Ow) \implies uR^Ow$.*

LEMMA 21. The canonical model of OA is in $\mathbb{OA}$.

PROOF. Since O satisfies the axioms of KD45, it is clear that $\left(A, R^O, g\right) \in \mathbb{KD}45$. Likewise it is clear that $\left(A, R^A, g\right) \in \mathbb{S}5$.

From the axiom $A\phi \rightarrow O\phi$ it is clear that if $A\phi$ is in a maximally consistent set u, then $O\phi$ is also in u. Recall that in a canonical model uRv iff for every $\Box\phi \in u$ it is the case that $\phi \in v$. So if uR^Ov, then for all ϕ such that $O\phi$ in u, it is the case that $\phi \in v$. For all ϕ such that $A\phi \in u$, we see that $O\phi \in u$ and so $\phi \in v$. Thus $uR^Ov \implies uR^Av$ for all $u, v \in A$. $\square$

Say that $\left(uR^Av \wedge vR^Ow\right) \implies uR^Ow$ is false. Then uR^Ow is false, and so there exists a formula ϕ such that $O\phi \in u$ but $\phi \notin w$. As $O\phi \in u$ it is clear from the axiom $O\phi \rightarrow AO\phi$ that $AO\phi \in w$. Given that $\left(uR^Av \wedge vR^Ow\right)$ we see that $O\phi \in v$ and hence that $\phi \in w$. Since $\phi \notin w$ we can say by contradiction that $\left(uR^Av \wedge vR^Ow\right) \implies uR^Ow$ is true.

DEFINITION 22. *We say that a model is connected if there is a path from every world to every other world. For example, we say a model $\left(A, R^O, R^A, g\right)$ in $\mathbb{OA}$ is connected iff for every u, v in A there exists a sequence $w_0, w_1, \ldots, w_i$ such that $w_0 = u$, $w_i = v$ and for each $j \in [1, i]$ we have $\langle w_{j-1}, w_j\rangle \in \left(R^A \cup R^O\right)$.*

LEMMA 23. *Given a connected model $\left(A, R^O, R^A, g\right)$ in $\mathbb{OA}$, for every $u, v \in A$ we have $uR^O = vR^O$ and $uR^A = vR^A$ is the complete graph over A.*

PROOF. As, $uR^Ov \implies uR^Av$ we see that $\left(A, R^A, g\right)$ is also connected. As $\left(A, R^A, g\right) \in \mathbb{S}5$ and is connected, we know that R^A is the complete graph over A. Say that $uR^O \neq vR^O$. WLOG we can say that there exists a world w such that $w \notin uR^O$ but $w \in vR^O$; however it is clear that uR^Av as R^A is the complete graph and so this contradicts the property $\left(uR^Av \wedge vR^Ow\right) \implies uR^Ow$ of $\mathbb{OA}$. The result follows from contradiction. $\square$

LEMMA 24. OA is sound and complete in $\mathbb{OA}$.

PROOF. Obvious from the above lemmas. $\square$

DEFINITION 25. *We define a function T from models in $\mathbb{OA}$ to models of RoCTL* as follows. Where $M = \left(A, R^O, R^A, g\right)$ is a connected model in $\mathbb{OA}$, we let $T(M) = \left(A', R^s, R^f, g'\right)$ be the member of $\mathbb{M}$ that satisfies:*

- $A' = A \cup \{w_0, w_Z\}$, where w_0 and w_Z are arbitrary worlds not in A.
- R^s is the minimal relation satisfying $w_0R^s = uR^O$ and wRw for all $w \in A$. The relation $R^f = uR^A$. The world u can be any arbitrary world in A, see Lemma 23.
- g' is the function on A' such that $g'(w) = g(w)$ for $w \in A$ and $g'(w_0) = \emptyset$.

Additionally, Where $v \in A$, let σ^v be the path $\langle w_0, v, v, \ldots\rangle$.

LEMMA 26. *For any formula ϕ, model $M \in \mathbb{OA}$ and $a \in A$ such that $M, a \vDash^O \phi$ it is the case that $T(M), \sigma^a \vDash_R \phi^N$.*

PROOF. Say there exists a formula ϕ which is a validity of this fragment of RoCTL* but which is not a validity of $\mathbb{KD}45$. As substituting opaque formulas does not affect validity, we can assume WLOG that all opaque subformulas of ϕ are atomic propositions. For any ψ, let ψ^N be ψ with every occurrence of every atom p in ψ replaced with Np.

Since ϕ is not valid in OA, there exists a connected structure $M = \left(A, R^O, R^A, g\right)$ and $a \in A$ such that $M, a \vDash_\Box \neg\phi$. Let $M = T(M)$. Say that ψ is the smallest formula such that it is not the case that

$$\forall_{w \in A} M', \sigma^a \vDash_R \psi^N \iff M, a \vDash_\Box \psi \ .$$

We will now consider each possible form of ψ, and show that the above property must hold.

$\psi = p$: Then we see that as $p \in g'(w) \iff p \in g(w)$ and $\sigma_1^w = w$ it is the case that $M', \sigma^w \vDash_R \psi^N$ iff $M, w \vDash_\Box \psi$

$\psi = N\phi$ or $\psi = \phi_1 U \phi_2$: As ψ is opaque, it is equivalent to an atom p.

$\psi = \neg\phi$: We see that as $M', \sigma^w \vDash_R \phi^N$ iff $M, \sigma^w \vDash_\Box \phi^N$, it must also be the case that $M', \sigma^w \vDash_R \neg\phi^N \iff M, w \vDash_\Box \neg\phi$.

$\psi = \phi_1 \wedge \phi_2$: As above.

$\psi = O\phi$. ($\Longrightarrow$) Say that $M', \sigma^v \vDash_R \psi^N$. Then for all $w \in A'$ such that w_0R^sw it is the case that $M', \sigma^w \vDash_R \phi^N$. From the definition of R^s above and Lemma 23 we know that $aR^O = vR^O$ and w_0R^sw iff aR^Ow. Since ϕ is smaller than ψ we see that for all $w \in aR^O$ we have $M, \sigma^w \vDash_\Box \phi^N$. . Thus $M, w \vDash_\Box O\phi$.

($\Longleftarrow$) Say that $M, w \vDash_\Box \psi$. Then for all $v \in wR^O$ we have $M, v \vDash_\Box \phi$, and from Lemma 8 we know $wR^O = aR^O$. Thus for all $v \in aR^O$ it is the case that $M, v \vDash_\Box \phi$ and so $M', \sigma^w \vDash_R \phi^N$. We see that $w_0R^s = aR^O$, as so for all failure-free σ^v it is the case that $v \in aR^O$, and $\sigma^v \vDash_R \phi^N$. Hence $\sigma^w \vDash_R \psi$.

$\psi = A\phi$: Say that $M', \sigma^v \vDash_R \psi^N$. Then for all $w \in A'$ such that $w_0R^{sf}w$ it is the case that $M', \sigma^w \vDash_R \phi^N$. From the definition of R^{sf} above and Lemma 23 we know that $aR^A = vR^A$ and $w_0R^{sf}w$ iff aR^Aw. Since ϕ is smaller than ψ we see that for all $w \in aR^A$ we have $M, \sigma^w \vDash_\Box \phi^N$. . Thus $M, w \vDash_\Box A\phi$.

PROOF. ($\Longleftarrow$) Say that $M, w \vDash_\Box \psi$. Then for all $v \in wR^A$ we have $M, v \vDash_\Box \phi$, and from Lemma 8 we know $wR^A = aR^A$. Thus for all $v \in aR^A$ it is the case that $M, v \vDash_\Box \phi$

and so $M', \sigma^w \models_R \phi^N$. We see that $w_0 R^{sf} = aR^A$, as so for all σ^v it is the case that $v \in aR^A$, and $\sigma^v \models_R \phi^N$. Hence $\sigma^w \models_R \psi$. □

THEOREM 27. OA *is sound and complete in* $\mathbb{M}$*, where temporal formulae are treated as opaque.*

The soundness proof of this is essentially just a fragment of the proof of Theorem 29, and we have weak completeness from the lemma above.

5. AXIOMATISATION OF N, O, A AND ▲

In this section we will define NOA▲, which we will then prove is a sound and weakly complete axiomatisation of the fragment of RoCTL* without U, consisting only of N, O, A and ▲.

For the N operator we include two axioms N: $\neg N\phi \leftrightarrow N\neg\phi$ and Necessitation, i.e. $N(\phi \rightarrow \psi) \rightarrow (N\phi \rightarrow N\psi)$ [18, p1016].

Below we give the axioms for the A operator in BCTL*. The operators after the + indicate that the same holds for the operator after the plus, e.g. the $+O$ after C10 indicates that $O\phi \rightarrow OO\phi$ is validity of RoCTL*.

C9 $A(\phi \rightarrow \psi) \rightarrow (A\phi \rightarrow A\psi) + \blacktriangle O$

C10 $A\phi \rightarrow AA\phi + O$

C11 $A\phi \rightarrow \phi + \blacktriangle$

C12 $\phi \rightarrow AE\phi$

C13 $A\neg\phi \leftrightarrow \neg E\phi + \blacktriangle O$

C14 $p \rightarrow Ap + \blacktriangle O$

C15 $AN\phi \rightarrow NA\phi + \blacktriangle$

For all of the above axioms we will also note that it is obligatory that a similar axiom holds for the Obligatory operator. For example, even though $\phi \rightarrow OP\phi$ (from C12) is not a validity, $O(\phi \rightarrow OP\phi)$ is. We do not need all of these validities as axioms. For example $p \rightarrow Op$ is not required as $p \rightarrow Ap \rightarrow Op$.

As well as the axioms from CTL*, we add the following axioms from the axiomatisation of OA [17].

O1 $A\phi \rightarrow O\phi$

O2 $OA\phi \rightarrow A\phi$

O3 $O\phi \rightarrow AO\phi$

We will also add fourth O axiom to handle interactions with the N axiom.

O4 $O(NO\phi \rightarrow N\phi)$

We add the following new axioms:

R1 $(ANO\phi \wedge N\blacktriangle\phi) \leftrightarrow \blacktriangle N\phi$

R2 $A\phi \rightarrow A\blacktriangle\phi$

R3 $\blacktriangle\phi \rightarrow O\phi$

We will add the following axioms from K:

C9$^{\blacktriangle}$ $\blacktriangle(\phi \rightarrow \psi) \rightarrow (\blacktriangle\phi \rightarrow \blacktriangle\psi)$

C9^{O} $O(\phi \rightarrow \psi) \rightarrow (O\phi \rightarrow O\psi)$

As noted previously, we do not need all the CTL* like axioms. We will add just the following CTL*-like axiom:

C15^{O} $O(ON\phi \rightarrow NO\phi)$

We will not add C15$^{\blacktriangle}$, as it is obvious that $\blacktriangle N\phi \rightarrow N\blacktriangle\phi$ can be derived from R1.

We have now defined the formal system OA. We will note that not all of the axioms of the A operator are validities of the other operator. If $O\phi \rightarrow \phi$ or $\phi \rightarrow OP\phi$ were valid, this would imply that everything which was true was also permissible. It is important that $ON\phi \rightarrow NO\phi$ is not valid in RoCTL*, as this allows us to use RoCTL* to specify Contrary-to-Duty obligations.

The formula $\phi \rightarrow \blacktriangle\Delta\phi$ is not valid in RoCTL*. The reason for this is that a deviation can prevent any number of failures occurring in the future, but the second deviation can cause only a single new failure. The ▲ operator is not transitive, so $\blacktriangle\phi \rightarrow \blacktriangle\blacktriangle\phi$ is not valid in RoCTL*.

5.1 Soundness of NOA▲

LEMMA 28. *If* ϕ *is valid in* $\mathbb{M}$ *then* $\Box\phi$ *is also valid in* $\mathbb{M}$*, for* $\Box \in N, O, A, R$.

PROOF. We see that if $\sigma \not\models \Box\phi$ then there is a path π such that $\pi \not\models \phi$. Hence it is not possible for ϕ to be valid if $\Box\phi$ is not. □

THEOREM 29. *The formal system* NOA▲ *is sound with respect to the semantics of RoCTL* and RoBCTL* models.*

Below we prove that NOA▲ is sound with respect to the semantics of RoCTL*. It is easy to adapt this proof to RoBCTL* models. From the semantics of RoCTL*, it is clear that if ϕ is a validity, so is $A\phi$, $\blacktriangle\phi$ and $O\phi$. Below we show that the axiom classes are all validities of RoCTL*.

O1 $A\phi \rightarrow O\phi$: Say that $\sigma \not\models A\phi \rightarrow O\phi$ for some formulas ϕ. Then $\sigma \models A\phi$, and so for every path $\pi \in SF(\sigma_0)$ it is the case that $\pi \models \phi$. Since $S(\sigma_0) \subseteq SF(\sigma_0)$, we see $\sigma \models O\phi$ and so $\sigma \models A\phi \rightarrow O\phi$. By contradiction we see that $A\phi \rightarrow O\phi$ is valid.

O2 $OA\phi \rightarrow A\phi$: Say that $\sigma \not\models OA\phi \rightarrow A\phi$. Then $\sigma \models OA\phi$, thus for every $\pi \in \mathcal{SF}(\sigma_0)$ it is the case that $\pi \models A\phi$. As R^s is serial, we know that such π exists. Thus for all $\theta \in S(\pi_0)$ it is the case that $\theta \models \phi$. Since $\pi_0 = \sigma_0$ we see that $\sigma \models A\phi$ and so $\sigma \models OA\phi \rightarrow A\phi$.

O3 $O\phi \rightarrow AO\phi$: As above we see that $S(\sigma_0) = S(\pi_0)$ for every π with $\pi_0 = \sigma_0$ and so this axiom as likewise valid.

O4 $O(NO\phi \rightarrow N\phi)$: Say that $\sigma \not\models O(NO\phi \rightarrow N\phi)$. Then there exists $\pi \in S(\sigma_0)$ such that $\pi \not\models NO\phi \rightarrow N\phi$. Thus $\pi \models NO\phi$ and so for all τ such that $\tau_0 \in S(\pi_1)$ we have $\tau \models \phi$. We see that as π and $\pi_{\geq 1}$ are failure-free, $\pi_{\geq 1} \in S(\pi_1)$ and so $\pi_{\geq 1} \models \phi$. Hence $\pi \models N\phi$ and $\pi \models O(NO\phi \rightarrow N\phi)$.

R1 $(ANO\phi \wedge N\blacktriangle\phi) \leftrightarrow \blacktriangle N\phi$: ($\rightarrow$) Say that

$$\sigma \not\models (ANO\phi \wedge N\blacktriangle\phi) \rightarrow \blacktriangle N\phi,$$

then $\sigma \models ANO\phi \wedge N\blacktriangle\phi$ and $\sigma \not\models \blacktriangle N\phi$, so there exists a deviation π from σ such that $\pi \not\models N\phi$. If π is a 0-deviation, then $\pi_{\geq 1}$ is failure-free, and since $\sigma \models ANO\phi$ it is easy to prove that $\pi \models \phi$. However as $\pi \not\models N\phi$ we can see that $\pi_{\geq 1} \not\models \phi$.

Hence, by contradiction π is not a 0-deviation from σ, and so π is an i-deviation from σ for some $i \geq 1$. We see that $\pi_1 = \sigma_1$ and that $\pi_{\geq 1}$ is an $(i-1)$-deviation from $\sigma_{\geq 1}$. As $\pi_{\geq 1} \not\models \phi$ it follows that$\sigma_{\geq 1} \not\models \blacktriangle\phi$ and that $\sigma \not\models N\blacktriangle\phi$. By contradiction $\sigma \models (ANO\phi \wedge N\blacktriangle\phi) \rightarrow \blacktriangle N\phi$ is valid.

($\leftarrow$) Say that $\sigma \not\models \blacktriangle N\phi \rightarrow (ANO\phi \wedge N\blacktriangle\phi)$, then $\sigma \models \blacktriangle N\phi$ and $\sigma \not\models (ANO\phi \wedge N\blacktriangle\phi)$. Say that $\sigma \not\models ANO\phi$, then there exists a path π with $\pi_0 = \sigma_0$ such that $\pi \not\models NO\phi$ and thus $\pi_{\geq 1} \not\models O\phi$. We see that there exists a path τ such that $\tau_0 \in S(\pi_1)$ and $\tau_0 \not\models \phi$. Thus $\sigma_0 \cdot \tau \not\models N\phi$, and we see that $\sigma_0 \cdot \tau$ is a 0-deviation from σ, so $\sigma \not\models \blacktriangle N\phi$. However, by assumption $\sigma \models \blacktriangle N\phi$

1. Say that $\sigma \not\models ANO\phi$, then there exists a path π with $\pi_0 = \sigma_0$ such that $\pi \not\models NO\phi$ and thus $\pi_{\geq 1} \not\models O\phi$. We see that there exists a path τ such that $\tau_0 \in S(\pi_1)$ and $\tau_0 \not\models \phi$. Thus $\sigma_0 \cdot \tau \not\models N\phi$, and we see that $\sigma_0 \cdot \tau$ is a 0-deviation from σ, so $\sigma \not\models \blacktriangle N\phi$. However, by assumption $\sigma \models \blacktriangle N\phi$
2. Say that $\sigma \not\models N\blacktriangle\phi$. Then we see that there is a deviation π from $\sigma_{\geq 1}$ such that $\pi \not\models \phi$, and that $\sigma_0 \cdot \pi$ is a deviation from σ. Since $\sigma_0 \cdot \pi \not\models N\phi$, it follows that $\sigma \not\models \blacktriangle N\phi$.

By contradiction, it follows that $(ANO\phi \wedge N\blacktriangle\phi) \leftrightarrow \blacktriangle N\phi$ is valid.

R2 $A\phi \rightarrow A\blacktriangle\phi$: Say that $\sigma \not\models A\phi \rightarrow A\blacktriangle\phi$. Then $\sigma \models A\phi$ and $\sigma \not\models A\blacktriangle\phi$. We see that for all $\pi \in \mathcal{SF}(\sigma)$, we have $\pi \models \phi$. For all deviations τ from π, we see that $\tau_0 = \pi_0 = \sigma_0$, so $\tau \in \mathcal{SF}(\phi)$ and so $\tau \models \phi$. Thus $\pi \models \blacktriangle\phi$ and $\sigma \models A\blacktriangle\phi$.

R3 $\blacktriangle\phi \rightarrow O\phi$: It is easy to show that $\delta(\sigma) \subseteq S(\sigma_0)$, and so if $\sigma \models \blacktriangle\phi$ then $\sigma \models O\phi$.

We will add the following axioms from K:

C9$^{\blacktriangle}$ $\blacktriangle(\phi \rightarrow \psi) \rightarrow (\blacktriangle\phi \rightarrow \blacktriangle\psi)$: If

$$\sigma \not\models \blacktriangle(\phi \rightarrow \psi) \quad \rightarrow \quad (\blacktriangle\phi \rightarrow \blacktriangle\psi)$$

then $\sigma \models \blacktriangle(\phi \rightarrow \psi) \wedge \blacktriangle\phi$. We see that for all paths $\pi \in \delta(\sigma)$, $\pi \models \phi$ and $\pi \models (\phi \rightarrow \psi)$. Hence $\pi \models \psi$. Thus $\sigma \models \blacktriangle\psi$.

C9^{O} $O(\phi \rightarrow \psi) \rightarrow (O\phi \rightarrow O\psi)$: As above, but replace $\delta(\sigma)$ with $S(\sigma_0)$

C15^{O} $O(ON\phi \rightarrow NO\phi)$: If $\sigma \not\models O(ON\phi \rightarrow NO\phi)$, we see that there exists $\pi \in S(\sigma)$ such that $\pi \not\models ON\phi \rightarrow NO\phi$. Since $\pi \not\models NO\phi$, there exists $\tau \in S(\pi_1)$ such that $\tau \not\models \phi$. Since $\pi_0 \cdot \tau$ is failure free $\pi_0 \cdot \tau \in S(\pi)$. As $\pi_0 \cdot \tau \not\models N\phi$, it follows that $\phi \not\models ON\phi$. By contradiction, C15^{O} must be valid.

6. REFERENCES

[1] Huib Aldewereld, Davide Grossi, Javier Vazquez-Salceda, and Frank Dignum. Designing normative behaviour by the use of landmarks. In *Agents, Norms and Institutions for Regulated Multiagent Systems*, Utrecht, The Netherlands, Jul 2005.

[2] A. Arnold, A. Vincent, and I. Walukiewicz. Games for synthesis of controllers with partial observation. *Theor. Comput. Sci.*, 303(1):7–34, 2003.

[3] Nuel Belnap. Backwards and forwards in the modal logic of agency. *Philosophy and Phenomenological Research*, 51(4):777–807, Dec 1991.

[4] Jan Broersen, Frank Dignum, Virginia Dignum, and John-Jules Ch. Meyer. *Designing a Deontic Logic of Deadlines*, volume 3065/2004 of *LNCS*, pages 43–56. Springer, 2004. doi: 10.1007/b98159.

[5] M. de Weerdt, A. Bos, H. Tonino, and C. Witteveen. A resource logic for multi-agent plan merging. *Annals of Mathematics and Artificial Intelligence*, 37(1-2):93–130, January 2003.

[6] James William Forrester. Gentle murder, or the adverbial samaritan. *The Journal of Philosophy*, 81(4):193–7, April 1984.

[7] Tim French, John C. M^{c}Cabe-Dansted, and Mark Reynolds. *A Temporal Logic of Robustness*, volume 4720 of *Lecture Notes in Computer Science*, pages 193–205. 2007. `http://dx.doi.org/10.1007/978-3-540-74621-8_13`.

[8] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[9] G. Hughes and M. Cresswell. *An Introduction to Modal Logic*. Methuen, London, 1968.

[10] Thierry Jéron, Hervé Marchand, Sophie Pinchinat, and Marie-Odile Cordier. Supervision patterns in discrete event systems diagnosis. In *8th International Workshop on Discrete Event Systems*, pages 262–268, July 2006.

[11] W. Long, Y. Sato, and M. Horigome. Quantification of sequential failure logic for fault tree analysis. *Reliability Engineering and System Safety*, 67:269–274, 2000.

[12] John C. McCabe-Dansted. A tableau for RoBCTL*. In Steffen Hölldobler, Carsten Lutz, and Heinrich Wansing, editors, *JELIA*, volume 5293 of *Lecture Notes in Computer Science*, pages 298–310. Springer, 2008.

[13] L. Thorne McCarty. Defeasible deontic reasoning. *Fundam. Inform.*, 21(1/2):125–148, 1994.

[14] John M^{c}Cabe-Dansted, Sophie Pinchinat, Tim French, and Mark Reynolds. On the expressivity of RoCTL*. Technical report, 2009. `http://www.csse.uwa.edu.au/~john/papers/RoCTLstar_Express_tech.pdf`, expanded version of paper to be presented at TIME09.

[15] John C. M^{c}Cabe-Dansted. A tableau for RoBCTL*. Technical report, UWA, 2008. `http://www.csse.uwa.edu.au/~john/papers/Da08Tableau.pdf`.

[16] John C. M^{c}Cabe-Dansted, Tim French, and Mark Reynolds. A temporal logic of robustness, RoCTL*. Technical report, UWA, 2007. `http://www.csse.uwa.edu.au/~john/papers/RoCTL07.pdf`.

[17] John C. M^{c}Cabe-Dansted, Tim French, and Mark Reynolds. Axioms for fragments of roctl* [expanded tech report]. Technical report, UWA, 2009. `http://www.csse.uwa.edu.au/~john/papers/Axioms_tech.pdf`.

[18] Mark Reynolds. An axiomatization of full computation tree logic. *The Journal of Symbolic Logic*, 66(3):1011–1057, September 2001.

[19] Agerri Rodrigo and Alonso Eduardo. Normative pragmatics for agent communication languages. In *Perspectives in Conceptual Modelling. (LNCS)*, volume 3770, pages 172–181. Springer, 2005.

[20] Leendert W. N. van der Torre and Yaohua Tan. The temporal analysis of Chisholm's paradox. In Ted Senator and Bruce Buchanan, editors, *Proceedings of the Fourteenth National Conference on Artificial Intelligence and the Ninth Innovative Applications of Artificial Intelligence Conference*, pages 650–655. AAAI Press, 1998.

Real-time Visual SLAM Using Python & OpenCV

Robert Reid
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: rob@csse.uwa.edu.au

ABSTRACT

In this paper we present our work developing a real-time visual SLAM implementation. To the best of our knowledge it is the first implementation to be described that provides Matlab-like rapid prototyping within a real-time multi-threaded environment. We have used the Python programming language to integrate several open source software libraries, including OpenCV, and believe the framework achieves an ideal balance between real-time performance and rapid prototypability.

Our extended Kalman filter (EKF) implementation combines key aspects from current visual SLAM research, including inverse depth parameterisation (IDP) for features. We describe in detail our EKF formulation and derivation of a measurement model specific to OpenCV.

Visual SLAM has received considerable attention in recent years from both the robotic and computer vision communities. We suggest that visual SLAM techniques will have many uses in the future at the core of a wide range of visual mapping and perception applications. We note, however, that there are currently very few software implementations available and suggest that more widely accessible implementations could serve as a basis for ongoing research and benefit the community greatly. Our contribution in this paper aims to fill a small part of this gap.

Keywords

Real-time Visual Simultaneous Localisation and Mapping, Inverse Depth Parametrisation, Python, OpenCV

1. INTRODUCTION

Simultaneous localisation and mapping (SLAM) is the name given to the problem of estimating the ego motion of a sensor platform while it moves through and creates a map of an unknown environment. The problem is often posed in the field of mobile robotics, where a robot must perform SLAM using its sensor data to navigate and interact in the real world. SLAM implementations have to contend with noisy and limited sensor data while operating within computational restraints. The presence of noise and such limitations have encouraged the development of a variety of Bayesian probabilistic approaches such as the Extended Kalman filters (EKF), Particle Filters and Sparse Information Filters. The mathematical basis of the SLAM problem is now well understood [15].

Early research typically focused on active sensors, such as time-of-flight laser scanners providing both range and bearing information. These were often supplemented with odometry information, such as wheel encoders, and limited to 2D planar environments, making the problem hard but tractable. A more difficult subset of the SLAM problem arises if both range and odometry information are not available. This "bearing-only" SLAM problem is more complex since a robot now has to *move* to triangulate the location of features as it adds them to its map, however it doesn't know how far it has moved or the direction.

More recently, solutions to this bearing-only SLAM problem, combined with cameras and computer vision techniques, have resulted in a new area of "visual SLAM" research pioneered by Davison et al. with their MonoSLAM work [13, 10, 12]. Using images from a calibrated camera, they identify salient visual features which they extract and follow over subsequent camera images. These "feature correspondences" are used as bearing-only measurements in an EKF SLAM system. It has been shown that such measurements taken by a camera moving with 6DOF can be used to create sparse 3D feature maps while accurately localising the camera. Since Davison's original work, there have been several significant advancements in the area [14, 23, 8, 28] making implementations both more robust and scalable. A review of visual SLAM techniques is given in [3].

Other visual mapping techniques, such as structure from motion (SFM) can accurately reconstruct camera ego motion and produce rich 3D maps, however they are historically performed off-line and are not suitable for interactive or real-time systems like visual SLAM. These differences have become blurred more recently with sequential SFM [4] and hybrid approaches such as Klein's parallel tracking and mapping [17], although they typically still suffer from problems such as long-term drift.

In visual SLAM features can serve as long-term landmarks and provided their correspondences are correctly identified the probabilistic SLAM map will continuously adapt to maintain its own integrity. This provides for important capabilities such as loop-closure, even in the presence of systematic errors such as those in the EKF linearisation. The

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

importance of identifying correct feature correspondences is stressed here, as EKF SLAM maps will rapidly degenerate if a incorrect matches are made. Recent joint compatibility branch and bound (JCBB) research by Clemente et al. [9] has been shown how to detect and reject incorrect feature matches.

A variety of rich visual feature representations, eg. SIFT descriptors have been shown to improve feature matching, however with prohibitive computational requirements. Recent works have shown multi-resolution descriptors [2], and affine-distorted patches [18, 14] used for feature matching in real-time.

Davison's original visual SLAM implementation used a delayed feature initialisation technique. Each feature was observed until enough parallax allowed its location to be estimated with a Euclidean XYZ point and its location uncertainty with a 3×3 covariance matrix. Recently, inverse depth parametrisation (IDP) techniques have been developed [19, 7, 8, 25] that allow each feature to be immediately initialised into the map. IDP features describe a roughly elliptical location uncertainty that potentially extend to "infinity". They are ideal for mapping features both close and at infinity and contribute to camera localisation as soon as they are initialised.

Online or real-time implementations (arguably the only useful kind) have a upper limit to the number of map features that can be maintained. An EKF visual SLAM implementation with N mapped features executes in $O\left(N^2\right)$ time. These scalability issues have been addressed with various sub-mapping techniques [23, 9].

There have been a few visual SLAM implementations released to the community. The C++ source to Davison's original work [10] was released as SceneLib 1.0 [11]. Unreleased updates to it have been described in a recent publication [14] and we suspect several other publications have been based on it also. Civera et al. presented a Matlab EKF implementation at the SLAM Summer School in 2006 [5]. This basic off-line implementation of IDP is a good learning tool. The OpenSLAM project [20] includes sources to a variety of non-visual SLAM implementations. Sola has very recently released a Matlab EKF SLAM toolbox [26] with plans to include visual SLAM algorithms. We note one commercial robotics platform that includes a visual SLAM implementation, although it relies on odometry [16].

1.1 Research Opportunities

Most futurists predicts that computer technology and robotics will play an increasing role in our everyday lives. For these systems to become more pervasive we will need to provide them with a form of spatial perception. In our opinion a type of probabilistic mapping framework using visual SLAM techniques will be at the core of any future research in visual mapping and perception.

Current research in visual SLAM techniques include mobile robotics, augmented reality, wearable computing, medical imaging in vivo and other applications that could benefit from small and cheap localisation and mapping.

While the theoretical basis for probabilistic SLAM is well understood, it is our belief that significant work still remains on several aspects of the visual SLAM problem. Further, we believe that entry into the visual SLAM research area is currently hampered by a lack of widely available software implementations.

We suggest the community needs one or more visual SLAM implementations produced using an open source software (OSS) approach. Ideally these implementations would be modular and rapidly prototypable, while still capable of operating in real-time. An ideal implementation might allow a robotics engineer to simply download a visual SLAM package into a robot's operating system and integrate it with minimal programming.

It is evident from various publications that off-line Matlab implementations are often preferred, later being optimised into compiled C/C++ code that is capable of executing in real-time. We question the efficiency of this approach, and ask whether it may be possible to combine rapid Matlab-like prototyping into a system while maintaining real-time performance. Ideally such a combination would provide a base for future research in visual SLAM and robotic mapping and perception.

1.2 Our Contribution

In this paper we describe the initial stages of our research including an implementation of a real-time visual SLAM algorithm using the Python language [22] and the OpenCV library [21, 1]. It reproduces parts of Davison's original work implementing an EKF-based visual SLAM algorithm, while using the newer IDP feature representation. To the best of our knowledge this it the first visual SLAM implementation providing Matlab-like rapid prototyping within a real-time multi-threaded environment.

To create this environment we have used Python to "glue" a number of free OSS libraries together. The resulting software has a modern graphical user interface (GUI) including real-time displays, plots and an interactive Python console. Most of the heavy computation is performed within optimised external libraries, allowing Python's Global Interpreter Lock (GIL) to be released and the system to execute multiple threads simultaneously. Further details of our software implementation are presented in section 3.

Our implementation can successfully map a small indoor area while running in real-time at 30Hz. It is currently limited to about 20 IDP features while executing on a quad-core 2.6Ghz Intel processor. We discuss options that could greatly improve performance in section 5 and note that these are presently being implemented. The hand-held camera used in our research was a Unibrain Firewire camera [29] providing 640×480 resolution colour images through an 81° field of view (FOV) wide-angle lens. We present some initial results from our work in section 4.

2. VSLAM EKF IMPLEMENTATION

Our visual SLAM EKF implementation is based on Davison et al's MonoSLAM work in [13, 10, 12, 14] while incorporating Montiel and Civera's IDP feature representation from [19, 7, 8]. We have derived the EKF measurement model

using Zhang's intrinsic camera parameterisation [32] as it is implemented in OpenCV [1].

To describe our implementation we have adopted Davison et al's mathematical notation where possible. The publications cited above omit parts of the EKF formulation, we have included them here for completeness. For EKF related notation and full derivations of the EKF readers are directed to Welch and Bishop's tutorial [30].

We begin by describing our implementation in steady-state operation below, while section 2.5 explains how the system is initialised. Section 2.6 and 2.7 describe new features initialisation and map management.

2.1 State Definition

Our EKF implementation models a camera moving with 6DOF at constant linear and angular velocity. The full state vector $\hat{x}$ is:

$$\hat{x} = \begin{bmatrix} \hat{x}_v & \hat{y}_1 & \hat{y}_2 & \dots & \hat{y}_{n+m} \end{bmatrix}^T \tag{1}$$

Where $\hat{x}_v$ represents the camera model and $\hat{y}_1, \hat{y}_2 \dots \hat{y}_{n+m}$ are the static 3D features within the current map. We describe each of these below. The EKF maintains an error covariance for the state vector in P.

The camera model $\hat{x}_v$ has 4 components:

$$\hat{x}_v = \begin{bmatrix} r^W & q^{WC} & v^W & \omega^C \end{bmatrix}^T \tag{2}$$

Where r^W is the camera's 3D position in the global coordinate system W, q^{WC} is a quaternion representing the camera's orientation, v^W is the camera's linear velocity and ω^C the angular rotation around each axis. The camera model has $(3+4+3+3) = 13$ elements.

The state vector has n highly localised map features $\hat{y}_1, \hat{y}_2 \dots \hat{y}_n$ parametrised as Euclidean XYZ points:

$$\hat{y}_i = \begin{bmatrix} X_i & Y_i & Z_i \end{bmatrix}^T \tag{3}$$

There are m features $\hat{y}_{n+1}, \hat{y}_{n+2} \dots \hat{y}_{n+m}$ defined using IDP parametrisation. These are either newly initialised features or ones that have been observed over minimal parallax:

$$\hat{y}_i = \begin{bmatrix} x_i & y_i & z_i & \theta_i & \phi_i & \rho_i \end{bmatrix}^T \tag{4}$$

In 4 and referring to figure 1 the first three parameters x_i, y_i, z_i are the estimate for the camera's optical centre $\left(r^W\right)$ from where a feature is first observed, while θ_i and ϕ_i represent the azimuth and elevation of a 3D ray originating from $(x_i, y_i, z_i)^T$ and extending through the observed feature (with respect to the global coordinate system). The feature is expected to lie somewhere along this ray with

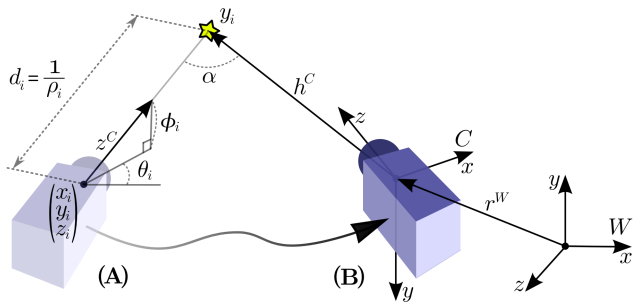


Figure 1: An IDP feature $\hat{y}_i$ is initialised by the camera at (A) and observed while the camera moves to a new location (B), r^W in world coordinate system W. The camera repeatedly observes the feature as ray h^C over a parallax angle α. The feature's inverse depth estimate ρ_i converges to represent a point on the ray z^C.

$\rho_i = \frac{1}{d_i}$ being the current *inverse* depth estimate. Refer to [8] for a more detailed description of the IDP representation.

The EKF SLAM implementation maintains Gaussian probability density distributions for the entire state. While it is operating, each element of the state vector $\hat{x}$ maintains its current expected value (mean). The uncertainty or variance in each of these values is captured in the diagonal of the covariance matrix P (square symmetric, positive definite). The off-diagonal components of P represent the covariances between the camera and each of the mapped features. These are often visualised as "springs" of varying stiffnesses linking the camera and each of the features in the map [15].

2.2 Motion Model & Prediction

Each EKF iteration begins with a prediction step, where a motion model is applied to the previous state $\hat{x}_{k-1}$ to estimate the current state $\hat{x}_k^-$. The motion model describes how the camera is expected to move over time. It is a function of the previous state, control inputs and process noise $f\left(\hat{x}_{k-1}, u_{k-1}, w_{k-1}\right)$, respectively. In our implementation the camera is modelled as though it were floating in "outer space". Any changes to the camera's otherwise constant linear and angular velocities, are modelled as finite impulses with a zero-mean Gaussian distribution (process noise w_{k-1}). We have no control inputs to the EKF ($u_{k-1} = 0$). The EKF update equation for the time interval Δt becomes:

$$\hat{x}_k^- = f\left(\hat{x}_{k-1}, 0, 0\right) = \begin{bmatrix} r^W + v^W \Delta t \\ q^{WC} \otimes q\left(\omega^C \Delta t\right) \\ v^W \\ \omega^C \\ \hat{y}_1 \\ \vdots \end{bmatrix} \tag{5}$$

Here $q\left(\omega^C \Delta t\right)$ is the rotation $\omega^C \Delta t$ converted to a quaternion, and $\otimes$ represents quaternion multiplication. 5 reflects the fact that only the position and orientation of the camera

are predicted to change in the state. Further, given the assumption that the environment is static, each of the mapped features are simply copied to $\hat{x}_k^-$.

The process noise w_k gives the model uncoupled linear and angular velocity impulses $a\Delta t$ and $\alpha\Delta t$ respectively. In our experiments we found $a = 4ms^{-2}$ and $\alpha = 4rads^{-2}$ to be appropriate. The process noise is incorporated in the 6×6 diagonal covariance matrix Q_k:

$$Q_k = diag\left(\left[\begin{array}{cccc} (a\Delta t)^2 & \cdots & (\alpha\Delta t)^2 & \cdots \end{array}\right]\right) \tag{6}$$

In the EKF prediction step, the error covariance P_k is also projected such that Q_k increases uncertainties in the system:

$$P_k^- = \left[A_k P_{k-1} A_k^T\right] + \left[W_k Q_{k-1} W_k^T\right] \tag{7}$$

In 7 the matrices A_k and W_k are the Jacobians:

$$A_k = A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}},\ W_k = W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}} \tag{8}$$

2.3 Measurement Model

Following each EKF prediction step and before the update step we aim to make sufficient observations or "measurements" of the system to evolve the state $\hat{x}_k^-$ and ideally further constrain the error covariance estimate P_k^-. In the visual SLAM case we attempt to measure the pixel location $z_i = (u, v)^T$ for each of the features that might be found somewhere in the current camera image. The measurement model $h\left(\hat{x}_k^-, \hat{v}_k\right)$ (defined below) gives a prediction for the pixel location $h_i = (u_d, v_d)^T$ of each of these features. The difference between the predicted pixel location $h_i = (u_d, v_d)^T$ and the measured location $z_i = (u, v)^T$ are used in the update step in section 2.4.

At this point our implementation deviates from Davison et al's work to incorporate the OpenCV libraries [21]. The camera's principal axis has been rotated by 180° (see figure 1). More significantly we have derived the measurement model for both XYZ and IDP features using Zhang's intrinsic camera lens parameterisation [32]. After experimenting with our wide-FOV camera lens, and observing negligible and noisy tangential distortion values for p_1 and p_2, we chose to remove them from the measurement model. We chose to incorporate a third radial distortion parameter ($k_3 r^6$) since it was observed to decreased calibration re-projection errors. Similarly through repeated calibrations we identified that the Unibrain camera's CCD has negligible skew $\gamma \simeq 0$ and a pixel aspect ratio $\frac{\alpha}{\beta} \simeq 1$. The resulting radial-only lens distortion equations are given below.

2.3.1 Measurement Prediction

The measurement model $h_i\left(\hat{x}_k, v_k\right)$ is used in the EKF to predict where feature i will be found in the current image. In our derivation we separate the prediction $h_i\left(\hat{x}_k^-, 0\right)$ into three nested functions:

$$h_i\left(\hat{x}_k^-, 0\right) = f_d\left(f_p\left(f_C\left(\hat{x}_k^-\right)\right)\right) \tag{9}$$

The innermost function $f_C\left(\hat{x}_k^-\right)$ transforms the feature's current position estimate into a camera-centric coordinates h^C. For features parametrised in XYZ:

$$h^C = f_C\left(\hat{x}_k^-\right) = R^{CW}\left(y_i^W - r^{WC}\right) \tag{10}$$

Where R^{CW} is the inverse of the quaternion rotation q^{WC} in matrix form. For points parametrised in IDP: $h^C = f_C\left(\hat{x}_k^-\right)$

$$= R^{CW}\left[\left(\rho_i \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} - r^{WC}\right) + \begin{bmatrix} cos\phi_i sin\theta_i \\ sin\phi_i \\ cos\phi_i cos\theta_i \end{bmatrix}\right] \tag{11}$$

In 9 the second nested function $f_p\left(h^C\right)$ performs the pinhole projection of a feature's location into normalised image coordinates h_p:

$$h_p = \begin{bmatrix} x \\ y \end{bmatrix} = f_p\left(h^C\right) = \frac{1}{h_z^C}\begin{bmatrix} h_x^C \\ h_y^C \end{bmatrix} \tag{12}$$

Finally the outer function $f_d\left(h_p\right)$ applies the radial-only OpenCV lens distortion model to produce the distorted camera image pixel coordinates: $h_i = (u_d, v_d)^T$

$$= f_d\left(h_p\right) = \begin{bmatrix} \alpha x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) + u_0 \\ \beta y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) + v_0 \end{bmatrix} \tag{13}$$

Where the normalised image radial distance $r = \sqrt{x^2 + y^2}$ and the camera calibration parameters are α, β, u_0, v_0, k_1, k_2, and k_3.

For each of the $n+m$ features in the map a prediction using 9 is made. However it can often be aborted before 12 if the feature is behind the camera ($h_z^C < 0$) or otherwise before 13 if it is predicted to be out of the scene. This also prevents unnecessary Jacobian calculations.

Given the q features predicted to exist in the current camera image, the EKF update step requires the Jacobian H_i for each feature. H_i is the partial derivative of the predicted pixel coordinates (u_d, v_d) with respect to the predicted state vector $\hat{x}_k^-$. Given the nested functions comprising $h_i\left(\hat{x}_k\right)$ from 9, H_i can be calculated analytically using the chain-rule:

$$H_i = \frac{\partial h_i}{\partial \hat{x}_k^-} = \begin{bmatrix} \partial u_d/\partial \hat{x}_k^- \\ \partial v_d/\partial \hat{x}_k^- \end{bmatrix} = \frac{\delta h_i}{\delta h_p} \cdot \frac{\delta h_p}{\delta h^C} \cdot \frac{\delta h^C}{\partial \hat{x}_k^-} \tag{14}$$

The state vector $\hat{x}_k^-$ has $(13+3n+6m) = p$ elements. Hence each of the features' Jacobians H_i are of dimension $2 \times p$. For each of the q features with pixel predictions, the Jacobians H_i are stacked vertically to form H_k^-. The minus superscript is used here to differentiate these as part of the EKF prediction. An innovation covariance S_k^- can then be calculated:

$$S_k^- = H_k^- \hat{P}_k^- \left(H_k^-\right)^T + R_k^- \tag{15}$$

To minimise computation in large EKF maps, S_i can be calculated separately for each of the q predicted features [14]. The measurement noise in our implementation has been estimated as Gaussian with $\sigma_{pixel} = 1.0$ pixel across the entire camera image. The measurement noise covariance R_k^- is thus diagonal and contains $2q \times 2q$ elements, $R_k^- = diag\begin{pmatrix} \sigma_{pixel}^2 & \sigma_{pixel}^2 & \cdots \end{pmatrix}$.

2.3.2 Feature Measurement

Similar to Davison et al's work, each visual feature is defined at the middle of the centre pixel in a 13×13 grey-scale image patch. In our tests these image patches were successful in matching features over a wide viewing angle, while proving reasonably tolerant against false matches. Feature measurements are performed using an normalised cross correlation (NCC) search. To minimise computation and false matches the innovation covariance S_k is used to limit the search region for each feature.

For each of the q predicted features the 2×2 innovation covariance S_i is extracted from the diagonal of S_k, where the sub-matrix has the same two row/column indices as the feature in H_k. The search area is restricted to 3σ, or the 99.7% confidence interval, such that the horizontal search range is $\pm 3\sqrt{S_i(0,0)}$ pixels centred around the feature's predicted location $h_i = (u_d, v_d)$ and the vertical search range is $\pm 3\sqrt{S_i(1,1)}$ pixels (notation using Python's zero based matrix indices). The actual image search is performed by the OpenCV library, with the search region in the current camera frame set to this rectangular area using cvSetImageROI(). The NCC search is performed using cvMatchTemplate(). Suitable matches are both gated by a threshold ($score > 0.97$) and compared against the 3σ search ellipse. After attempting to match each of the q predicted features we find $r = [0, q]$ successful measurements.

The search ellipses are plotted in our implementation using the Cholesky decomposition of S_i. Figure 2 shows the 3σ search ellipse for a number of mapped features.

As noted above the feature image patches are grey-scale. We briefly attempted to use the separate RGB colour information with a similar NCC search on separate colour channels, however decided the colour-to-grey-scale image conversion had a minimal cost compared to the increase in measurement noise. We suspect this is an interaction with the "BGGR" colour Bayer mask in the Unibrain camera.

2.4 State Update & Correction

To complete an EKF iteration the state vector $\hat{x}_k^-$ and error covariance P_k^- predictions are updated using the r successful feature prediction and measurements. The Kalman gain K_k is first calculated:

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k \right)^{-1} = P_k^- H_k^T S_k^{-1} \quad (16)$$

The Jacobian H_k here is composed by vertically stacking each of the H_i values for the r successfully measured features into a $2r \times (13 + 3n + 6m)$ matrix. Note that H_k is the same as H_k^- from earlier with the rows for the unsuccessfully measured features removed. Similarly the $2r \times 2r$ innovation covariance S_k is the same as the $2q \times 2q$ matrix S_k^- given in 15 with the pairs of rows/columns from the unmeasured features removed.

The state estimate is updated using the Kalman gain and the measurement residual:

$$\hat{x}_k = \hat{x}_k^- + K_k \left(z_k - h\left(\hat{x}_k^-, 0 \right) \right) \quad (17)$$

Where z_k is the r successful feature measurements $z_i = (u, v)^T$ stacked vertically and $h\left(\hat{x}_k^-, 0\right)$ is the feature predictions $h_i = (u_d, v_d)^T$ also stacked vertically ($2r \times 1$ vectors). The final EKF step updates the error covariance:

$$P_k = (I - K_k H_k) P_k^- \quad (18)$$

The quaternion component q^{WC} in the state vector $\hat{x}_k$, representing the camera's orientation, has a redundant dimension. As a part of the EKF update this should be normalised with $q_n^{WC} = \frac{q^{WC}}{|q^{WC}|}$. The error covariance P_k must be updated also with $P_k = N P_k N^T$, where block matrix $N = \begin{bmatrix} I_{(3\times3)} & 0 & 0 \\ 0 & Q_{(4\times4)} & 0 \\ 0 & 0 & I \end{bmatrix}$, and having the same dimensions as P_k, and Q the Jacobian $\frac{\delta q_n^{WC}}{\delta q^{WC}}$.

2.5 EKF Initialisation

A scale ambiguity exists when a visual SLAM system is initialised with no features. The scale and global coordinate system becomes defined as the camera and initial features are added to the map. This ambiguity arises since the distance to features when they are first observed is unknown. To illustrate this, a certain amount of parallax α observing a set of features might indicate the camera has moved 1m toward features that are 10m distant, or 2m toward features that are 20m distant. Furthermore, without additional sensor data, the camera's orientation is indeterminate.

Recent work by Civera et al. [6] has shown a dimensionless monocular visual SLAM formulation. However in the majority of visual SLAM implementations (including this one) the EKF initialisation issue is avoided by providing a visual *initialisation target* of known dimensions. Typically 4 or more planar features (e.g. the corners of a 25×20cm printed black rectangle) are assumed to initially be in front of the camera. These features are initialised in the EKF map with zero uncertainty and covariance. The global coordinate system W aligns itself with the target, making it simple to align the map and the camera's workspace.

Rather than use a separate initialisation target, our implementation initialises itself using the checkerboard pattern used in lens calibration. The "corners" are extracted using cvFindChessboardCorners() and refined using cvFindCornerSubPix(). The OpenCV function cvFindExtrinsicCameraParams() and these corners are then used to produce

a very accurate estimate for the camera position r^W and orientation q^{WC}. These are used to initialise the camera state with a position uncertainty of $\sigma_{rW} = 5mm$. The camera's linear and angular velocities are initialised to zero. The map is initialised by adding a feature for each of the 4 outermost corners of the checkerboard, parametrised in XYZ with zero uncertainty and covariance.

2.6 Feature Initialisation

Once the visual SLAM EKF system is online, the 4 initial features serve to keep the camera reasonably well localised. After each EKF cycle if the system predicts less than 12 features to be visible it will attempt to find and initialise a new feature. If a suitable image "corner" is found, the new feature is initialised immediately using IDP, contributing to the EKF in the next iteration. Feature detection and initialisation are described below.

New features are located using the Shi and Tomasi corner detector [24] as implemented in the OpenCV library. To ensure features are well distributed over the current camera image and to minimise search time we divide the current camera image into a 8×6 grid. Over several attempts we randomly pick one of the 48 grid squares until we find one where its bordering squares and itself contain no predicted features. This empty grid square is then searched using cvCornerMinEigenVal(). If the highest scoring pixel location is above a threshold (currently 0.005 for our laboratory and camera) then a new feature is added from the 13×13 pixel image patch centred on this location.

The measured pixel coordinates of the new feature $z_i = (u_d, v_d)^T$ must first be undistorted to retrieve the normalised image coordinates $z_p = (x, y)^T$. This is the inverse of the parametric distortion equation 13 and cannot be performed analytically. Given the relative infrequency of this undistortion calculation, we opt to approximate the inverse iteratively. Readers are referred to the source code for the internal OpenCV function cvUndistortPoints(). For our wide-FOV lens we have determined that at least 10 iterations are required for an adequate approximation. In the appendix to [27] Sol et al show how to calculate a least-squares optimal solution for the coefficients of an undistortion equation of similar form to distortion equation 13.

The undistorted normalised image coordinates $z_p = (x, y)^T$ describe the pin-hole projection of the feature onto a plane at $z = 1$ in the camera's coordinate frame C. Using this we can define a 3D ray z^C extending from the camera's origin r^W and through the feature, giving $z^C = \begin{bmatrix} x & y & 1 \end{bmatrix}^T$ in the camera's coordinate frame C also. The direction of this 3D ray in the world frame W is simply calculated using $z^W = R^{WC} z^C$, where R^{WC} is the matrix representing the quaternion rotation q^{WC}. Given the 3D ray in the world coordinate frame, the new feature's azimuth $\theta_i = atan2\left(z_x^W, z_z^W\right)$ and elevation $\phi_i = atan2\left(z_y^W, \sqrt{(z_x^W)^2 + (z_z^W)^2}\right)$ are trivially defined.

The new IDP feature $\hat{y}_i^{new}$ is added to the bottom of the state vector $\hat{x}_{k|k}^{new}$ before the next EKF iteration:

$$\hat{y}_i^{new} = \begin{bmatrix} r_x^W & r_y^W & r_z^W & \theta_i & \phi_i & \rho_{init} \end{bmatrix}^T \tag{19}$$

The state error covariance $P_{k|k}^{new}$ is constructed:

$$P_{k|k}^{new} = J \begin{bmatrix} P_{k|k} & 0 & 0 & 0 \\ 0 & \sigma_{pixel}^2 & 0 & 0 \\ 0 & 0 & \sigma_{pixel}^2 & 0 \\ 0 & 0 & 0 & \sigma_\rho^2 \end{bmatrix} J^T \tag{20}$$

Where J is composed of Jacobians:

$$J = \begin{bmatrix} I & | & 0 \\ --- & + & ---- \\ \frac{\delta y_i}{\delta \hat{x}_k} & | & \frac{\delta y_i}{\delta h_i}, \frac{\delta y_i}{\delta \rho} \end{bmatrix} \tag{21}$$

Like the other Jacobians in this EKF implementation these partial derivatives are complex yet tractable.

In our experiments we found $\rho_{init} = 0.25$, to be adequate, representing an initial feature depth $d = \frac{1}{\rho_{init}} = 4m$. For the initial inverse depth uncertainty we used $\sigma_\rho = 0.25$, ensuring $\rho_i = 0$ (infinite distance) is well within the uncertainty bounds. As reported in [8] these exact values are somewhat unimportant.

2.7 Feature Map Maintenance

To help keep the EKF execution time within the real-time constraints, features are occasionally removed from the map. Removing a feature involves deleting the appropriate 3 or 6 elements from the state vector $\hat{x}$, and slicing the corresponding 3 or 6 complete rows and column from the error covariance P.

In our implementation a feature is removed if it is not successfully measured in the 3 camera images following initialisation. This ensures that the feature must remain in view for more than 0.1 seconds and is not pixel noise or interference . Features are also removed if they have been predicted to exist in the camera image more than 10 times, and then subsequently failed to be measured in more than 50% of attempts. This acts to remove badly localised features, such as the edge a round object, or the visual intersection of two edges. There are many possible improvements to these algorithms, such as considering the parallax angle between observations to further identify badly localised features (a feature observed over a large parallax angle should be well localised).

3. SOFTWARE IMPLEMENTATION

Our software implementation uses an array of open-source software (OSS) components. In this section we give an overview of how we have used Python [22] to integrate the Scientific Python (SciPy) libraries, OpenCV [21, 1] and GTK+ to create a multi-threaded real-time prototyping environment. We have developed the system on an Ubuntu Linux workstation and written the code using the Pydev plug-in within the Eclipse development environment.

3.1 Software Components

Python is a popular high-level programming language. We chose to use it for many reasons, including its concise notation, untyped syntax and powerful inbuilt objects (lists, tuples, etc). We chose the CPython 2.6 implementation due to its wide support and easy integration into external C/C++ code and libraries. We plan to use other CPython optimisations in the near future such as Pyrex/Cython and the JIT-like compilation module Psyco. We note our plans to also use Google's "unladen-swallow" project as it matures. It is based on CPython and promises significant speed increases. Of particular interest here is the project's near-term goal to remove the GIL, which we are currently working around, to allow completely multi-threaded Python code.

Our Matlab-like prototyping environment is created by importing functionality from the Python/SciPy modules:

- *Numpy* defines high-performance N-dimensional arrays and basic linear algebra. The numpy.matlib module defines alternate matrix functions that operate in a more Matlab-like way ($A*B$ performs $dot(A, B)$). Most Numpy functions call underlying C implementations and libraries such as LAPACK/BLAS.

- *Matplotlib* defines a rich array of Matlab-like plotting routines. The GTK "canvas" allows plots to be buffered for smooth animations. We also note here that the plot canvas can readily be changed to output postscript files.

- *Ipython* provides an embedded interactive console in the main GUI. This allows snippets of Python code to be executed during real-time operation permitting arbitrary inspection of platform internals.

While OpenCV v1.1 provides a standard SWIG wrapper for Python, our attempts to use them in a multi-threaded Python environment failed. The standard wrapper does not release the GIL while executing external OpenCV code. After modifying the the OpenCV wrapper source to release the GIL, we obtained multi-threaded behaviour however experienced intermittent crashes.

To achieve reliable multi-threaded behaviour, we used the Python "ctypes" interface to directly interface the OpenCV library. The ctypes interface releases the GIL by default. More recently members of the OSS community have implemented a "ctypes-opencv" interface as a Python module.

To provide a capable modern GUI we used the cross-platform GTK+ toolkit. The GUI template file was rapidly prototyped using the drag-and-drop Glade editor. The Python PyGTK module provides complete access to all on-screen objects, including direct access to pixel-buffers for video rendering. The initial GUI design is shown in figure 2.

3.2 Design Overview

The platform has been designed with an object oriented (OO) paradigm and made to support real-time multi threaded execution.

The platform starts when the first thread (the GUI thread) calls the gtk.main() function. From here the GUI is built according to the Glade template, and various GTK call-backs are automatically connected to their underlying Python code. The GTK onIdle() call-back is then repetitively called by the GUI thread to maintain on-screen elements, such as status updates, plotting charts and blitting the camera frame-buffer.

We have created several Python "worker" thread classes that perform simple tasks in parallel across multiple processor cores. One such class uses the OpenCV cvGrabFrame() and cvRetrieveFrame() functions to capture sequential images from the camera and maintains a circular frame-buffer. Another such worker class track the calibration chessboard in real-time using OpenCV functions.

3.2.1 Camera Calibration

Camera calibration is performed using OpenCV library functions. When the "start" button is pressed in the GUI tab for camera calibration, a Python call-back creates 3 worker thread objects. The first captures camera images, the second tracks the calibration chessboard and the third repeatedly selects groups of the calibration chessboard "corners" and executes the OpenCV cvCalibrateCamera2() function. This design allows 2 CPU cores in an Intel Core2 Duo to be run at 100% load with the GUI and capture threads using 8% and 4% of the third and fourth cores, respectively. A "save" button in the GUI causes the the calibration parameters to be written to disk.

3.2.2 Visual SLAM

The EKF implementation as described throughout section 2 is contained in another worker thread class. Figure 2 shows a snapshot of the GUI while the system is running. It displays status information and plots, and allows the thread to be started, single-stepped and stopped. Our initial EKF implementation was written as a large loop within this one class. In our opinion it is quite easy to follow and the linear algebra implemented in Numpy/Python code is very readable.

4. RESULTS

4.1 Checkerboard Pattern

While developing the EKF implementation we performed tests on the same A4 sized checkerboard pattern used for the camera calibration. This provided us with 48 planar visual features in a known 3D configuration. The system was initialised as described in section 2.5, providing the 4 outermost corner features as XYZ features with zero uncertainty. From the 44 remaining checkerboard "corners", up to 20 of them were then initialised into the EKF map as IDP features using the procedure given in section 2.6.

In our tests, the camera remained highly localised within 20cm to 1m from the checkerboard. Between 1 and 1.5m the camera became less localised, and at over 1.5m the 3σ feature uncertainty ellipse allowed incorrect feature matches and EKF tracking was lost. Directly after initialisation a 0.5m sideways movement was enough to cause each IDP feature estimate to rapidly converged around the known value.

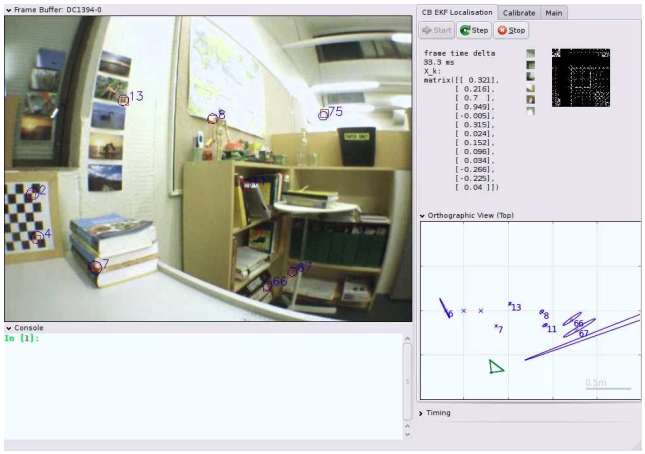


Figure 2: Screen-shot of our visual SLAM implementation mapping a small laboratory environment. Feature search ellipses are shown on the camera image in blue. The error covariance matrix P_k is visualised in the top right. A top-down orthographic view of IDP feature location uncertainties is shown on the lower right. The interactive Python console is in the lower left. Execution time plots are collapsed in the lower right.

4.2 Laboratory

While prototyping the system we performed many real-time tests, including the complete EKF implementation as described in section 2. Tests have been performed within a large laboratory, however limiting camera motion to a small area. The current area limitation arises because of the small number of features (about 20) that can be mapped in real-time.

We start the system holding the camera within 1m of the checkerboard pattern with it inside the camera's view. The EKF initialises and immediately begins tracking the camera using the four XYZ corner features. Over the next several iterations 8 or more new IDP features are added to the map. With minimal "SLAM wiggles" [17] the IDP feature locations begin to converge. Currently we can maintain enough map features to move the camera such that the original four XYZ features can pass well out of view. At this point providing the EKF is maintaining real-time operation the camera can be panned back to the start or panned anywhere within the already observed workspace. Figure 2 shows a screen-shot of the system in operation. Given the real-time nature of this work, video sequences are far more informative. Videos of the system in operation are available on the author's website: http://www.rrfx.net/2009/08/yanchep.html

5. DISCUSSION & CONCLUSIONS

The goal of this research was to create a highly capable real-time prototyping system while using it to reproduce the latest visual SLAM research. We have successfully shown a visual SLAM EKF implementation operating in real-time within a Matlab-like prototyping framework. The system is constructed using free and open source software components. Each component is cross-platform compatible, with the system having been developed using Ubuntu Linux and also tested on Windows XP.

5.1 Comparison

It is difficult to make a direct comparison to the performance reported by Civera et al. in their C++ implementation [8] without being able to execute their code and image sequences on the same hardware. We subjectively evaluate our current system to be several times slower. Their system is reported to handle 50 IDP features in real-time at 30Hz, compared to our 20. Factors affecting this difference include the frame-buffer size (they use a 320×200 pixel monochrome camera, while we use a 640×480 pixel colour camera) and processor speed (we use an Intel 2.6Ghz Core2, while they report using a less powerful 1.6Ghz Pentium M processor). A large factor making our system slower is the computational cost of the byte-code interpreted Python language and our general-purpose math libraries, compared to specialised C++ versions.

From a research-driven perspective it is difficult to quantify the difference in software development effort required to produce our system compared to an optimised C++ implementation. For many the performance loss may well be worth the increase in productivity. Developing within our Python implementation has a certain "bomb-proof" prototyping feel to it. Adding a new capability would take no more effort than it might to implement a new Matlab script.

5.2 Future Work

We have not spent much time profiling and optimising our system. We expect to be able to maintain several more real-time map features after some basic optimisations. An important feature currently missing from our implementation is the conversion of features from IDP to XYZ representation [7], this will provide a small performance boost by reducing the EKF state size. Further optimisations, as discussed in section 3.1, could easily double or triple the maximum real-time map size. The the EKF currently runs in a single-thread, however much of the code could actually be parallelised. Python would be well suited to this task if we can continue to overcome limitations of the GIL. We estimate at least doubling the map size again in a parallelised version.

To make our implementation more robust, we plan to incorporate other capabilities such as re-localisation [31], so the system is able to recover from tracking failures. Either richer richer feature descriptors [2] or estimating the orientation of planar feature patches [18, 14] could further improve feature tracking. Once the system can map 50 features in real-time, implementing an EKF sub-mapping techniques [23, 9] would be highly desirable to make the system scale to much larger maps.

6. REFERENCES

[1] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 2008.

[2] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway. Real-time and robust monocular SLAM

using predictive multi-resolution descriptors. *Lecture Notes in Computer Science*, 4292:276, 2006.

[3] Z. Chen, J. Samarabandu, and R. Rodrigo. Recent advances in simultaneous localization and map-building using computer vision. *Advanced Robotics*, 21(3):233–265, 2007.

[4] A. Chiuso, S. Soatto, P. Favaro, and H. Jin. "MFm": 3-D Motion From 2-D Motion Causally Integrated Over Time, Parts I & II. In *European Conference on Computer Vision*, pages 735–763, 2000.

[5] J. Civera, A.J. Davison, and JMM Montiel. SLAM using Monocular Vision. SLAM Summer School, 2006. Available online: http://www.robots.ox.ac.uk/ ˜SSS06/Website/Practicals.htm.

[6] J. Civera, A.J. Davison, and JMM Montiel. Dimensionless monocular SLAM. *Lecture Notes in Computer Science*, 4478:412, 2007.

[7] J. Civera, A.J. Davison, and JMM Montiel. Inverse Depth to Depth Conversion for Monocular SLAM. In *IEEE International Conference on Robotics and Automation*, volume 10, 2007.

[8] J. Civera, A.J. Davison, and JMM Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Transactions on Robotics*, 2008.

[9] L.A. Clemente, A.J. Davison, I.D. Reid, J. Neira, and J.D. Tardos. Mapping Large Loops with a Single Hand-Held Camera. In *Robotics Science and Systems*, 2007.

[10] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *IEEE International Conference on Computer Vision*, pages 1403–1410, 2003.

[11] A.J. Davison. SceneLib v1.0 Open-source SLAM Library, 2006. Available online: http://www.doc.ic.ac.uk/ ajd/Scene/index.html.

[12] A.J. Davison, Y. González Cid, and N. Kita. Real-time 3D SLAM with wide-angle vision. In *IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon*, jul 2004.

[13] A.J. Davison and D.W. Murray. Simultaneous Localization and Map-Building Using Active Vision. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, pages 865–880, 2002.

[14] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. MonoSLAM: real-time single camera SLAM. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, pages 1052–1067, 2007.

[15] H. Durrant-Whyte and T. Bailey. Simultaneous Localisation and Mapping (SLAM): Parts I & II. *Robotics and Automation Magazine*, 13:99–117, 2006.

[16] Evolution Robotics. ERSP 3.1 Robotic Development Platform, 2006. Available online: http://www.evolution.com/products/ersp.

[17] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.

[18] N. Molton, A.J. Davison, and I. Reid. Locally Planar Patch Features for Real-time Structure from Motion. In *British Machine Vision Conference*, 2004.

[19] JMM Montiel, J. Civera, and A.J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. In *Robotics: Science and Systems Conference*, volume 9, 2006.

[20] Open Source. OpenSLAM Project. Online: http://openslam.org.

[21] Open Source, Intel Corporation, and Willow Garage. OpenCV. Available online: http://opencv.willowgarage.com/wiki/.

[22] Open Source and Python Software Foundation. Python. Available online: http://www.python.org.

[23] P. Pinies and J.D. Tardos. Scalable SLAM building conditionally independent local maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3466–3471, 2007.

[24] J. Shi and C. Tomasi. Good Features to Track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[25] J. Sola. Multi-camera VSLAM: from former information losses to self-calibration. *IROS Visual SLAM Workshop*, 2007.

[26] J. Sola. 6DOF, EKF-SLAM toolbox for MATLAB, 2009. Available online: http://www.laas.fr/˜jsola.

[27] J. Sola, A. Monin, and M. Devy. Undelayed Initialization for Monocular SLAM. *LAAS*, 2008.

[28] J. Sola, A. Monin, M. Devy, and T. Vidal-Calleja. Fusing Monocular Information in Multicamera SLAM. *IEEE Transactions on Robotics*, 24(5):958–968, 2008.

[29] Unibrain. Fire-i Digital Board Camera Remote CCD. Available online: http://www.unibrain.com/Products.

[30] G. Welch and G. Bishop. An introduction to the Kalman filter. *University of North Carolina at Chapel Hill, Chapel Hill, NC*, 1995.

[31] B. Williams, G. Klein, and I. Reid. Real-Time SLAM Relocalisation. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[32] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 22(11):1330–1334, 2000.

Discriminative Appearance Descriptors for 3D Human pose recovery from monocular Images

S.Sedai, M.Bennamoun and D.Huynh
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {suman,bennamou,du}@csse.uwa.edu.au

ABSTRACT

In this paper we propose a novel discriminative appearance descriptor for 3D human pose estimation from monocular images. Our image-descriptor is based on the intermediate local appearance descriptors that we design to encapsulate local appearance context and to be resilient to noise. We encode the image by the histogram of such local appearance context descriptors computed in an image to obtain the final image-descriptor for pose estimation. We name the final image-descriptor the *Histogram of Local Appearance Context* (HLAC). We then use *Relevance Vector Machine* (RVM) regression to learn the direct mapping between the proposed HLAC image-descriptor space and the 3D pose space. To train and evaluate our approach, we used a synchronized video and 3D motion dataset. We compared our proposed HLAC descriptor with *Histogram of Shape Context* and *Histogram of SIFT* like descriptors. The evaluation results show that HLAC descriptor outperforms both of them in the context of 3D Human pose estimation.

Keywords: human pose estimation; local feature descriptors; surveillance; performance evaluation

1. INTRODUCTION

3D human pose recovery from an image or a video frame is a popular research area due to its applicability in areas ranging from biometric, character animation and human computer interaction (HCI) to visual surveillance. For example, in video-based smart surveillance systems, 3D poses can be used to infer the action of the subject in a scene. 3D human pose estimation from a single-view image is challenging because it involves not only estimating a large number of parameters but also resolving the ambiguous relationships between image features and 3D human poses. The choice of image feature descriptors plays a crucial role in resolving those ambiguities.

Most pose estimation systems use the silhouette of a human subject as image features [1, 2, 3]. Such features are insensitive to variations of the foreground and background texture, and they can only be obtained after the subject has been segmented from the image. However, a good segmentation can only be obtained by enforcing constraints on the environment, e.g. a plain background. Furthermore, silhouette features contain only information pertinent to the shape of the occluding body parts, which cause some silhouette features to give multiple pose solutions. For example, a particular silhouette may generate forward-backward ambiguity and additional motion information must be used to resolve such ambiguities [1]. Therefore, most recent research uses appearance based features [4, 5] as they do not require an explicit segmentation of the subject from an image and are more informative than silhouettes. For these reasons, they are more applicable to pose estimation from video, taken in outdoor environments or even in static images where silhouettes could not be extracted easily. However, appearance-based features are more sensitive to foreground/background clutter that does not contribute to the pose. As a result, the computed features become less distinctive and the image to pose relationship may become ambiguous (since it may produce quite different features for similar poses). The approach in [4] uses Non-Negative Matrix Factorization to filter out the background clutter from the appearance features and use regression to estimate the upper body pose. Methods such as [6] depend on feature selection based regression to select the useful local appearance features for pose estimation. The work in [5] used a learning technique to efficiently create a histogram of local appearance features to encode the image and the pose is estimated by regression.

In our approach, we design local appearance descriptors that are distinctive and resilient to noise by incorporating local context information and dimensionality reduction. We name our local descriptor as *Local Appearance Context* (LAC). We then use a histogram of such LAC descriptors computed in an image as our final image-descriptor. We name our image-descriptor as *Histogram of Local Appearance Context* (HLAC). Our HLAC image-descriptor is both distinctive and resilient to noise and is expected to reduce the ambiguous relationship between image features and their corresponding poses. We use *Relevance vector Machine* (RVM) regression to learn the mapping between HLAC image features and 3D poses. RVM has been used by [1] for pose estimation using *Histogram of Shape Context* (HoSC), a shape descriptor but it has not been used for pose estimation using appearance-based descriptors. We use RVM as our regressor because it is known to give better generalization performance. Figure 1 presents a block diagram showing our feature representation and RVM regression steps for training and testing our approach. We have compared the pose estimation performance of our proposed HLAC image-descriptor with the approach of [5] and with the HoSC and *Histogram of SIFT* (HSIFT) like descriptors and found that

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

our features outperform these techniques.

Details of our feature representation are presented in Section 2. Section 3 describes the Relevance Vector Machine Regression used for our pose estimation problem. Section 4 presents the evaluation results and Section 5 presents our conclusion.

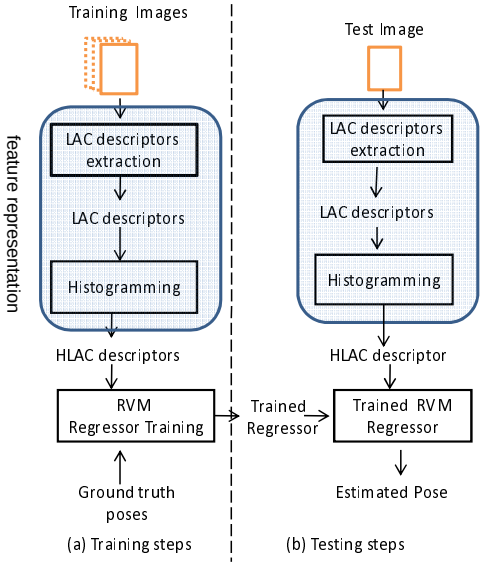


Figure 1: Block diagram showing the training and testing steps for our pose estimation algorithm

2. HISTOGRAM OF LOCAL APPEARANCE CONTEXT (HLAC) DESCRIPTOR

Our feature representation is based on two steps. First our proposed *Local Appearance Context* (LAC) descriptors are computed on the (regularly sampled) locations in an image window containing human subject. Subsection 2.1 covers the design and 2.1.1 covers the computational aspect of LAC descriptors. In the subsequent step, the histogram of such local descriptors is constructed, which is a HLAC image-descriptor as illustrated in Figure 2. The steps for constructing a HLAC image-descriptor from LAC descriptors are covered in Section 2.2.

2.1 Local Appearance Context (LAC) Descriptors

Our local appearance context (LAC) descriptor centered at a given point of the local image region is computed in two steps:

First, the local image region is partitioned into log-polar spatial blocks designated by n_r concentric squares resulting in $4n_r$ spatial blocks as shown in Figure 2. Our approximation of the log-polar partitioning leads to four square blocks around the center and $4(n_r - 1)$ L-shaped spatial blocks wedged between the remaining concentric squares. It makes the descriptor more sensitive to positions near to the center than to those far from the center.

Next, we compute the histogram of orientation of the image gradient in all of the $4n_r$ blocks. The number of orientation

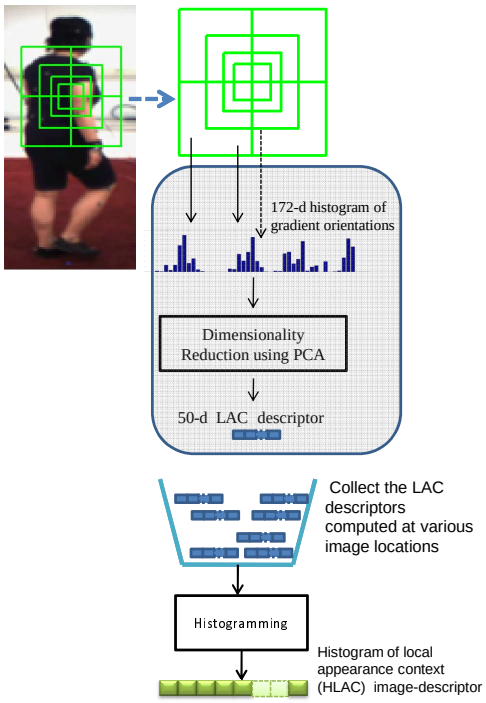


Figure 2: Block diagram showing how to compute LAC and HLAC descriptors. LAC is an intermediate local descriptor and HLAC is a final image-descriptor that encodes a image window containing a human subject

bins in the blocks that are far from the center are taken fewer than that in the locations that are nearby. This further allows the descriptor to be less sensitive to the positions far from the center than those nearby and hence encapsulates more informative context. For the pose estimation problem, we take $n_r = 4$ where the width of each square is taken in log-scale as $12, 20, 30$ and 48 pixels. The width of the innermost square is chosen such that it covers the limbs. The width of the outermost square is chosen to cover the width of the torso. The number of orientation bins for the blocks corresponding to each concentric square ring are taken respectively as $16, 12, 9$ and 6 given that the orientation of the gradient of the pixel takes the value between 0° to 360°. Hence the total dimension of the local descriptor becomes $(16+12+9+6)4 = 172$. The number of orientation bins for the blocks corresponding to inner concentric square is taken to be 16, to make the descriptor more sensitive to the pixels near center. The number of orientation bins for the blocks corresponding to outermost concentric square ring is taken to be 6, since those blocks are farthest from the center and we want our descriptor to be least influenced by the pixels in those blocks.

The orientation bins at each spatial blocks are then concatenated to form a single LAC vector of 172 dimensions as shown in Figure 2. The next step of the process is to apply the PCA to reduce the dimensions of LAC vectors down to 50. First, the PCA is performed on the covariance matrix of the LAC vectors computed on 5000 image windows containing human subjects to obtain the principal

feature space. Then the feature vector is projected onto the principal feature space and the 50 most significant components corresponding to the 50 eigenvectors with the largest eigenvalues are retained as our final LAC descriptor. The remaining components with lower eigenvalues are rejected, henceforth reducing the dimension of the LAC descriptors to 50. This makes the LAC descriptor more robust to noise in the image. Also, the LAC descriptor is a compact representation of the local image region as it encodes the local image characteristics as well as the surrounding context.

We would like to clarify the distinction between our LAC descriptor and the *Gradient Orientation and Histogram* GLOH descriptor [7] which also computes the gradient orientation histogram on the log-polar spatial sectors. Our feature is different from GLOH feature in two different ways: First, the spatial locations in our LAC descriptor is designated by the concentric square as opposed to a concentric circle in the case of the GLOH feature. We do this in order to efficiently compute the feature using the integral image which is presented in Subsection 2.1.1. Secondly, as opposed to GLOH descriptor where the number of orientation bins are taken to be the same for all spatial blocks, we allow the number of orientation bins for the spatial blocks that are far from the center to be sparser than those that are nearby the center. This makes our local descriptor more contextual as it is less effected by the image image pixels in the locations that are far from the center.

Here we re-emphasize that adding context information in the local descriptor helps the descriptor to be more distinctive. For example, when a local image region is corrupted by noise, using information from its surrounding region helps to disambiguate it. Moreover, similar looking local patches are often prevalent at multiple locations in the same image. However, their relationship with the surrounding locations is what makes them unique in encoding the pose of the human subject appearing in an image.

2.1.1 Fast Computation of Gradient Orientation Histogram

It is essential that the LAC descriptors are computed efficiently. We describe in this Section a method for fast computation of the gradient orientation histogram for the LAC descriptor using integral images. Given an image I and the spatial blocks $\{B_i\}_{i=1}^{4}$ inside each concentric square ring as shown in Figure 3, our goal is to compute the gradient orientation histogram in each of those blocks. Let N_b be the number of orientation bins to be used to construct the orientation histogram in each of the spatial blocks. First the gradients of the image I in the x and y directions are computed as

$$I_x(x,y) = I(x+1,y) - I(x-1,y), \tag{1}$$
$$I_y(x,y) = I(x,y+1 - I(x,y-1). \tag{2}$$

The magnitude and the orientations of the image gradient at a pixel location (x,y) can be computed as

$$M(x,y) = \sqrt{I_x(x,y)^2 + I_y(x,y)^2}, \tag{3}$$
$$O(x,y) = \tan^{-1}\left(I_y(x,y)/I_x(x,y)\right) \tag{4}$$

Then, given a number of orientation bins N_b, the k^{th} bin of the orientation histogram can take the orientation values in the interval $bin_k = [s_k, s_k + 2\pi/N_b]$ where $s_k = (k-1)2\pi/N_b$. The magnitude of the image gradient at pixel location (x,y) corresponding to the k^{th} bin can be expressed as,

$$M_k(x,y) = \begin{cases} M(x,y) & O(x,y) \in bin_k \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The value for the k^{th} orientation bin of the histogram, at the spatial block B_i is then defined by

$$b_{i,k} = \sum_{x,y \in B_i} M_k(x,y) \Big/ \sum_{x,y \in B_i} M(x,y)\ , \tag{6}$$

for $k = 1, \cdots, N_b$. Computing the summation of the gradient magnitudes inside the block is computationally expensive. For a faster computation of the orientation bins, we use integral images to calculate the summation in the numerator and the denominator terms of Eq. (6). In order to do so, we compute the integral image of each of M_k as $\mathbf{IM}_k$ where $k = 1, \cdots, N_b$ and integral image of the magnitude image M as $\mathbf{IM}$. We need to compute the gradient orientation histogram in two types of spatial blocks: square and L-shaped. If we take the corner points of a block B_i in a clockwise manner starting from the upper-left corner as $\{p_j\}_{j=1}^{N_p}$ where the number of corners, $N_p = 4$ for square blocks and $N_p = 6$ for L-shaped blocks as shown in Figure 3. Then, the summation terms in Eq. (6) can be computed as

$$\sum_{x,y \in B_i} M_k(x,y) = \sum_{j=1}^{N_p} (-1)^{j-1} \mathbf{IM}_k(p_j) \tag{7}$$

$$\sum_{x,y \in B_i} M(x,y) = \sum_{j=1}^{N_p} (-1)^{j-1} \mathbf{IM}(p_j) \tag{8}$$

In this way, the gradient orientation histogram can be computed quickly in each spatial block corresponding to each of the concentric square rings. The orientation histograms are then concatenated and dimensionality reduced to 50 to obtain the LAC descriptor as described in Section 2.1.

2.2 Histogramming LAC descriptors

The method of encoding an image with the distribution (or histogram) of local descriptors is commonly known as *bag-of-feature* modeling. It has been used in learning based pose estimation to construct the shape descriptor, e.g Histogram of Shape Context [1] and the appearance descriptor [5]. In this paper, we encode the image window containing the human subject as the distribution of the LAC descriptors com-

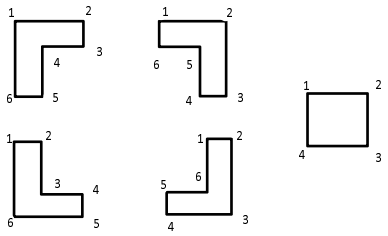


Figure 3: Square and different types of L-shaped blocks shows the clockwise numbering of points starting from the top left corner. The value of the integral image corresponding to the even corner is added and odd corner is subtracted to find the sum of pixel values inside a block.

puted in various locations of the image. The resulting image-descriptor is called Histogram of LAC (HLAC) descriptor and is constructed in two steps: First, the dictionary of the LAC descriptor is created by *k-means* clustering of the LAC descriptors computed on the local image patches extracted from a large number of images. The local image patches can be taken by regular sampling over image regions or by using the feature detectors. The centroids of the clusters $\mathbf{C} = (\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_K})$ together constitutes a feature dictionary. We found that a dictionary of size, K =200 gave the best performance.

In the second step, to construct the HLAC descriptor of the given image, L LAC descriptors are computed at points regularly sampled within an image as $V = (\mathbf{v_1}, \mathbf{v_2}, .., \mathbf{v_L})$. Then each descriptor is allowed to softly vote on each entry (bin) of the feature dictionary. For example, to vote a local descriptor $\mathbf{v_i}$ on each bin of the dictionary $\mathbf{C}$, one can place a Gaussian around the descriptor vector $\mathbf{v_i}$ and compute the posterior probability for each $\mathbf{c_j}$. As opposed to hard voting where each LAC descriptor is assigned to one of the bins, in soft voting, assignment is divided into bins according to the probability that the LAC descriptor is near to each bin. Soft votings reduce the effect of spatial quantization. Then, each voting is accumulated and normalized to get the final image-descriptor $\mathbf{x}$. The i^{th} element, x_i, of the HLAC image-descriptor $\mathbf{x}$ is computed as

$$x_i = \frac{1}{L} \sum_{j=1}^{L} e^{-(\mathbf{c}_i - \mathbf{v}_j)^T \mathbf{A}^{-1}, (\mathbf{c}_i - \mathbf{v}_j)} \qquad (9)$$

where A is the covariance matrix and $i = 1, \ldots, K$. We choose the common spherical covariance matrix A such that each LAC descriptor votes significantly to 5 to 10 bins of the feature dictionary. This produces a compact and distinctive HLAC image-descriptor of 200-dimension for the whole image.

3. POSE ESTIMATION

We use Relevance Vector Machine (RVM) Regression to learn the direct relationship between the proposed HLAC image-descriptor and the 3D poses. Given the training images, we first extract the HLAC descriptors from the image window containing the Human subject. Then we train the RVM regressor using the HLAC features and the corresponding pose vectors. Each pose is represented by a 31-dimensional pose vector, as described in the next Subsection.

3.1 Pose parameterization

We represent the 3D human body pose as the orientations of the ten body parts namely torso, head, upper (and lower) left (and right) arm (and leg). The orientation of the torso is taken with respect to the global coordinate system whereas the orientation of the other limbs are calculated relative to its adjoining limb as shown in Figure 4. The orientation of each body part is represented by the three Euler angle components.

The azimuth component of the torso angle θ covers 360° range and hence can wrap around 0° and 360° to be in that range. This causes discontinuity, for example, the regressor interprets the orientations whose values are around 0° and 360° differently, although they are similar. To preserve the continuity, we replace θ by $(a, b) = \cos(\theta), \sin(\theta)$ before regression as done by [1] so that it can be reconstructed later as $\theta = \text{atan2}(b, a)$ from the regression output. Hence, the orientation of the torso is represented by four parameters.

The 3D human pose is then represented by the orientations of all ten limbs i.e a 31-dimensional pose vector (4 for torso and 27 for other nine limbs) for regression.

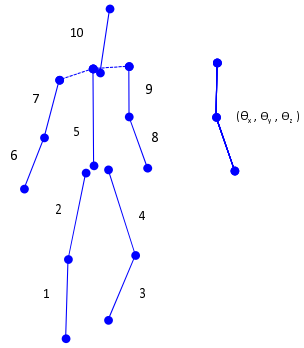


Figure 4: Parameterization of the 3D-Human Pose using relative orientations of each limb shown by solid stick. The orientation of each limb is represented by the three Euler angle components $(\theta_x, \theta_y, \theta_z)$

3.2 Relevance Vector Machine Regression

We formulate the pose estimation as a function approximation problem. We learn the relationship between the m-dimensional feature variable $\mathbf{x}$ and the d-dimensional pose variable $\mathbf{y}$ by direct regression from the training data samples, $\mathbf{T} = \{\mathbf{x_i}; \mathbf{y_i}\}_{i=1}^{N}$. Once the regression function $\mathbf{y} = F(\mathbf{x})$ is learned, the pose $\widehat{\mathbf{y}}$ for a new image feature $\widehat{\mathbf{x}}$ can then be estimated on the fly. Since the output pose is of d-dimension, we train an individual regressor for each output dimensions as $y = f(\mathbf{x})$. The overall pose regressor $\mathbf{y} = F(\mathbf{x})$ is then just a collection of each regressor trained to predict each output dimension. We model $f(\mathbf{x})$ using a Relevance Vector Machine [8] regressor because it is known to give a generalized solution. The relevance vector regressor is the extension of the linear model with the basis functions

taken as the kernels. Each kernel is associated with each of the data points from the training set. The relationship between $\mathbf{x}$ and each dimension y of $\mathbf{y}$ then takes the form

$$y = f(\mathbf{x}) = \sum_{i=1}^{N} w_i K(\mathbf{x}, \mathbf{x}_i). \quad (10)$$

Here we use Gaussian kernel given by,

$$K(\mathbf{x}, \mathbf{x}_i) = e^{-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}}, \quad (11)$$

where σ is the kernel width and it is estimated from the training data. The goal is to find the weight values $\mathbf{w} = (w_1, w_2, ...w_N)$ for each output dimension from the supplied training data $(\mathbf{X}, \mathbf{Y})$ such that the regression function, $F(\mathbf{x})$ generalizes to new observations. We train the RVM separately for each output dimension to obtain the set of weight vectors for each output dimension.

RVM adopts the Bayesian technique to find the weights. It starts by defining the Gaussian prior over the weight parameters as $p(\mathbf{w}|\alpha)$ which is controlled by the independent set of hyper-parameters $\alpha = (\alpha_1, \cdots, \alpha_N)$ corresponding to each weight. Then, the point estimate of the hyper-parameters are computed from the data by optimizing the marginal likelihood function $p(\mathbf{y}|\alpha)$, see [8] for detail. The posterior distribution of the weights is then directly computed from the hyper-parameter estimates. The author showed that, in general, the posterior distribution of most of the weights sharply peaks around zero, leading to a sparse solution. This implies that the weight values are zero for the most of the kernel bases centered at the training data that are redundant for the learning, and only a subset of the weights are non-zero. This leads to an automatic selection of the kernel-basis. As a result, only a subset of training vectors (only input feature) are enough to define the model. This produces a sparse model which is free from over-fitting and hence can generalize to unseen image observations. Moreover, by selecting a small number of relevant kernel-bases, a significant computational advantage is gained. Hence, this learning approach is suitable for pose estimation in real time. In our experiments, we found that only around 200 to 300 relevant vectors are selected out of nearly 1400 training samples.

4. EXPERIMENTAL RESULTS

We trained and evaluated the proposed pose estimation system using the HumanEva [9] dataset provided by Brown University. The dataset contains video frames and corresponding 3D poses of three subjects carrying out actions such as walking, jogging boxing and hand gesture. We use the walking sequences of the dataset in order to compare our result with [5] as well as to compare our features with HoSC and HSIFT features. The dataset was originally partitioned into training, validation and testing sets. However, the ground truth of the testing set is not provided, so we used the validation set (containing 1464 image frames) as our test set and the original training set (containing 1421 image frames) as our training set. The same set was used by [5].

The dataset provides the human pose for each video frame as 3D marker locations. We converted these 3D marker locations into the orientations of the ten body parts, where the orientation of each body part is calculated relative to its adjoining limb. Then, the pose is represented by a 31-dimensional vector as described in Section 3.1.

For both of our training and testing sets, we cropped an image window containing the human subject from each video frame and scaled it to a fixed size of 128×64 pixels. We used the camera calibration parameters and motion data provided in the dataset to crop an image window from the image frame. We assumed that, in a real scenario, such a cropped image will be provided by a human detector algorithm such as the one in [10].

To encode an image with a HLAC descriptor, we first extracted the LAC descriptors on regularly sampled locations in an image (every four pixels in the x and y image axes) by the process described in Section 2.1. In the second step, the histogram of these extracted LAC descriptors were computed by soft voting each LAC descriptor to each entry of the LAC feature dictionary as described in Section 2.2. The computed histogram was the HLAC descriptor to be used as the final image-descriptor of a whole image. Such descriptors are computed for each of the images in our training and testing set. (Note that the LAC feature dictionary consists of K LAC vectors obtained previously by k-means clustering the LAC vectors and is computed only once).

Similarly, we compared our image-descriptor with the image-descriptor computed from SIFT [11] like descriptors characterized by 4×4 square blocks where each block is of size 8×8 pixels. The number of orientation bins is taken to be 12, producing 192-dimensional local SIFT descriptor. Then, the *Histogram of SIFT* (HSIFT) descriptor of an image was obtained by soft-voting each of the SIFT descriptors computed on the local image regions to each of the K bins in the SIFT feature dictionary.

Also, the *Histogram of Shape Context* (HoSC) descriptor [1] was generated in a similar manner. We first extracted the silhouette of the human subject from an image using the background subtraction method proposed in [12]. We then computed the Shape Context (SC) descriptors [13] characterized by 12 angular bins and 5 radial bins on 400 regularly sampled regions in the silhouette. The histogram of shape context was then obtained by soft-voting each of the SC descriptors to each of the K entries of the SC feature dictionary. For all these cases we found the optimal size of the feature dictionary (K) to be 200. For $K > 200$ the performance of the pose estimation did not increase. The dimensions of HLAC, HSIFT, HoSC were therefore all set to 200.

We then trained the RVM regressor using the feature vectors extracted from the images and the pose vectors in the training set. We empirically found that a kernel width of $\sigma = 0.003$ for HLAC and HSIFT features and $\sigma = 0.01$ for HoSC features gave the best result. Other values of the kernel width around the optimal value gave similar result. We then predicted the poses of the images in the testing set using the trained regressor. The final output of the regressor is a 30-dimensional vector, which collectively denotes a particular pose. Since the ground truth poses of the images in the testing set are available, we computed the error between

Table 1: Comparison of RMS error of our approach(HLAC descriptor +RVM) with the approach of [5]

Errors	Ours	Nings [5]
Avg RMS	**5.22**°	6.68°
Global RMS	**4.41**°	5.75°

the estimated pose and the ground-truth pose to compare the performance of our approach with [5] and to evaluate our proposed feature with other state of art features. We computed the RMS absolute difference error between the estimated pose and the ground truth pose, which is also used in [5, 1]. The RMS error is given by,

$$E(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{d}\sum_{i=1}^{d} |(y_i - \hat{y_i}) \text{mod} \pm 180^\circ| \,. \qquad (12)$$

We found the average RMS error for our proposed HLAC descriptor to be as low as 5.22°. The comparison of our results with the results in [5] given in Table 1 shows that our method gives better performance. Also, we compared the RMS pose estimation error of our proposed HLAC descriptor with the HSIFT and HoSC descriptors. The comparison results in Table 2 suggest that our HLAC-descriptor gives the lowest RMS error over all limbs and for the global(torso) orientation. For example, we found a remarkable improvement of 3.18° over HoSC and 2.0° over HSIFT descriptors on the RMS error of the orientation of the global (torso) angle for a view corresponding to camera C1. For the other limbs, we found a small improvement between 0° to 1° compared to both HoSC and HSIFT as shown by the bar chart in Figure 5. Similarly, for the views corresponding to camera C2 and C3, our descriptor gives superior performance as shown by Table 2.

Figure 6 shows the comparison between the estimated angle and the ground truth of the torso of subject S1 completing one circular path for the HLAC descriptor in Figure 6(a) and the HoSC descriptor in Figure 6(b). The estimated pose from our HLAC image-descriptor is close to the ground truth whereas the HoSC descriptor gives many erroneous output. This is because the silhouette itself is not a distinctive feature; there may be multiple pose solutions for the same silhouette e.g foward-backward ambiguity. Our descriptor can resolve such ambiguities more effectively and hence can estimate the pose of the person more accurately. To evaluate our results qualitatively, Figure 7 illustrates the estimated poses of the corresponding input images as 3D body model. These results shows that the proposed HLAC image-descriptor is more distinctive and informative than both HSIFT and HoSC descriptors and can estimate the pose of a walking person more accurately.

5. CONCLUSION

In this paper, we propose a *Histogram of Local Appearance Context* (HLAC) image-descriptor for 3D human pose estimation in monocular images. We choose *Relevance Vector Machine* (RVM) regression as our learning algorithm to estimate the pose from the proposed image-descriptor. Experimental results show that using our image-descriptor along

Table 2: Comparison of RMS pose estimation error of our HLAC descriptor with HoSC and HSIFT descriptors for each view.

Features	HLAC		HoSC		HSIFT	
Cameras	Global	Avg	Global	Avg	Global	Avg
C1	**4.41**°	**5.22**°	7.6°	5.98°	6.46°	5.71°
C2	**3.11**°	**4.60**°	7.89°	5.53°	4.91°	5.55°
C3	**3.38**°	**4.65**°	7.07°	5.54°	5.42°	5.55°

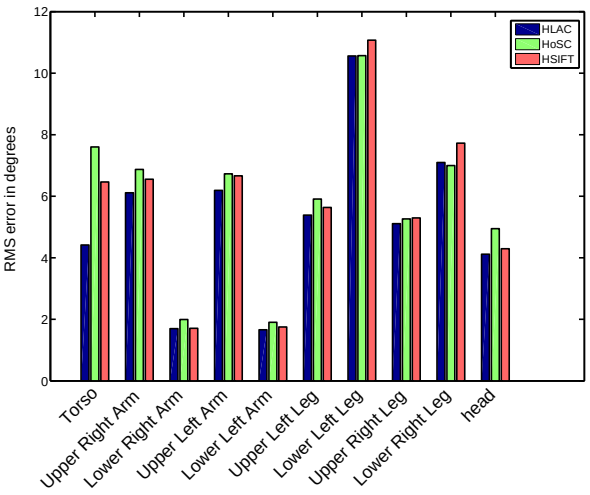
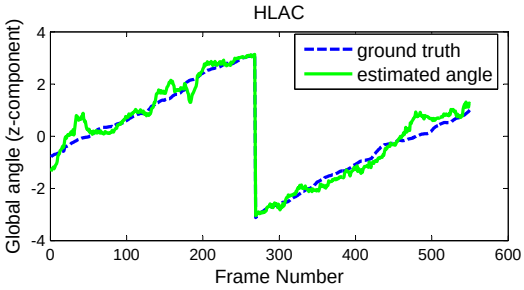
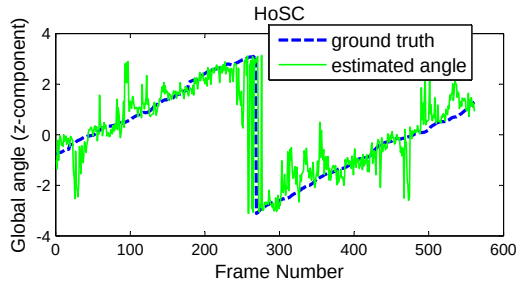


Figure 5: Comparison of RMS error of individual joint angles for each feature



(a) HLAC descriptor



(b) HoSC descriptor

Figure 6: Comparison of estimated torso angle with the ground truth on the test set of Subject S1 using (a) the HLAC-descriptor, (b) the HoSC descriptor.

with an RVM regressor gains better pose estimation performance than the approach of [5]. The performance of our proposed descriptor is found to be better than other state of the art descriptors like Histogram of Shape Context and SIFT like descriptors. Specifically, our descriptor gained significant improvement for the global orientation compared to both of these descriptors. Also, the results show that there is no forward- backward ambiguity during the pose estimation. This shows that our descriptor is more informative than both, HoSC and SIFT descriptor for 3D Human pose estimation. The success of our approach is due to the design of our feature which incorporates local appearance context and minimizes the effect of clutters. Our descriptor does not require background subtraction which makes it applicable to pose estimation in cluttered scenes. Our HLAC image-descriptor is created by histogramming local descriptors; there might be loss of spatial information between local descriptors that are useful for pose estimation. In future research, we will investigate on incorporating such spatial information in our HLAC image-descriptor to further improve the pose estimation performance.

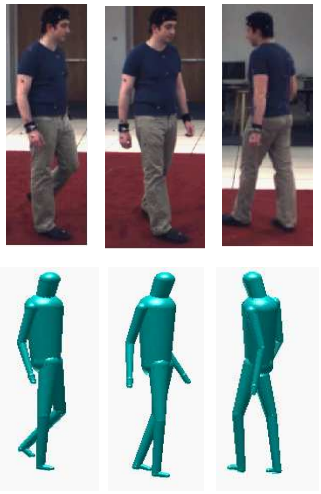
Acknowledgment

This work was partially supported by ARC grant application DP0771294. We would like to thank Brown University for providing the HumanEva dataset.

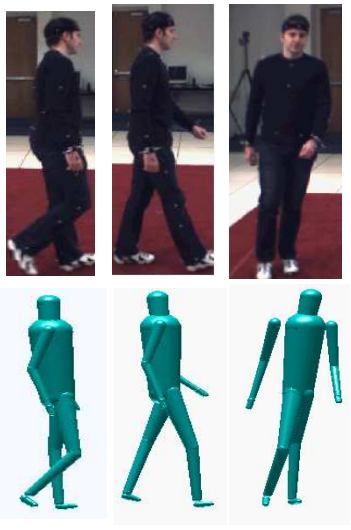
6. REFERENCES

[1] A. Agarwal and B. Triggs, "Recovering 3d human pose from monocular images," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 28, no. 1, 2006.

(a) Subject:S1



(b) Subject:S2



(c) Subject:S3

Figure 7: The input images from the test set and the corresponding output poses shown as 3D Human body model for Subject (a) S1, (b) S2 and (c) S3

[2] R. Rosales and S. Sclaroff, "Learning body pose via specialized maps," *NIPS*, pp. 1263–1270, 2002.

[3] J. Deutscher and I. D. Reid, "Articulated body motion capture by stochastic search," *Int'l Journal of Computer Vision*, vol. 61, no. 2, 2005.

[4] A. Agarwal and B. Triggs, "A local basis representation for estimating human pose from cluttered images," in *Asian Conference on Computer Vision*, 2006.

[5] H. Ning, W. Xu, Y. Gong, and T. Huang, "Discriminative learning of visual words for 3d human pose estimation," in *IEEE Conference on CVPR*, 2008.

[6] A. Bissacco, M.-H. Yang, and S. Soatto, "Fast human pose estimation using appearance and motion via multi-dimensional boosting regression," *IEEE Conference on CVPR*, pp. 1–8, 2007.

[7] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on PAMI*, vol. 27, 2005.

[8] M. E. Tipping, "The relevance vector machine," *Advances in neural information processing systems*, vol. 12, pp. 652–658, 2000.

[9] L. Sigal and M. J. Black, "Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion," Brown University, Department of Computer Science, Tech. Rep. CS-06-08, 2006.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on CVPR*, vol. 1, 2005, pp. 886–893.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of the IEEE*, no. 7, 2002.

[13] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. on PAMI*, vol. 24, no. 4, pp. 509–522, 2002.

Using NEAT for Continuous Adaptation and Teamwork Formation in Pacman

Mark Wittkamp, Luigi Barone
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {wittkamp, luigi}@csse.uwa.edu.au

Phil Hingston
School of Computer and Information Science
Edith Cowan University
2 Bradford Street, Mount Lawley, W.A. 6050, Australia.
email: p.hingston@ecu.edu.au

ABSTRACT

Despite games often being used as a testbed for new computational intelligence techniques, the majority of artificial intelligence in commercial games is scripted. This means that the computer agents are non-adaptive and often inherently exploitable because of it. In this paper, we describe a learning system designed for team strategy development in a real time multi-agent domain. We test our system in the game of Pacman, evolving adaptive strategies for the ghosts in simulated real time against a competent Pacman player. Our agents (the ghosts) are controlled by neural networks, whose weights and structure are incrementally evolved via an implementation of the NEAT (Neuro-Evolution of Augmenting Topologies) algorithm. We demonstrate the design and successful implementation of this system by evolving a number of interesting and complex team strategies that outperform the ghosts' strategies of the original arcade version of the game.

Keywords: Computational Intelligence, Neural Networks, Neuro-Evolution, NEAT, Real-time Learning, Teamwork Formation, Multi-Agent, Games.
CR Classifications: I.2.8 Artificial Intelligence: Problem Solving

1. INTRODUCTION

Games are often used as a test-bed to further the development of artificial intelligence (AI) techniques. Games are suitable in this respect because they involve similar problems to those encountered in real life, but are typically simpler and more clearly defined. They have a finite number of rules and actions for players to make and there is generally some well understood goal. Successful approaches to artificial intelligence in games can often be applied to similar real life problems. There are additional challenges facing players in video games, as opposed to traditional games (board games and card games for example). Video games generally have a far greater number of actions available for players to make and these actions also often have temporal significance.

While developing adaptive behaviour has been demonstrated using opponent modeling together with evolutionary algorithms [16, 17], the problem becomes much more complicated when we add the requirement for this to be done in real time as the game is played. In a relatively simple example, Quinn et al. [11] have witnessed the real time evolution of cooperative and coordinated behaviour for a team of robots; achieving the goal of moving to a new location while staying within sensor range of each other.

Contrast real time learning with offline learning, where artificial players practice (generally by playing games) to become better players for future games. When an artificial player is able to learn a strategy offline, the amount of CPU time available is near limitless because it is a one-off learning rather than an ongoing, real time process. One can allow the learning and fine tuning of artificial players to continuously run for days or even weeks.

For real time adaptation, the time allowed for learning is considerably less. Not only must the learning yield results quickly enough so that adaptive behaviour can be achieved while the game is being played, but only a portion of the CPU time will be available due to the game's own running requirements. Further still, computational intelligence techniques typically require many iterations and many more test cases for the evolution process to yield desirable results and so, speeding up of such techniques is of greater importance.

2. LIMITATIONS IN VIDEO GAME AI

Despite a large amount of research being undertaken in the field of video game AI, the majority of AI strategy in commercial games is scripted [2]. It is also difficult to implement game AI for complex games. Most developers resort to scripts because they are understandable, predictable, easy to modify and extend, and are usable by non-programmers [14]. While these scripts respond to the actions of human players, the behaviour will be the same time and time again. While game developers sometimes use adaptive learning techniques during development, the adaptive learning portion itself is rarely included in the released product — it is commonly used to determine some optimal parameters for a fixed playing strategy to use [3]. This commonly results in artificial opponents having weaknesses which, once discovered, are easily exploited. Predetermined behaviour also leads to repetitive and boring artificial players (be it opponents or team mates). While stochastic or deterministic systems can be employed to add some variety into the behaviour of the

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

artificial players, they generally offer only slight variation to some predetermined strategy. Too much variation from this strategy also has the potential for seemingly random or irrational behaviour. *Appearing* random may not necessarily imply an ineffective strategy, but it can adversely affect the human player's immersion in the artificial environment.

A limitation with a lot of current game AI is that the teams of opponents are all self-interested. While having a good individual may be useful for a team, it is very different from team-interested individuals who prioritise the good of the team over personal gain in every stage of their strategy. Without team based learning, artificial players will in some respect always be "greedy". The obvious limitation with self-interested behaviour is that no matter how well the individual parts may be tuned, certain team-strategies may never arise — a self-interested individual would never sacrifice himself to draw fire away from comrades, or to lead opponents into an ambush. Teams working together effectively can expect to complete tasks more efficiently. Team based learning may also prove useful where the goal to be accomplished is too large or complex to be achieved by individuals without team coordination — RoboCup soccer [8] for instance.

This paper explores the use of computational intelligence techniques for real time learning in the game of Pacman. Focusing on teamwork development, we examine how these techniques can be used to evolve strategies for the ghost agents in the game. Making use of continuous short-term learning to regularly update ghost strategies, we introduce a novel framework that parallelises offline strategy learning with actual game play; constant adaptation over short time periods meaning the ghosts do not need to learn complex general strategies that will be used in all game situations.

The rest of the paper is structured as follows. Sections 3 and 4 provided background material that further motivate the problem and introduce the salient features of the underlying technologies used. Section 5 introduces our learning system for the game, discussing how our approach can be used for real time adaptation and team strategy development. Section 6 details the AI used in the original arcade version of the game, reporting statistics that will be used as a baseline comparison for our work. Section 7 then reports on experiments with a number of different fitness metrics; results indicate that our system is able to evolve players that yield emergent team-work capable of superior performance to the AI used in the original arcade version. Finally, Section 8 summarises and concludes the work.

3. NEURO-EVOLUTION OF AUGMENTING TOPOLOGIES (NEAT)

Evolutionary algorithms (EAs) have been demonstrated to be a powerful tool for designing and training neural networks. Recently, Stanley and Miikkulainen have developed a new and powerful evolutionary algorithm called *Neuro-Evolution Through Augmenting Topologies* (NEAT) [12]. NEAT incrementally evolves the topology and weights of neural networks simultaneously.

The NEAT algorithm has been applied to a number of interesting problem domains, yielding some very impressive results. For example, the application of NEAT to the game of Othello saw the development of the sophisticated mobility strategy as a required step towards defeating alpha-beta search [1]. This strategy is considered to be very difficult to learn and typically requires a great deal of practice for humans to master. The development of this strategy, even at an intermediate level, suggests that NEAT has the ability to seek out weaknesses in opponents and develop strategies to exploit them [1].

NEAT not only offers a method of evolving both neural network weights and topologies, but it also has a number of features making it an attractive choice for our desired application area — that is, real time learning, where learning speed is important. While NEAT has been applied to real time learning in games in the past [13], most previous work has focussed on self-interested individuals and not team-based strategy development that we seek for this work.

In genetic programming [9] (an EA where individuals take the form of program trees), compatibility checks must be made before two parents' sub-trees can be subjected to crossover. This is to avoid the problem of a type mismatch; for example, an integer value may be passed to a boolean function. Another solution would be to overload all functions such that they can accept *any* input types, however this is less likely to yield *meaningful* crossovers. To encourage meaningful crossovers, NEAT maintains *historical markings* on an individual's genes that correspond to when the gene first came into existence; this value is inherited unchanged by an individual's children. This marking is used during crossover to dictate how individuals' genes should be combined to produce offspring. These markings also provide a useful means of encouraging meaningful crossovers without expensive topological analysis.

NEAT utilises *speciation* to avoid premature convergence to suboptimal solutions. Potential innovations are protected, giving newer structures a better opportunity to develop rather than being discarded early on in favour of existing, more developed, structures. This is done by allowing individuals to compete primarily against other members of their species rather than with the entire population. The number of offspring allowed per species is proportional to the average fitness of that species.

Bloat is the name given to the problem that results from individuals' structures becoming unnecessarily large (i.e. without an increase in performance), leading to slower execution of the EA and individuals that can not be further optimized. NEAT has a number of features to avoid bloat. The historical marking mechanism avoids the crossing of sections arbitrarily from highly varying structures. NEAT builds up from an initial population consisting of minimal networks without hidden nodes and structural complexity grows incrementally via mutation; complexity is only added if it yields a fitness advantage. NEAT's use of speciation/niching also aids in ensuring smaller structures are kept in the population as long as they are competitive. Should a species' members become bloated, these members will split off and form a new species. Only if this new species exhibits superior performance will the original species begin to die off.

4. THE GAME OF PACMAN

The human player's goal in the video game Pacman [7] is to navigate *Pacman* through a maze and progress to the next level by collecting (*eating*) all the *pellets* and *power-pills* in the maze (see Figure 1 for the default Pacman level map). There are four opposing *ghosts*, who attempt to stop Pacman from doing this by chasing him down and eating him. In the centre of the maze is a *hideout* area that Pacman is unable to enter. At the beginning of each level, one ghost begins just above this hideout while the remaining three venture out one after another every two seconds.

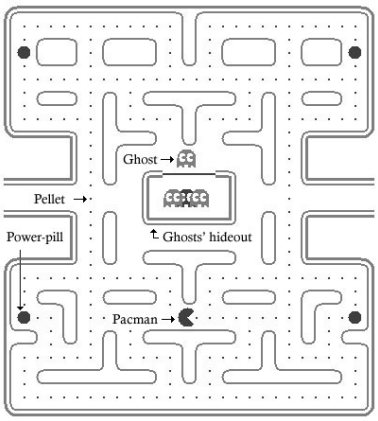


Figure 1: Default Pacman level map

Pacman begins the game with three lives and loses one each time he is eaten by a ghost. The game is over when Pacman runs out of lives.

Typically, the ghosts chase Pacman throughout the maze in an attempt to restrict his progress with the threat of consuming him and losing one of his lives. When Pacman eats a power-pill however, the situation is reversed and for a limited amount of time, Pacman is able to eat the ghosts and the ghosts cannot harm him. When eaten by Pacman, the ghost is consumed and loses its body, at which point it can not be eaten again nor pose any threat to Pacman until it is *restored*. The ghost must make a trip back to the hideout in order to restore itself. We define these three different ghost states as: *chasing* (pursuing Pacman), *fleeing* (evading Pacman when Pacman has consumed a power-pill), and *returning* (returning back to the hideout to be restored).

Other than a game of survival, there is also a point scoring system to Pacman. The consumption of pellets, power-pills, and the bonus items that appear on occasion throughout the game all give Pacman points. Eating ghosts also awards Pacman with points, increasing exponentially for each ghost eaten while still under the effects of that power-pill. When Pacman reaches certain scores, he is awarded an extra life. Despite its simplicity, the game of Pacman provides an interesting environment for potentially very complex team strategy development as the ghosts need to cooperate together to "trap" Pacman in order to kill him. In this paper, we work with a heavily modified version of Pacman based upon a Java applet version [4].

5. TEAM LEARNING SYSTEM FOR PACMAN

In this work, we aim to construct an environment where a number of ghosts are able to learn as a team in real time to exploit Pacman's weaknesses. The learning scheme for the ghosts is designed to work in parallel with the execution of the Pacman game, however, for this paper, we run it in "simulated real time" where we simply pause the game in progress and allow NEAT to take over. The work reported here is intended to be a proof of concept for a complete real time implementation of our system.

We focus our attention on developing strategies for the case where ghosts are either chasing or fleeing. We do not force our system to learn what to do when a ghost has been eaten (and must return to the hideout to be restored) — there are virtually no scenarios where reaching the hideout as fast as possible would not be the optimal behavior. The representation we use for ghosts is a feed-forward neural network which we evolve through the use of the NEAT algorithm — we have disallowed NEAT from producing recurrent links. Note that we are using the original NEAT algorithm, not the rtNEAT extension. In rtNEAT, population members are all active in the game area and have their fitness evaluated once their "lifetime" timer expires, at which time they may be replaced by a new child individual. We actually run four separate instances of NEAT with separate populations, only the best of which ever becoming active participants in the real Pacman game.

For the experiments reported in this work, we hand-coded a *pacbot* to play the Pacman game and act as a training partner to our team of ghosts. The pacbot is a decent player, capable of completing the first level almost every time when faced against the default ghost strategies (described in Section 6). The pacbot uses shortest path information for the level map and takes many factors into account, including distance to the nearest chasing and fleeing ghosts, distance to the nearest pellet and power-pill, how many points will be gained by consuming a particular ghost, distance to the bonus item, and how many points will be gained by eating this item.

5.1 Precomputed information

The Pacman levels are made up of a number of adjacent cells, each of which is either a pathway or a wall. Pacman, as well as the ghosts, pellets, power-pills, and bonus items may occupy these pathways.

We wish to concentrate on developing high level game play and avoid complicating the problem with lower level tasks such as navigating around walls, finding intersections, and so on. We are aware of a study that evolved a Pacman playing agent using a very small set of raw inputs and was able to produce a basic player, but its ultimate ability was hampered by having to learn how to avoid walls [5]. Our interest lies in the team-work and game strategies that can be evolved, rather than evolving ghosts from raw game data or minimal information.

The ghosts process the level map as a graph made up of a series of interconnected nodes that correspond to the pathway cells that the ghosts are able to reach. The initial positions of all pellets and power-pills are stored and updated in the environment model as the game progresses. We also precompute an all-pairs shortest-path table for every node in the level map, but we store only the length of these paths — storing the paths themselves would require a large amount of memory (the default Pacman level map contains 308 nodes with an average shortest-path length of over 21). Finally, we also allow our ghosts to have access to a precomputed table of shortest-path lengths from each cell to its nearest intersection — that is, any node that has more than two connecting nodes.

5.2 Learning structure

We use a neural network as the representation of an individual ghost strategy, which we evolve by running a series of simulation games in between the progression of the real game. We aim to have real time learning which learns only the specific team behaviours necessary to do well in the short-term, allowing for a high level of adaptivity which we hope will overcome the lack of a more generalised game playing strategy. Short-term learning also offers the

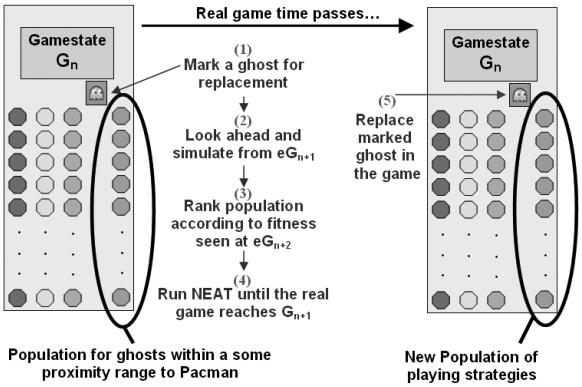


Figure 2: Pictorial representation of our learning system

advantage that any resulting bad strategies do not have a prolonged detrimental effect on the overall team's performance.

The NEAT algorithm provides a means of evolving the structure and weights of neural networks simultaneously and avoids the need for us to impose any sort of structural limitations upon the network topology. Using evolutionary selection pressure that rewards an individual ghost based on the performance of the team, we use NEAT to train our ghost strategies.

To allow for heterogeneous strategy development, we keep a number of populations in memory, divided up amongst the four separate ghosts. A simplistic approach might be to allow each ghost to have its own population, however, as relative distance to Pacman is more likely to decide what strategy to employ than a ghost's arbitrary colour, we use a population scheme based on a ghost's proximity ranking to Pacman instead.

In each time slice, a ghost is marked for training (cycling through the four different ghosts in turn) and then allowed to evolve from the population corresponding to its proximity to Pacman (i.e. closest, second closest and so on). A time slice is about as long as it takes for a ghost to travel a distance of 15 cells (approximately three to four seconds of game time). The selection pressure we use is based on the behaviour of the team as a whole, so we reward how well a particular ghost strategy works with the current team in place. Section 7 reports experiments with different fitness schemes; results showing that the fitness scheme has a major impact on the strategies evolved by the learning system.

We begin by initialising a number of randomly generated neural network populations, each corresponding to ghosts classified according to their distance to Pacman. Let G_{n} be the state of the game at the beginning of time slice n. During these time slices and at their boundaries, a number of tasks must be performed. The general ghost training algorithm is as follows (see also Figure 2):

1. Mark a ghost for learning during current time slice, beginning at G_{n}.
2. Look ahead (based on our models of the other ghosts and Pacman) and store the game state as we expect it to be like at the beginning of the next slice through simulated play ($eG_{\mathrm{n+1}}$). This will be the starting state for the NEAT simulation runs.
3. The fitness of a ghost strategy is determined by evaluating the game state that we expect to reach when the strategy is used in place of the marked ghost ($eG_{\mathrm{n+2}}$). This evaluation is an evaluation of the end state, and we experiment with various fitness schemes.
4. In parallel to the running of the actual game, run NEAT until the actual game reaches $G_{\mathrm{n+1}}$.
5. The best individual from the simulations is substituted into the game, replacing the marked ghost.
6. Repeat the process for the next ghost in turn.

Our aim in this work is to determine the feasibility of the proposed approach for learning in the game of Pacman. Instead of restricting NEAT for a limited amount of time (point 4 above), in this work, we allow the marked ghost to learn by simulating play with a perfect Pacman player model for a definite number of generations (20 generations). We maintain four separate populations (each of size 10) to allow for variation between ghost strategies. Our eventual aim is for this learning to occur in parallel, and hence will be limited by the time available while the game is in play (the time slice). For the moment however, we simply pause the game's progress and allow the simulations to run sequentially.

The accuracy of the estimated next game state will be closely related to the accuracy model of the Pacman player model. The estimations for these experiments will be very close to perfect because we are providing NEAT with a perfect model (the game itself has slight variation due to processor load and interleaving of events). Also, the ghosts are deterministic and have perfect knowledge of each others' behaviour and as such will not introduce any error into game state estimations.

5.3 Ghost neural network

We aim to evolve a neural network that acts as a move evaluator for the cells adjacent to a ghost. This approach has been successfully tried before, albeit to control Pacman rather than ghosts [10]. Our network is applied up to four times (once for each adjacent cell); the cell with the highest evaluated score is where the ghost will move to next. The neural network we construct will therefore only have one output value — the score of a cell. The advantage of this is that only four cells will ever need to be considered to make a move, and the network must only learn to output a single value. The score provides a measure of desirability of the resulting game state if the training ghost moved to the cell under consideration. In contrast with the original ghost AI, our network controlled ghosts are permitted to turn back on themselves and also make move decisions at every game cell position (not only intersections).

An approach where we evaluate only a ghost's adjacent cells may at first seem to be a very localised and "greedy" choice. However, when we consider that the inputs to the neural network contain precomputed high-level information as described in Section 5.1, it becomes clear that the few cells explicitly under consideration are capable of painting a much larger picture and encompassing information from many other cells.

Allowing our neural networks to receive their inputs as higher-level information allows complex behaviour to be very simply represented by the ghosts' neural network. For example, consider a neural network with a single input: the shortest-path distance to

Pacman. For a ghost to chase Pacman, all that would be required is a positive weight connecting this single input to the output. Similarly, to flee from a power-pill charged Pacman all that would be required is a negative weight connecting the input to the output.

We selected neural inputs to give the ghosts sufficient information to be used as fundamental building blocks for general ghost strategy development. There are 19 inputs in total, which we hope will be sufficient to allow for complex and diverse team strategies to develop. With this many inputs, we hope to maintain expressiveness in our evolved strategies while avoiding the other extreme of overwhelming the learning system with excessively large/noisy raw game data. Many of these 19 neural network inputs are based on information we found useful when constructing the pacbot player.

The full list of inputs is listed below. To keep the list concise, we first introduce some terminology. The inputs are written from the point of view of the ghost using the neural network as if that ghost was positioned on the cell that is currently being evaluated. We use the word $this$ to refer to the current ghost (the one whom the neural network is being used to control). Recalling the three different ghost states (chasing, fleeing, or returning), we define a function called $status(g)$ that returns $\{-1, 0, 1\}$ respectively depending on the state of ghost g. As the ghosts need information about their relative positions to key objects in the game, we use a function called $closest(t, o)$ that returns the closest object of a particular type t to the object o (e.g. $closest(Powerpill, Pacman)$ represents the closest power-pill to Pacman). We refer to the length of the (Manhattan) shortest path from points a and b as $dist(a, b)$.

Each evolving ghost strategy has access to the following information in the form of its neural network inputs:

1. $status(this)$
2. $status(closest(Ghost, this))$
3. $status(closest(Ghost, Pacman))$
4. $dist(this, Pacman)$
5. $dist(this, closest(Ghost, this))$
6. $dist(Pacman, closest(Ghost, this))$
7. $dist(Pacman, closest(Ghost, Pacman))$
8. $dist(this, closest(Powerpill, this)$
9. $dist(Pacman, closest(Powerpill, this))$
10. $dist(this, closest(Powerpill, Pacman))$
11. $dist(Pacman, closest(Powerpill, Pacman))$
12. $dist(this, closest(Pellet, this))$
13. $dist(Pacman, closest(Pellet, this))$
14. $dist(this, closest(Pellet, Pacman))$
15. $dist(Pacman, closest(Pellet, Pacman))$
16. $dist(this, closest(Intersection, this))$
17. $dist(this, closest(Intersection, Pacman))$
18. $dist(Pacman, closest(Intersection, this))$
19. $dist(Pacman, closest(Intersection, Pacman))$

5.4 Performance assessment

After a simulated game run has finished, the performance of the ghosts is evaluated as a whole and this group score is used to evaluate the variable component during that simulation — i.e. the ghost currently in training.

Ideally, we would like the fitness to be measured by some high-level means (for example, the amount of time Pacman managed to stay alive, or Pacman's achieved score). This would be desirable in some respects since it would minimise the amount of bias in our evaluation metric. The problem with having such a high-level evaluation metric is that most of the time there would be very little, if any, selection pressure for the evolutionary algorithm to use. If only the death of Pacman is given as positive feedback, a mostly flat fitness landscape results, with little search gradient to guide the evolutionary search. Useful feedback will only be forthcoming if a strategy is found that is good (or lucky) enough to eat Pacman in the first place — but it does not provide a means of comparing all the group behaviours that *failed* to eat Pacman.

The aim of a fitness function is to guide the evolutionary algorithm toward learning effective team strategies. We tried a number of evaluation mechanisms; these are outlined in the corresponding experiments in Section 7. Our selection pressure is a function of the team's fitness where lower values are better. The fitness functions are designed to reward individual ghosts based on the performance of the team they are a part of (global reinforcement), as opposed to directly evaluating an individual solely by their own performance (local reinforcement). We expect global reinforcement to produce more varied team behaviours since ghosts will learn strategies that not only serve to benefit their own situation but the team as a whole.

6. ORIGINAL GHOST AI IN PACMAN

In order to show the benefits of our learning system for use in evolving good ghost strategies, we compare the performance of strategies evolved using our approach to the strategies used in the original arcade game of Pacman. This section describes the main elements of the AI used to control the ghosts in the original version.

In the original Pacman game, the ghosts alternate between the two modes of play (*attack* and *scatter*) several times until eventually remaining in a permanent state of attack [6]. Each of the four ghosts has its own unique attacking strategy, and each heads toward their own "home" corner when scattering; the home corners are the top-right, top-left, bottom-right, and bottom-left corners for the red, pink, blue, and orange ghosts respectively.

When a ghost reaches an intersection, it decides upon a *target cell* — this target may in fact be off the level map, but this does not cause any problems. The ghost then moves in the direction to minimise its horizontal or vertical distance to that target (whichever is greater). When attacking, the red ghost's target is the current position of Pacman. The pink ghost's target is the cell four positions in front of Pacman. The blue ghost's target is the position such that the cell two positions in front of Pacman is the midpoint between it and the red ghost. The orange ghost will target Pacman when it is far away, but will target its home corner instead when within a vertical and horizontal distance of eight cells to Pacman. Ghosts revert to a pseudo-random behaviour when under the effect of a power-pill [15].

6.1 Performance of the Original Ghost AI

To measure the success of our learning system, we compare the performance of our evolved strategies against a simulation of the original Pacman ghost AI described above. Table 1 reports these results, averaged over 40 runs to compensate for the non-determinism in the game.

Table 1 reports two statistics about the performance of the pacbot

Table 1: Performance of the pacbot versus the original ghost AI

Score at end of first level:	4665
Lives lost during first level:	1.4

versus the original ghost AI: the score at the end of the first level, and the number of lives lost in completing the level — the level terminating when either the pacbot has eaten all pellets and power-pills or has died three times.

These results confirm the ability of the pacbot; the pacbot successfully completing the first level 39 out of 40 runs. If we continue the game to completion (repeating the level until the pacbot has lost all three of its lives), the pacbot obtains on average a final score of 7338. This score is roughly equivalent to that of a novice human player, but we should mention that the pacbot was designed to be a 'safe' player — its primary goal is to stay alive and pass the level; achieiving a high score is a secondary concern. Note also that the score obtained by the pacbot in completing the first level (4665) is significantly greater than the "baseline" score of 2700 (the score obtainable by just eating pellets and power-pills), indicating that the pacbot uses the power-pills to eat ghosts to maximise its score.

7. EXPERIMENTAL FRAMEWORK AND RESULTS

In this section, we report experiments with various fitness functions used in our evolutionary learning system and discuss our observations.

7.1 Experiment 1 - Chasing/Evading Pacman

In our first experiment, we aim to learn a simple chasing and evading strategy for the ghosts. We use an evaluation metric that primarily rewards killing Pacman and then a secondary reward for minimising the distance of chasing ghosts to Pacman and maximising the distance of fleeing ghosts to Pacman. We choose a total ordering on the separate rewards, using the secondary measure to resolve ties between solutions with the same performance on the first measure. That is, all solutions with an inferior performance on the primary measure obtain a lower fitness than solutions with superior performance on the primary measure regardless of performance in any subsequent measure. The metric we use is:

$$\begin{aligned} \text{rank 1}: &\quad \text{Pacman}'\text{s number of lives} \\ \text{rank 2}: &\quad \Sigma_{i=1}^{n} dist(Ghost_i^c, Pacman) \\ &\quad -\Sigma_{i=1}^{n} dist(Ghost_i^f, Pacman) \end{aligned}$$

where $Ghost^c$ denotes a chasing ghost and $Ghost^f$ denotes a fleeing ghost.

A comparison of the performance of the pacbot when played against our learning system (initialised with random behaviour) using this evaluation function and the original ghost AI is shown in Table 2. Reported results for our learning system are averaged over 25 independent runs.

These results show that using this evaluation metric, the pacbot obtained a lower score against the evolved strategies than it was able to against the original ghost AI. Using this performance metric,

Table 2: Performance of the pacbot versus the learning system for experiment 1

	Score	Lives lost
Original AI	4665	1.4
Experiment 1	3812	2.7

the ghosts seemingly have learned to successfully evade Pacman (thus reducing the score the pacbot was able to obtain from eating ghosts) but we did not see the increase of kills that we were hoping for — the number of lives taken by the evolved strategies was only marginally greater than the original AI at killing Pacman.

Upon inspection of the resulting ghost strategies, we observe that the ghosts do indeed learn to chase Pacman, vigorously pursuing the pacbot as it moves through the maze. We do note however that the ghosts often cluster together, typically within a few cells of each other, and often approach Pacman from the same direction. This clustering behaviour reduces the overall effectiveness of the team, effectively reducing the team of four ghosts into a team of one or two ghosts, reducing the ability of the team to "trap" Pacman by approaching from different sides (which is ultimately what is needed for a successful kill). An example of this behaviour is shown in Figure 3.

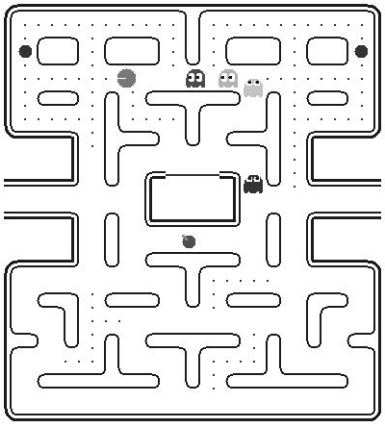


Figure 3: Clustering of ghosts as they chase Pacman

This clustering behaviour explains the reduced score obtained by Pacman — the pacbot is not surrounded by ghosts and hence is less likely to find a nearby ghost to eat after eating a power-pill (it must always proceed in the one correct direction to kill the ghosts instead of any number of directions that lead to a ghost if the ghosts are spread out). We also observe that being so clustered together, the ghosts have a reduced ability to kill Pacman (fewer effective ghosts makes trapping Pacman less likely). This seems to indicate that the system would benefit from a more directed fitness function that promotes dispersion among the ghosts.

Examination of the resultant ghost behaviours also highlights a beneficial feature of our approach. By using continuous short-term learning, the system does not need to learn complex general game-playing strategies that will be used in all situations in the game, and instead need only learn "simple" strategies suitable for the time slice of use. Indeed, since the fitness function evaluates the behaviour (phenotype) of the ghosts rather than the decision network

(genotype) that results in this behaviour, any strategy that manifests with good behaviour during the time slice in which it is to be used will suffice. That is, the search for a rewarding behaviour is made easier by the many-to-one relationship that exists between genotype and phenotype over a time slice.

For example, when learning continuously, a ghost never actually needs to evolve a specific genotype that explicitly encodes a "chase Pacman" strategy; all that is required is a genotype that (for the current time slice starting from this exact game configuration) behaves as if the ghost moves towards Pacman. An example of this behaviour is shown in Figures 4 and 5.

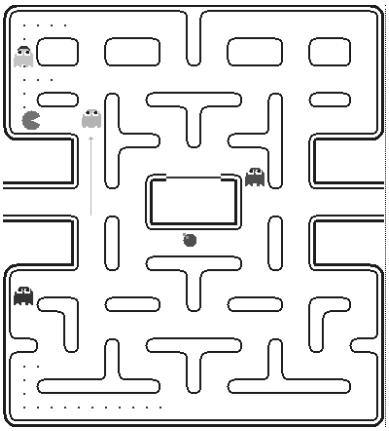


Figure 4: A ghost chasing Pacman during its time slice

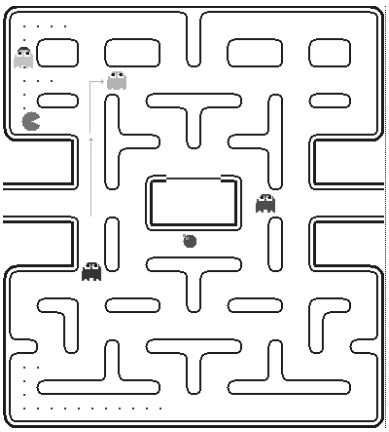


Figure 5: Continued game play of the ghost strategy from Figure 4

Figure 4 plots the trajectory of one ghost during a time slice. At first appearance, the ghost appears to be chasing Pacman. However, when we allow play the ghost beyond that time slice, it becomes evident in Figure 5 that the behaviour encoded by the strategy is no longer one of chasing Pacman, and actually represents some other strategy entirely. So, while the system may not be able to learn good general strategies for the game, through the use of continual adaptation, the system is able to "simulate" a good strategy without the need for offline learning.

7.2 Experiment 2 - Remaining Dispersed

In Experiment 1, we found that ghosts often cluster together, limiting their influence over the maze. This was seen to be undesirable behaviour since the ghosts were not often able to trap Pacman by attacking from different directions. To overcome this problem, we extend the performance evaluation metric to consider the dispersion of the ghosts. We reward strategies that maximise the distance between ghosts. As before, we use an evaluation function with a tiered approach over a number of separate metrics:

$$
\begin{array}{ll}
\text{rank 1}: & \text{Pacman}'\text{s number of lives} \\
\text{rank 2}: & Min^{n}_{i=1} dist(Ghost^{c}_{i}, Pacman) \\
 & -Max^{n}_{i=1} dist(Ghost^{f}_{i}, Pacman) \\
\text{rank 3}: & -\Sigma^{n}_{i=2} dist(Ghost^{c}_{i-1}, Ghost^{c}_{i}) \\
 & -\Sigma^{n}_{i=1} dist(Ghost^{f}_{i-1}, Ghost^{f}_{i})
\end{array}
$$

where $Ghost^c$ denotes a chasing ghost and $Ghost^f$ denotes a fleeing ghost.

Apart from killing Pacman, the evaluation function is designed to minimise the distance between Pacman and the closest chasing ghost while, at a lesser priority, maximising the distance between all pairs of ghosts in the same state (returning ghosts are ignored). The first tier evaluation is used to reward ghosts that kill Pacman. The second tier rewards how close the *closest* chasing ghost is to Pacman, and how far the *furthest* fleeing ghost is from Pacman. Note that this differs from Experiment 1, since a reward for minimising the distance to Pacman for *all* ghosts would counter-act our desire for dispersion. The third tier promotes dispersion directly by rewarding ghosts that are more distant from other ghosts of the same status than closer ghosts.

A comparison of the performance of the pacbot when played against our learning system (initialised with random behaviour) using this evaluation function and the original ghost AI is shown in Table 3. Reported results for our learning system are averaged over 25 independent runs.

Table 3: Performance of the pacbot versus the learning system for experiment 2

	Score	Lives lost
Original AI	4665	1.4
Experiment 1	3812	2.7
Experiment 2	4177	2.6

As is evidenced from the results reported in Table 3, this evaluation function produces a team of ghosts that is better at killing Pacman when compared to both the original ghost AI and the evaluation function used in Experiment 1. Indeed, dispersion appears to successfully promote attacks from different sides, enabling the ghosts to trap Pacman and obtain a kill. Dispersion does however come at a cost — an increase in the average score obtained by the pacbot during play.

As mentioned in the previous section, a network of disperse ghosts is easier for the pacbot to score against (consume) after eating a power-pill. This may seem counter-intuitive to those familiar with game (human players often attempt to force the ghosts to bunch together and allow for easier mass-consumption). In contrast to a human player, the pacbot does not view a bunch of ghosts as a hugely profitable target, but a minor goal relative to completing the level.

With more dispersed ghosts, it is more likely that the pacbot will encounter a ghost while collecting pellets and then proceed to consume a nearby ghost. In contrast, a tight cluster of ghosts require the pacbot to choose one of a limited number of "critical" paths that intersect with the cluster, increasing the likelihood for failure and scoring nothing. Of course, should the pacbot successfully intersect the cluster, the pacbot is richly rewarded, however experimental results indicate that the "conservative" situation scores better on average than the "higher-risk-higher-reward" situation.

As expected, the ghosts in this experiment managed to evolve a behaviour leading them to organise themselves in a dispersed manner, while minimising the shortest distance of the closest chasing ghosts to Pacman. As the fitness function only rewards chasing behaviour if a ghost is able to become the closest ghost, what tends to evolve is team behaviour in which the separate ghosts "sit" in very dispersed locations (the ghosts actually oscillate back and forth on the spot due to the game restriction that ghosts must remain in motion), with the single closest ghost actively pursuing Pacman through the maze. At first thought, this behaviour seemingly prevents the emergence of the desirable team-work behavior being sought (two or more ghosts closing in on Pacman from opposing directions) and appears counter to the results reported in Table 3. However, further analysis shows that team-work does indeed emerge, due in part to the way the learning system updates ghost strategies.

Recall from the discussion of Experiment 1, we outlined a problem with evolved strategies being kept in play for longer than their intended time slice. A side effect of this leads to desirable behaviour in this experiment — Pacman often becomes trapped by two ghosts approaching from opposite sides due to the "lag" in updating the different ghosts. An example of this behaviour is shown in Figure 6.

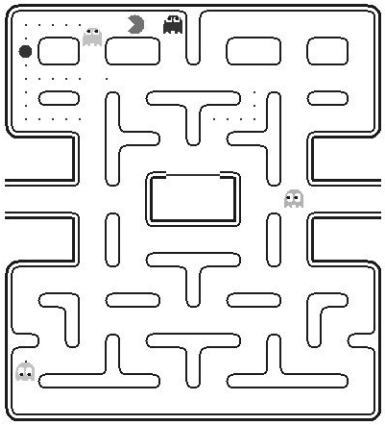


Figure 6: Ghosts trapping Pacman

This behaviour develops as follows:

1. One ghost is initially a chaser, with the remaining ghosts being content with maintaining dispersion.
2. A non-chasing ghost, formerly interested in maintaining dispersion, then "realises" that it can become the *new* closest ghost to Pacman and adopts a chasing-based strategy too.
3. Due to the delay in updating the previous closest ghost (recall the cyclic nature of updating the ghost strategies employed by the learning system), the old chasing ghost maintains a strategy of chasing Pacman.
4. The two chasing ghosts converge on Pacman and attempt to enact a kill.

This outcome is made even more effective because the chasing ghosts typically approach Pacman from opposite sides due to the previous positioning of ghosts via the dispersion reward.

Table 4: Performance of the pacbot versus the learning system for experiment 3

	Score	Lives lost
Original AI	4665	1.4
Experiment 1	3812	2.7
Experiment 2	4177	2.6
Experiment 3	3987	3.0

7.3 Experiment 3 - Protection Behaviour

The aim of this experiment is to see if we can evolve strategies where ghosts protect either a vulnerable team mate or a power-pill/pellet. To achieve this, we alter the evaluation metric for this experiment by rewarding strategies that minimise both the number of ghosts that have been eaten and the number of vulnerable (fleeing) ghosts:

$$
\begin{array}{ll}
\text{rank } 1: & \text{Pacman}'\text{s number of lives} \\
\text{rank } 2: & \text{count}(Ghost^r) \\
\text{rank } 3: & \text{count}(Ghost^f) \\
\text{rank } 4: & Min_{\text{i=1}}^{\text{n}} dist(Ghost_{\text{i}}^c, Pacman) \\
 & -Max_{\text{i=1}}^{\text{n}} dist(Ghost_{\text{i}}^f, Pacman) \\
\text{rank } 5: & -\Sigma_{\text{i=2}}^{\text{n}} dist(Ghost_{\text{i-1}}^c, Ghost_{\text{i}}^c) \\
 & -\Sigma_{\text{i=1}}^{\text{n}} dist(Ghost_{\text{i-1}}^f, Ghost_{\text{i}}^f)
\end{array}
$$

where $Ghost^c$ denotes a chasing ghost, $Ghost^f$ denotes a fleeing ghost, $Ghost^r$ denotes a returning ghost, and $count$ is a function that returns the number of objects of a particular type.

A comparison of the performance of the pacbot when played against our learning system (initialised with random behaviour) using this evaluation function and the original ghost AI is shown in Table 4. Reported results for our learning system are averaged over 25 independent runs.

The performance results reported in Table 4 indicate that this evaluation function does little to promote successful team-work that leads to killing Pacman. The ghosts are driven to minimise the number of vulnerable and fleeing ghosts, seemingly negating the effects introduced by the dispersion metric of Experiment 2.

We have however witnessed three examples of different protection behaviour emerging from the system, although they do not commonly occur due to the specific situations required for such an opportunity to arise:

1. A chasing ghost moving towards a vulnerable ghost to protect it from Pacman.

2. A vulnerable ghost moving towards a chasing ghost to protect itself from Pacman.

3. A chasing ghost moving toward a power-pill to prevent Pacman from eating it.

An interesting (and obviously unplanned) emergent behaviour that we observed was that of suicidal behaviour, where in certain situations, ghosts would run straight into Pacman! This occurred because the vulnerable ghosts learned that killing themselves could yield a reward if they were able to make it back to the hideout before the time slice was up. Similarly, we saw some strange shepherding behaviours where a training ghost would learn to behave in a way that would cause suicides in the other (fixed) ghost strategies, who obviously had their actions influenced by the actions of the ghost in training. This was an exciting result as it demonstrates the level of complex team behaviour that we hoped we would see, albeit misappropriated by a less than perfect fitness function.

7.4 Experiment 4 - Ambushing Pacman

Furthering the idea of trapping Pacman, in our final experiment, we aim to define a fitness function that attempts to directly this concept. To achieve this, we include the number of maze intersections that are "controlled" by Pacman — controlled in the sense that Pacman can reach the intersection before any chasing ghost can. We also only consider intersections within a threshold distance of 20 from Pacman, intersections further away than this are ignored. This controlled-intersection value captures the "freedom" of Pacman; the lower the number, the lower the number of options Pacman has in order to escape from being killed. Analogous to the discussion about consuming clustered ghosts after a power-pill, errors in the form of deviations from these "critical" paths lead to sub-optimal performance by the pacbot.

Using the successful evaluation function of Experiment 2, we add as a second tier metric a measure that explicitly captures this value, thus driving the evolutionary process to reward strategies that limit the number of escape routes available to Pacman. We additionally add a fifth measure to the evaluation function that captures Pacman's score, attempting to drive Pacman to the less valuable areas of the maze, thus prolonging the game and increasing the likelihood of forcing Pacman into a trap. The evaluation function we use in this final experiment is:

$$
\begin{array}{ll}
\text{rank 1}: & \text{Pacman}'\text{s number of lives} \\
\text{rank 2}: & \text{Intersections "controlled" by Pacman} \\
\text{rank 3}: & Min_{i=1}^{n} dist(Ghost_i^c, Pacman) \\
 & -Max_{i=1}^{n} dist(Ghost_i^f, Pacman) \\
\text{rank 4}: & -\Sigma_{i=2}^{n} dist(Ghost_{i-1}^c, Ghost_i^c) \\
 & -\Sigma_{i=1}^{n} dist(Ghost_{i-1}^f, Ghost_i^f) \\
\text{rank 5}: & \text{Pacman}'\text{s score}
\end{array}
$$

where $Ghost^c$ denotes a chasing ghost and $Ghost^f$ denotes a fleeing ghost.

A comparison of the performance of the pacbot when played against our learning system (initialised with random behaviour) using this evaluation function and the original ghost AI is shown in Table 5. Reported results for our learning system are averaged over 25 independent runs.

Table 5: Performance of the pacbot versus the learning system for experiment 4

	Score	Lives lost
Original AI	4665	1.4
Experiment 1	3812	1.7
Experiment 2	4177	1.6
Experiment 3	3987	2.0
Experiment 4	4092	2.0

The results reported in Table 5 clearly indicate that this evaluation function is strong in both maximising the kill-rate of Pacman and minimising the score the pacbot is able to obtain, strictly dominating all previous evaluation functions and the original ghost AI on both measures. In particular, we note the significant increase in the average number of lives lost by the pacbot in comparison to the original ghost AI, and unlike Experiment 2, we also observe a decrease in the average score obtained by the pacbot in completing the level.

For completeness, we also continued the runs of Experiment 4 to game end (repeating the level until Pacman dies three times), recording an average final score of 4260 for the pacbot in play against our learning system using this evaluation function. In contrast, the pacbot obtained an average final score of 7338 against the original ghost AI, further demonstrating the superior performance of our learning system over the original ghost AI. Indeed, these results show that our learning system produces more difficult opponents than the original ghost AI, at least for our hand-coded pacbot player.

From examination of the resultant evolved strategies, we observe that the ghosts tend to attack as a group (thus limiting the number of intersections controlled by Pacman), and importantly attack from different directions (again as a consequence of the desire to restrict escape routes, but also to ensure dispersion). Attacks are not as "directed" as Experiment 1 (the ghosts driven to remain disperse and therefore do not simply head straight for Pacman), but they appear more "structured", surrounding Pacman to limit escape routes before proceeding to ambush Pacman into a trap. Figure 7 gives an example of how the ghosts surround Pacman in order to restrict its number of escape routes.

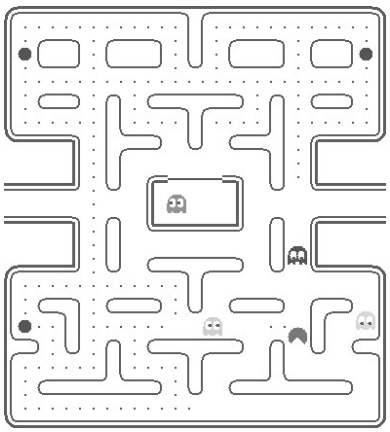


Figure 7: Ghosts blocking off escape routes

The evolution of structured team-work is an impressive result, with no specific information or expert guidance provided to the system to help organise the ghosts to work effectively together. Indeed, this emergent behaviour is exactly what is sought by this work — a system capable of learning and adapting to a player's style without direction from a human designer.

8. CONCLUSIONS

This paper has proposed a novel system for real time team strategy development using computational intelligence techniques for the game of Pacman. Our approach reduces the problem to be solved by the ghosts' neural network to one of game-state valuation — each ghost using their neural network to decide on the worth of each adjacent cell in order to select the next best cell to move to (i.e. the network acts as a *move evaluator*). This is made possible, in part, through the precomputed high-level data we make available to the neural networks.

As a proof of concept, we have implemented the system in simulated real time, with learning occurring in (pseudo) parallel to actual game play. Using continuous short-term learning, each ghost in the team is subjected to evolutionary selection pressure in order to learn a good strategy for use in the next time slice. Time slices are deliberately kept short, thus allowing the system to learn relatively simple strategies that only need to suffice for a short period of time and not complex general game-playing strategies that will be used in all game situations.

Our approach of having multiple distinct evolving populations depending on the proximity of the ghost to Pacman has allowed us to evolve very interesting and complex team behaviours with noticeable role development. Results show that our system is able to evolve players that yield emergent team-work capable of superior performance than the AI used in the original arcade version of the game.

Furthermore, our system has developed this team behaviour using a small population and a small number of generations for learning per time slice. This gives us confidence that our goal of implementing the system in true real time will show similar successes. It is expected that this work will lead to a fully functional implementation capable of delivering real time team strategy development in the near future. Future work will also examine the importance of the accuracy of the opponent model in learning, and explore ways of using a multi-objective approach for simultaneously evolving different strategies that satisfy the different metrics of interest without an explicitly stated ordering of such metrics.

9. ACKNOWLEDGEMENT

This research is partly funded by a postgraduate scholarship grant from the Australian Research Council.

10. REFERENCES

[1] Timothy Andersen, Kenneth Stanley, and Risto Miikkulainen. Neuro-evolution through augmenting topologies applied to evolving neural networks to play Othello. Technical report, Department of Computer Sciences, University of Texas at Austin.

[2] Lee Berger. Scripting: Overview and code generation. In *AI Game Programming Wisdom*, volume 1, pages 505–510. MIT Press, 2002.

[3] Darryl Charles, Colin Fyfe, Daniel Livingstone, and Stephen McGlinchey. *Biologically Inspired Artificial Intelligence for Computer Games*. Medical Information Science Reference, 2007.

[4] Benny Chow. Java pac-man implementation. `http://www.bennychow.com/play_pacman.shtml`.

[5] M. Gallagher and M. Ledwich. Evolving Pac-Man Players: Can We Learn from Raw Input? *IEEE Symposium on Computational Intelligence and Games*, pages 282–287, 2007.

[6] Marcus Hanshew. Pac-man faq/strategy guide, July 2006. `http://www.gamefaqs.com/coinop/arcade/file/589548/439591`.

[7] Toru Iwatani. *Pac-man*. Namco, 1980. `http://en.wikipedia.org/wiki/Pac-Man`.

[8] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. RoboCup: the robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 340–347. ACM Press, 5–8, 1997.

[9] John R. Koza. *Genetic Programming: On the Programming of Computers By Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[10] S.M. Lucas. Evolving a neural network location evaluator to play ms. pacman. *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 203–210, 2005.

[11] M. Quinn, L. Smith, G. Mayley, and P. Husband. Evolving teamwork and role allocation with real robots. In *In Proceedings of the 8th International Conference on The Simulation and Synthesis of Living Systems (Artificial Life VIII)*, 2002.

[12] K. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[13] Kenneth Stanley, Bobby Bryant, and Risto Miikkulainen. Real-time neuroevolution in the NERO video game. *IEEE Transactions of Evolutionary Computation*, 9(6):653–668, 2005.

[14] Paul Tozour. *The Perils of AI Scripting*. Charles River Media, Inc., 2002.

[15] Twin Galaxies Forums. Pac-man ghost behavior revealed, August 2008. `http://www.twingalaxies.com/forums/viewtopic.php?t=12231`.

[16] Mark Wittkamp and Luigi Barone. Evolving adaptive play for the game of spoof using genetic programming. In *Proceedings of the 2006 IEEE Symposium on Computational Intelligence and Games*. IEEE Publications, 2006.

[17] Mark Wittkamp, Luigi Barone, and Lyndon While. A comparison of genetic programming and look-up table learning for the game of spoof. In *Proceedings of the 2006 IEEE Symposium on Computational Intelligence and Games*. IEEE Publications, 2006.

An Efficient Approach Using Domain Knowledge for Evaluating Aggregate Queries in WSN

Chi Yang[#1], Rachel Cardell-Oliver[#2]
[#]School of Computer Science and Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia
[1]cyang@csse.uwa.edu.au
[2]rachel@csse.uwa.edu.au

ABSTRACT
We study the problem of evaluating aggregate queries over a set of observations made by nodes in a wireless sensor network. The novelty of our solution is to use domain knowledge about the possible values of observations in order to reduce the number of observations required to evaluate the query. Reducing the number of observations required is important, because it reduces the energy cost and latency of answering the query. We present two protocols for partial and opportunistic aggregate performance. Opportunistic use of domain knowledge provides the best performance. Both protocols are more efficient than standard aggregate protocols because they allow earlier query completion.

Categories and Subject Descriptors
C.2.4 [***Computer Communication Networks***]: Distributed Systems - distributed applications

General Terms
Management & Performance

Keywords
WSN, Domain Knowledge, Aggregate Queries, Optimization

1. INTRODUCTION
Wireless Sensor Networks (WSN) are deployed in our physical world to enable scientists to monitor the spatial and temporal properties of a landscape. Based on different application purposes, different approaches are proposed to manage the data of WSN [1-2]. Whatever the system is, it is almost unavoidable to discuss data aggregation which collects and computes the information from the whole or partial network. For example, if some application needs to know the average temperature of an area monitored by a WSN, all the nodes in that area should report the monitored local temperature to compute the average. Hence, the strategy of navigating the members in a WSN to get the aggregation with less cost becomes an optimal problem which is discussed from different aspects [1-2, 6, 11].

1.1 Motivation
In this paper, we propose a new approach for processing aggregate queries with the help of domain knowledge and sensing data constraints. We present this knowledge and constraints as a rule to guide the data aggregation protocol over WSN. To explain the real world requirement of data aggregation in WSN and our research motivation, we use a simple motivating example in Fig 1.

First of all, there are two queries. Query 1: sum of node1 and node2. Query 2 whether sum of node1 and node2 is larger than 80. With the information offered in Fig 1(a), for evaluating both Query 1 and 2, *node1* and *node2* send their readings back to the Sink node, and the *Sink* node merges those 2 readings. The result for Query 1 is 80 and the result for Query 2 is FALSE. So, under the scenario of Fig 1(a), the results of Query 1 and Query 2 are all generated in the *Sink* node. That is the popular operation adopted by most of the WSN query systems for data aggregation. However evaluation strategy in Fig 1(a) neglects the information carried by data and their constraints. In Fig 1(b), the query evaluation takes the constraints and domain knowledge into consideration. For Query 1, the readings of *node1* and *node2* have to be sent back to the Sink because the final value is required by the query. For Query 2, as soon as *node1* gets reading 30, the final logic result can be determined with the reference to the domain knowledge of *node2* where its maximum value has no chance to exceed 50. In other words, the sum of *node1* and *node2* has no chance to be larger than 80. As a result, *node1* itself can produce the final result and there is no need for sending any readings back to the *Sink*. So, compared with the method in Fig 1(a), the method in Fig 1(b) saves the cost of aggregating data transmission.

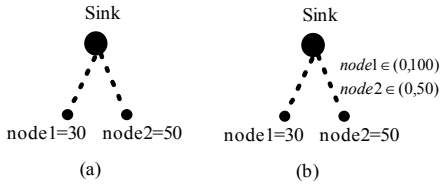


Fig 1. Motivating Example for Aggregate Query Processing

With the above example, we briefly conclude our motivation as using domain knowledge and constraints of data to control the data aggregation w.r.t. a class of application requirement at run-time. Hence, it helps the query system to avoid blind and unnecessary aggregation and package transmission during query evaluation.

1.2 Related Work
In terms of current data aggregation in WSN query systems, *TinyDB* [3] has proposed a way to provide support for decentralized aggregation which aggregates data during the data transmission based on distributed tree-based algorithm on sensor nodes. Its component-based aggregation framework allows it to be extended by wiring new customer aggregate components. *Regiment* [5] describes the global and group behavior of sensor networks to support decentralized data aggregation. In addition, some specific methods of data aggregation can be found in the following work [6-9]. Particularly, Random Walk techniques [7] are proposed to answer WSN queries by sequentially visiting a constant fraction (80%) of sensor network. Sparse Aggregation [8] proposes a

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9-10 September 2009.

way to recognize the query relevant nodes from a sparse set and avoid blind data aggregation.

[3] and [5] focus on data fusion, data sharing and base station side query rewriting. [7] and [8] discuss different methods to use the partial sensors to answer queries instead of a whole WSN. But according to our knowledge, no current work considers queries, domain knowledge and partial reports together to optimize protocols. That is the main contribution of this paper.

1.3 Organization of the Paper

The rest of this paper is organized as follows. In Section 2, we give a classification of the general application requirement for WSN. In Section 3, based on the classification, the data aggregation rule is offered. The corresponding scheduling protocols for the rule are also offered. Through a case study, we demonstrate the energy efficiency of the new protocols. In Section 4, an empirical study is given to verify the new protocols with the rule. In Section 5, we conclude the paper and outline the future work.

2. QUERY CLASSIFICATION

Fig 2 presents a typical SQL-like query language for WSN. From the motivation example in Section 1, we see that not all WSN query evaluation (application requirement answering) can be optimized by domain knowledge. To figure out the queries which our proposed aggregation approach has an effect on, we give a classification of queries over WSN through the following example. The different requirements are Query 1, return the sum of *node1* and *node2*; Query 2, If sum of node1 and node2 is larger than 80, return the readings of node1 and node2. Query 1 requires the final result of data aggregation. The other requires a logical judgement first before any other operation. To express the above two WSN application requirements formally, we can use different query languages such as SQL-like WSN Query language [3] and Situation Representation proposed by [4]. If using SQL-like WSN Query language [3] as shown in the Fig 2, the main difference between these two queries of the example can be found in "Having" clause where a predicate needs to be evaluated for the relationship between SUM(node1, node2) and 80.

Query Template with SQL-like WSN Query Language

SELECT	**{attribute, aggregate}**
FROM	**{Sensordata, S}**
WHERE	**{predicate}**
GROUP BY	**{attributes}**
HAVING	**{predicate}**
DURATION	**Intervals**
Interval	**time span *e***
predicate:=	**attribute | aggregate | aggregate Oper Constant | aggregate Oper aggregate | predicate ∪ predicate | predicate ∩ predicate | ¬ predicate**
Oper:=	**'>' | '<' | '≥' | '≤' | '≠' | '≡'**

Query 1)		*Query 1)*	
SELECT	**SUM(*node1, node2*)**	**SELECT**	***node1, node2***
FROM	**all sensors**	**FROM**	**all sensors**
WHERE	**wholeRegion**	**WHERE**	**whole region**
Interval	**1s**	**Having**	**SUM(*node1, node2*)>80**
		Interval	**1s**

Fig 2. Examples with SQL-like WSN Query Language

In Query 2, "HAVING" clause has an aggregate function "SUM" which needs to be evaluated. Instead of returning all the sensor reports to get the exact result of "SUM" function, logic values of these clauses are expected; whereas in Query 1, the sensor reports and the value of "SUM" are directly required by queries. Hence under the scenario of Query 2, if the logic judgement can be got as soon as possible, the accurate value of "SUM" may not be important. For example, to evaluate Query 2 in Fig 2, if we know the real report of *node1* is 30 and the maximum value of *node2* is 50, without receiving the exact value of *node2*, the predicate clauses of Query 2 can be evaluated to be false. As a result, with the cancellation of *node2*'s report, the answer of Query 2 can still be afforded. A similar scenario can be observed with the Situation Representation [4]. Based on above discussion, we partition the whole WSN query set Q into to 2 subsets Q_{AP} and Q_{NAP} where *AP* stands for aggregate predicate and *NAP* stands for non-aggregate predicate. The above Q_{AP}, particularly Q_{AP} with "AVERAGE" and "SUM" functions is our research interest. DATA AGGREGATION PROTOCOLS WITH DOMAIN KNOWLEDGE

In this section, we provide a domain knowledge based approach for aggregate query evaluation. For a given requirement (query) $q \in Q_{AP}$, and an unknown report data set S, to evaluate each application requirement $q \in Q_{AP}$ against S, the approach uses the domain knowledge of the whole network and constraints brought by real reports of $s \in S$, to build a subset of observations $S' \subseteq S$. Because the evaluation costs are $cost(S') < cost(S)$ in terms of the whole requirement set Q_{AP}, our proposed approach brings a significant cost saving. This approach will be introduced in following 3 subsections. *A)* Domain knowledge rule for aggregate query processing; *B)* Protocols with the proposed rule; *C)* Case study.

2.1 Rule for Data Aggregation

Suppose there is a sensor network $S = \{s_1, s_2, s_3, \ldots, s_n\}$ where there are n independent sensor nodes. For each node s_i, it collects m different readings which are called attributes. So, the readings of a node s_i can be described by a vector $s_i = \{att_1, att_2, att_3, \ldots, att_m\}$. Then we can get a matrix for all the sensor readings within a WSN, S.

$$S = \begin{bmatrix} s_1.att_1, s_1.att_2 & \cdots & s_1.att_m \\ \vdots & \ddots & \vdots \\ s_n.att_1, s_n.att_2 & \cdots & s_n.att_m \end{bmatrix}$$

Every column of S stands for the total reports of the whole WSN in terms of a certain attribute, and every row is the readings of one node. All of the aggregate operations require the processing over 1 or several columns in the matrix S. In addition, under most of situations, there exists the domain knowledge for each sensor reading, such as bounds, denoted as $S_i.att_j \in [LB_{ij}, UB_{ij}]$. LB_{ij} is the minimum possible reading of sensor s_i at its att_j, and UB_{ij} is the maximum possible reading of sensor s_i at its att_j. Hence, without any real readings, we can give the upper bound S_{UB} and the lower bound S_{LB} of the whole network readings in terms of their j^{th} attribute.

$$S_{LB} = \sum_{i=1}^{n} (LB_{ij}); \quad S_{UB} = \sum_{i=1}^{n} (UB_{ij})$$

However, with the continuous real reports from the sensors, we need to tight the lower bound S_{LB} and the upper bound S_{UB}. Suppose that k sensor have reported their readings at a certain time. The situations of the other n-k sensors are not clear. We can partition the sensor network into 2 sets based on k reported values and n-k yet to be reported. Then, the lower and upper bounds S_B are as follows.

$$S_B \rightarrow \begin{cases} S_{LB} = \sum_{i=1}^{k} (s_i.att_j) + \sum_{i=k+1}^{n} (LB_{ij}) \\ S_{UB} = \sum_{i=1}^{k} (s_i.att_j) + \sum_{i=k+1}^{n} (UB_{ij}) \end{cases}$$

As each new report is received (formula (1) to (2)), S_{LB} increases and S_{UB} decreases over time. Therefore, the uncertainty which means the difference between S_{LB} and S_{UB} decreases. Finally, the query result becomes more certain. Our goal is to answer queries as soon as possible with the more certainty.

$$S_B \rightarrow \begin{cases} S_{LB} = \sum_{i=1}^{m} (s_i.att_j) + \sum_{i=m+1}^{n} (LB_{ij}) \\ S_{UB} = \sum_{i=1}^{m} (s_i.att_j) + \sum_{i=m+1}^{n} (UB_{ij}) \end{cases} \quad (1)$$

$$S_B' \rightarrow \begin{cases} S_{LB} = \sum_{i=1}^{m+1} (s_i.att_j) + \sum_{i=m+2}^{n} (LB_{ij}) \\ S_{UB} = \sum_{i=1}^{m+1} (s_i.att_j) + \sum_{i=m+2}^{n} (UB_{ij}) \end{cases} \quad (2)$$

2.1.1 aggregate Oper Constant

Now we can rewrite the predicate of the query "*aggregate Oper Constant*" as follows with the introduction of S_B. $\forall$ *S_B Oper Constant,* $\exists$ *S_B - Constant Oper* 0. Because S_B is a domain, the *S_B – Constant* is also a domain. As shown in Figure 3, S_{LB} and S_{UB} are the lower bound and the upper bound of *S_B - Constant*. Totally, there are 3 types of relationship between *S_B – Constant* and 0. For scenario (a), we know that the aggregate result is definitely larger than 0. For scenario (b), the relationship between final result and 0 is still uncertain. For scenario (c), the aggregate result is less than 0. According to the above analysis, whatever the "*Oper*" is, if 0 $\notin S_B$, the predicate is determined. If 0 $\in$ S_B, the predicate is uncertain.

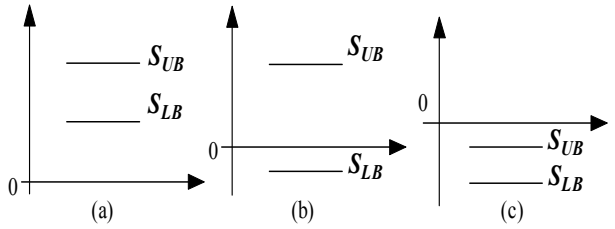


Fig 3. Relationship between S_B – Constant and 0

2.1.2 aggregate Oper aggregate

$\forall$ *S_{B1} Oper S_{B2},* $\exists$ *S_{B1} - S_{B2} Oper* 0. We can get a new S_B by merging the 2 bounds S_{B1} and S_{B2} together. The merged S_B is computed as follows.

$$S_B \rightarrow \begin{cases} S_{LB} = S_{LB1} - S_{UB2} \\ S_{UB} = S_{UB1} - S_{LB2} \end{cases}$$

Hence, we normalize the scenario *2)* to the scenario *1)* which compares the relationship between a S_B and 0. With the above normalization, we conclude our rule.

Aggregate Rule: In a WSN query system, for a query with its predicate carrying aggregate functions, the aggregation processing can be guided using the formula (***S_B Oper 0***) to determine whether an on-going predicate side data aggregation can be terminated.

(S_B Oper 0) Rule:

if (0 $\in$ S_B)
  continue data aggregation;
if (0 $\notin$ S_B & S_B Oper 0 == 'FALSE')
  terminate and reset interval;
if (0 $\notin$ S_B & S_B Oper 0 == 'TRUE')
  terminate aggregate, continue interval;

The ***S_B Oper 0*** rule offers the guide for data aggregation.

2.2 Protocols with (S_B Oper 0) Rule

Based on the difference of nodes themselves in a cluster-head WSN, we divide the protocols to "Sink" side for query propagation and "Slave" side for data collection. Totally, three protocols are offered, one protocol for "Sink" side initialization and two protocols with the rule for "Slave" side. Specifically, "Slave" side protocols include partial aggregate sequential report (PAS) and partial aggregate opportunistic report (PAO). The inputs of the protocols are a query $q \in Q$, and a whole reports set S. The output is a subset, $S'=Sub(S)$.

Sink Side Initialization Protocol

1. Initialize Q_{NAP}, Q_{AP};
2. if($\forall q, \exists Aggregate \in q.predicate$)
3. $q \rightarrow Q_{AP}$;
4. else
5. $q \rightarrow Q_{NAP}$;
6. $SLB0=\sum_{i=1}^{n} LBij$, $SUB0=\sum_{i=1}^{n} UBij$; //line6: compute the initial SB0 without report
7. Propogate(q, S_{B0}, $Start$) //propogate query, bounds and starting time to "slaves"

In addition to previous tasks completed by the query platform, such as query parsing and propagation, Sink Side Initialization Protocol also completes the task of query classification (Line 1 to 5), initial domain bound S_{B0} computation (Line 6). Finally, the initial S_{B0} and the results of query parsing will be propagated to all "Slave" nodes for distributed query evaluation.

Protocol PAS is an optimized algorithm based on Round Robin scheduling which requires all the "Slave" nodes to forward their readings back to the "Sink". From line 1 to line 2, it checks whether the time is the starting point of a query time interval, if yes, it will initialize the variables for future processing within this interval. In line 3, it checks whether it is the reporting time slot for the current sensor which runs the protocol. If it is the reporting time slot of the current node, it will check the query classification in line 4. If the query belongs to Q_{NAP}, it will call the normal data aggregate strategy in line 5. But if the query $q \in Q_{PA}$, in line 6 to 12 a normalization is carried out to normalize the bounds of different aggregate predicates to the uniform *S_B Oper 0* one. After that normalization, lines 13 to 16 are used to calculate the results such as partial aggregate and new domain bounds S_B based on new sensor reports. Lines 17 to 22 are the expression of our proposed rule which computes the predicate with the data knowledge bound and partial aggregate to control the data package transmission. More specifically, based on the rule, if 0 $\in$ S_B, PAS keeps working to the next time slot of the next node because the predicate of the query still can not be determined. If 0 $\notin S_B$, PAS will complete the predicate evaluation and data transmission as soon as possible, because the final result of the predicate can be got earlier. From line 23 to 25, if

it is not the reporting time slot for the current sensor, it activates the *overhear*(), to overhear the reports of other sensors in the cluster-head network.

Algorithm 2: PAS with (S_B *Oper 0*)

```
1.  if(slot(T)==start)
2.    S_i.att_j.Agg=0; S_i.Count=0;
3.  if(slot(T)==i)
4.    if(q∈Q_NAP) // q is a query without aggregate predicate
5.      Normal decentralized data aggregation;
6.    else  // q is a query with aggregate predicate
7.     if(∀q.predicate, ∃ 'Average()') // normalize 'Average' to 'Sum' aggregate
8.       q.Threshold*=n
9.     if(q.predicate ∈ Aggregate Oper Threshold) // line 8-13, normalize S_B0
10.      S_LB0=S_LB0-Threshold; S_UB0=S_UB0-Threshold;
11.    if(q.predicate ∈ Aggregate Oper Aggregate)
12.      S_LB0=S_LB1-S_UB2; S_UB0=S_UB1-S_LB2;
13.    S_i.att_j.Agg = All Child(S_i.att_j.Agg); S_i.Count = All Child(S_i.Count);
14.    S_i.att_j.Agg += S_i.att_j; S_i.Count++;
15.    S_i.att_j.B = ALL Child(S_i.att_j.B) + B_ij; // get the partial S_B of i reporting sensors S_i
16.    S_B = S_B0 - S_i.att_j.B + S_i.att_j.Agg; // compute the current S_B of sensor S_i
17.    if(0∈S_B) // line 22-27: different data routing strategies according to the rule
18.      DataRouting(S_i.att_j.B, S_i.att_j.Agg, Count, S_i);
19.    if(0∉S_B And S_B Oper 0 == 'FALSE')
20.      Propagate('Cancel'); Reset(Time Interval T)
21.    if(0∉S_B And S_B Oper 0 == 'TRUE')
22.      DataRouting(S_i);
23.  else
24.   if(overhear(S_i))
25.     S_i.att_j.Agg=+S_k.att_j.Agg; S_i.Count++;
```

Protocol PAO is optimized from Protocol PAS by allowing distinguished data for a predicate to report first. The distinguished data means the sensor reports which have more effect to accelerate the astringency of S_B in terms of a query $q \in Q_{AP}$. However, there are 2 important differences between Protocol PAS and PAO. The first one is in line 3, protocol PAO not only checks the reporting slot for itself, but also check whether the time t is less than 1/k *SLOT, which is a sub-slot left out specially for eagerly reporting cancellation message by any other sensor nodes. The second difference is the extra lines 26 to 30 in PAO protocol, which means if any sensor with distinguished reports can make a decision for a predicate, it will use the first sub-slot<1/k *SLOT belonging to every time slot to report cancellation messages. Hence, this flexible report eagerly terminates unnecessary package transmissions.

Protocol 3: PAO with (S_B *Oper 0*) based on PAS

```
.....
3.  if(slot(T)==i&&t>SLOT/k) //time slot is for data report but not for 'cancel'
.....
26.   if(t<SLOT/k) // for 'cancel' message
27.     if(0∉S_B And S_B Oper 0 == 'FALSE')
28.       Propagate('Cancel'); Reset(Time Interval T);
29.     if(0∉S_B And S_B Oper 0 == 'TRUE')
30.       DataRouting(S_i);
```

2.3 Case Study

To study the Energy costs for a WSN query processing protocol, we can use energy consumption model as follows [10]. Q_s is the size of the query to be propagated. E_b is the in-network energy cost to send 1 bit per hop. E_r is the in-network energy cost to receive 1 bit per hop. T_s is the size of the data package to be transmitted. E_{cn} is the energy cost for a computation of an in-network partial aggregation. C_s is the size of the aggregation cancellation message. D is the duration for a specific query. T is the time interval for each query answer return. With these definitions, we use a data aggregate example in Fig 4 with 1 sink node and 10 slave sensors to demonstrate how the protocols work. In Fig 4, the sink and all nodes form a 1-hop net work. All the slave sensors collect attributes including temperature, humidity and light. We assume the communication between nodes is lossless. Available domain knowledge is temperature ranging from 0 to 100. A user requirement is as follows and threshold is T*=80. We compare the energy costs among the RR, PAS and PAO protocols.

```
Select   (Light, Humidity);
Where    AreaID=1;
Having   Average(Temperature)>T*;
Interval 1s;
```

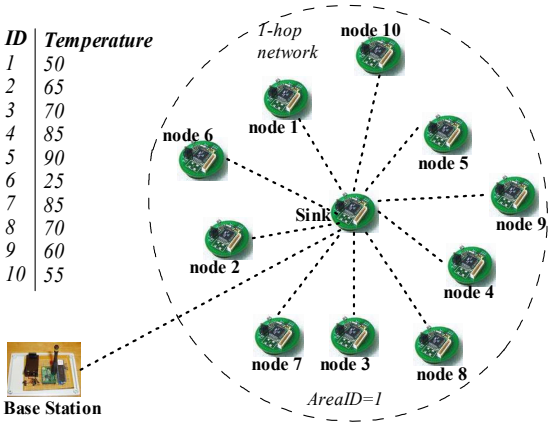


Fig 4. Aggregate example in 1-hop network (10 slave nodes) with 1 Sink

2.3.1 Energy Cost of Round Robin Scheduling

The RR protocol schedules the sensor reports based on a fixed order. In Fig 4, RR scheduling is based on the sensor's ID from 1 to 10. Specifically, when receiving the query from the sink node, all 10 slave sensors in the area 1 report their readings of the humidity, light and temperature attributes. The sink node evaluates the query predicate and sends the final result back to the base-station. The energy cost for the propagation is $E_{qb}+E_{qr}$. $E_{qb}=Q_s \times E_b$ is the energy cost for sending the query at sink node. $E_{qr}=Q_s \times E_r \times 10$ is the total energy cost for receiving the query by all slave nodes because all 10 nodes receive the query propagation package from sink node. The energy cost during the data package returning stage is $E_{ds}+ E_{dr} = 10 \times T_s \times (E_b+E_r)$, where E_{ds} is the energy cost from all 10 sensors for sending a data package back to the sink node and E_{dr} is the energy cost at the sink node for receiving all 10 packages from its slave nodes. So in an interval $T=1s$, within area 1, the energy cost for the query evaluation of Round Robin scheduling is $E_{RR} = E_{qb}+E_{qr}+E_{ds}+E_{dr}$.

2.3.2 Energy Cost of PAS Algorithm

At the stage of query propagation, the energy cost of the PAS is similar to that of Round Robin scheduling. That is $E_{qb}+E_{qr}= Q_s \times (E_b+ E_r \times 10)$. At the stage of data package returning, every slave node in the WSN recalculates the

bounds to judge the query predicate based on the rule continuously. PAS improves Round Robin by terminating the sequential data reporting when a clear decision can be made for a predicate. In this example, when the sensor 1 to 5 forwards its data package to the sink node in its reporting time slot, all the other nodes can overhear it and do a partial data aggregation to judge the predicate independently based on the rule. At the same time, the threshold 80 keeps in the domain of upper and lower bounds. However, in the time slot for sensor 6 to report, it uses temperature from sensor 1 to 6 to calculate the relationship between Threshold and $[\sum_{i=1}^{k} reading(i) + (n-k)Lower,\ \sum_{i=1}^{k} reading(i) + (n-k)Upper]$. Sensor 6 gets Threshold 80>[385, 785]/10. In other words, the predicate has no opportunity to be larger than 80, which has a conflict to the query predicate. Then, a cancellation message is sent. The rest of the slave nodes with ID from 7 to 10 overhear the cancellation message, terminate their packages to be sent and reset for next interval. The total energy cost of PAS in a time interval is $E_{PAS}= E_{qb}+E_{qr}+E_{ds}+E_{dr}+E_c+E_{cn}$.

2.3.3 Energy Cost of PAO Algorithm

The PAO protocol schedules nodes with important readings to report first, which accelerates the evaluation of the predicate. The energy cost for query propagation is still $E_{qb}+E_{qr}= Q_s\times(E_b+ E_r\times10)$. At the stage of the data collection, according to the temperature list in Fig 4, when Sensor 4 reports to the sink node, it will be overheard by all the other nodes including node 6. With this partial aggregation, node 6 gets Threshold 80>[295, 795]/10. With that result, node 6 will send a cancellation message in the time slot which originally is scheduled for node 5 to report its data packages. So, this change of node 6's reporting order saves the package transmission of node 5 and the query evaluation can be terminated with 5 packages transmission totally. The energy cost of PAO is $E_{PAO}=E_{qb}+E_{qr}+E_{ds}+E_{dr}+E_c+E_{cn}$. Compared to PAS, the energy saving of PAO is mainly achieved by smaller E_{ds} and E_{dr} which are analysed in Table 1.

Table 1. Energy cost of different scheduling algorithms

Protocols	Energy Cost (Package size unset)	Final Cost
RR	$(Q_s+10\times T_s)\times E_b+10\times(T_s+ Q_s)\times E_r$	$560\times E_b+1280\times E_r$
PAS	$(C_s+5\times T_s+Q_s)\times E_b+(C_s+10\times Q_s+5\times T_s)\times E_r$	$329\times E_b+1049\times E_r$
PAO	$(C_s+4\times T_s+Q_s)\times E_b+(C_s+10\times Q_s+4\times T_s)\times E_r$	$281\times E_b+1001\times E_r$

Table 1 compares different energy costs of our three protocols. In different applications, Q_s, T_s and C_s change. Normally, C_s is less than T_s and Q_s. Suppose C_s is 1 byte, Ts is 6 bytes and Q_s is 10 bytes in our case study. We get final costs for the 3 protocols, $E_{RR}=560\times E_b+1280\times E_r$, $E_{PAS}= 329\times E_b+1049\times E_r$ $E_{PAO}= 281\times E_b+1001\times E_r$. It can be observed that energy cost $E_{RR}>>E_{PAS}>E_{PAO}$.

3. Empirical Study

To study the performance gains of new protocols, we carry out an empirical study for Round Robin (RR), Partial Aggregate Sequential Report (PAS) and Partial Aggregate Opportunistic Report (PAO) protocols.

3.1 Assumptions

In order to compare the difference of package transmission brought by different approaches, 2 important aspects should be considered; (1) the features of the data, (2) the query and observations. In terms of (1), the evaluation is carried out based on two classes of data distributions. The first class consists of randomly distributed data ranging from 0 to 100. The second class consists of a normally distributed data which is commonly less than 30 with the probability of 80 %, but with small possibility to have readings larger 90 with the probability of 10%. In terms of (2), the query requires to count the packages transmitted when a certain predicate can be evaluated, and the observation is the total size of transmission in bytes. In this evaluation, a small data package is 16 bytes and cancellation information is 1 byte. Furthermore, Fig 5 and 6 are based on the average value of 5 repetitive experiment results.

3.2 Evaluation Result Analysis

Fig 5 (a) and (b) show RR, PAS and PAO to make the decision of a predicate over uniform random distributed data set. Every sink node has 5 or 10 slave nodes and their working interval is 1s. In Fig 5(a), the curve of RR indicates that RR imposes equal transmission cost 80 bytes, whatever the threshold is. The curves of PAS & PAO indicate that with the increase of the threshold of the predicate within the whole range of the aggregate domain, the transmission increase to the peak, 73 bytes, then goes back to 0. This scenario means when the threshold is closer to the upper bound or the lower bound of the aggregate domain, PAS & PAO have more opportunity to make a decision of the predicate and terminate potential transmission. As shown in Fig 5 (b), with the increase of the sensor number to 10, PAS and PAO achieve approximate half the total transmission cost of RR in the whole query set $q\in Q_{AP}$. But it can be found that PAS and PAO don't optimize the worst case transmission cost. In addition, in Fig 5 (a) and (b), it can be found that even their curves are quite similar, PAO outperforms PAS a little under. Theoretically, PAO should achieve better than PAS because it schedules more qualified nodes to report first. Under the random distributed reports, to evaluate the relationship between $[\sum_{i=1}^{k} reading(i) + (n-k)Lower, \sum_{i=1}^{k} reading(i) + (n-k)Upper]$ and a threshold for rule, there are 2 important factors. One factor is $\sum_{i=1}^{k-1} reading(k-1)$, the aggregate of reported readings. The other is the current reporting node k. Because the data reports are randomly distributed, $\sum_{i=1}^{k-1} reading(k-1)$ has much larger influence to the expression than a single report k. Although PAS does not report the most qualified k as soon as possible, but with the following k+1 or k+2 nodes, it also has great opportunity to make the decision. So, only small difference around 10 bytes between PAS and PAO is observed Fig 5 (a) and (b).

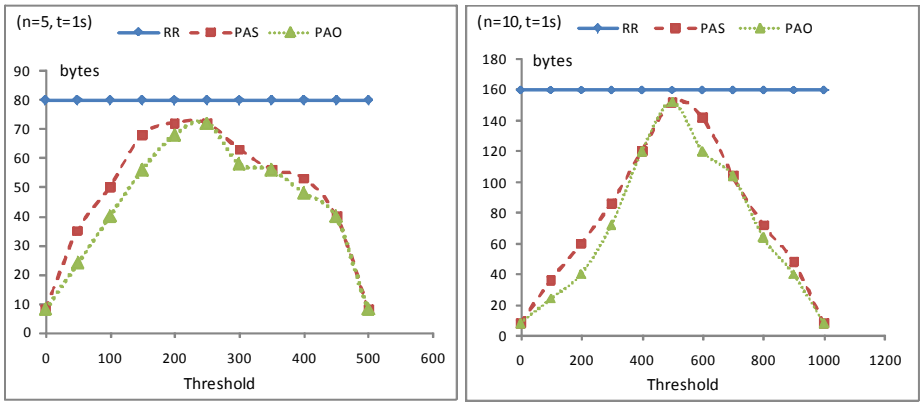


Fig 5. Evaluating Aggregate Predicate on Uniform Random Distribution

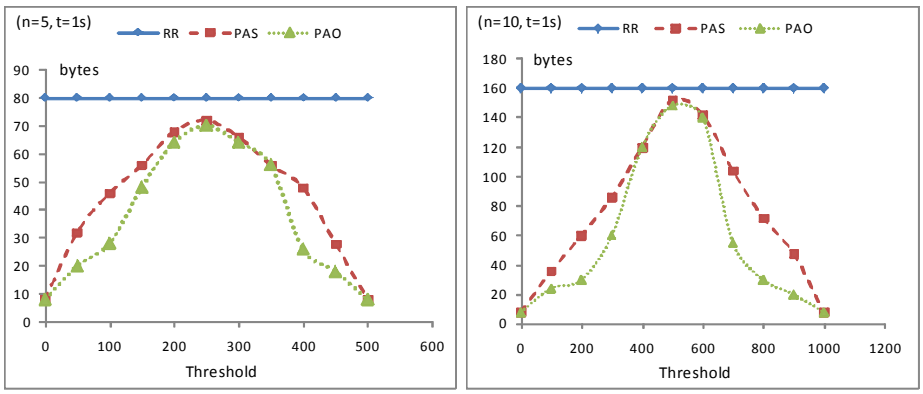


Fig 6. Evaluating Aggregate Predicate on Normal Distribution Reports

However, in real world applications, reports are more likely to follow a normal distribution than uniform random distribution. Fig 6 shows all the three approaches for data aggregate over normal data distribution. Similar to the results in Fig 5, PAO and PAS approaches outperform RR approach in terms of the whole query set with its threshold changing within the domain. PAS and PAO approaches also achieve the half total transmission cost of RR on average. And the worst case total transmissions of the 3 approaches are still similar, around 80 bytes in Fig 6 (a) and 160 bytes in Fig 6 (b). Under the normal distribution, PAO has a significant improvement of performance compared to PAS. When the query threshold is much closer to the bounds of the domain, PAO saves as many as 40 bytes compared to PAS at some threshold values. The reason for this performance gain is that normal distribution has 80% probability to get relative steady reports less than 30. But it also can have small opportunity to get reports larger than 90. So during some query evaluations, for example, one report of 90 has the effects of three reports of 30, or vice versa. In other words, one report can replace the other three in terms of decision making. PAO allows these distinguished report to report first to finish queries. This effect can be observed more clearly when the threshold is close to the domain bounds where one very large or very small report which is scheduled to report first can determine the predicate judgement. So, under normal data distribution, the rule based aggregate protocols also have a better performance gains than RR approach without the rule. Furthermore, PAO earns more significant gains than PAS for its flexible scheduling report of performance compared to PAS. When the query threshold is much closer to the bounds of the domain, PAO saves as many as 40 bytes compared to PAS at some threshold values. The reason for this performance gain is that normal distribution has 80% probability to get relative steady reports less than 30. But it also can have small opportunity to get reports larger than 90. So, in some query evaluations, for example, one report of 90 has the effect of three reports of 30, or vice versa. In other words, one report can replace the other three in terms of decision making. PAO allows these distinguished reports to be routed first to finish queries. This effect can be observed more clearly when the threshold is close to the domain bounds where one very large or very small report which is scheduled to report first can determine the predicate judgement. Under normal data distribution, the rule based aggregate protocols have a better performance gains than RR approach without the rule. Furthermore, PAO earns more significant gains than PAS for its flexible scheduling reports.

4. Conclusions and Future Work

In this paper we proposed a novel rule based approach to process data aggregate queries in a WSN. First, we gave a classification of queries for data aggregation. Based on that classification, we introduced a domain knowledge based rule for query evaluation. Two protocols, PAS and PAO are offered. We evaluate the performance of each protocol under different assumptions about the distribution of data observations. Opportunistic use of domain knowledge in PAO provides

the best performance. PAS is more efficient than RR because its partial aggregation allows earlier query termination. In the future, we plan to develop aggregation approaches for approximate query answers by query relaxations, and for more general sensor network topologies.

5. Acknowledgement

Chi Yang is supported by a scholarship from University of Western Australia and the Motorola Global Software Group, Perth, Australia.

6. REFERENCES

[1] J. Gehrke and S. Madden, "Querying Processing in Sensor Networks," IEEE Pervasive Computing, March. 2004.

[2] R. Sugihara and R. K. Gupta, "Programming Models for Sensor Networks: A Survey," ACM Transactions on Sensor Networks, vol. 4, no. 2, March. 2008.

[3] S. Madden, M. Franklin, J. Hellerstein and W. Hong, "TinyDB: An Acquisitional Query Processing System for Sensor Networks," ACM Transactions on Database Systems (TODS), vol. 30, no. 1, pp. 122-173, 2005.

[4] R. Cardell-Oliver and W. Liu, "Representation and Recognition of Situations in Sensor Networks," to appear in IEEE Communications Magazine, Special Issue on Situation Management, August 2009.

[5] R. Newton, G. Morrisett and M. Welsh, "The Regiment Macroprogramming System," Proc. of ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'07), pp. 489-498, April. 2007.

[6] S. Xiang, H. B. Lim, K. Tan and Y. Zhou, "Two-Tier Multiple Query Optimization for Sensor Networks," Proc. of 27th International Conference on Distributed Computing Systems (ICDCS'07), 2007.

[7] A. Chen and B. Carlos, "Efficient and Robust Query Processing in Dynamic Environments Using Random Walk Techniques," Proc. of ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'04), pp. 277-286, April. 2004.

[8] J. Gao, L. Guibas and J. Hershberger, "Sparse Data Aggregation in Sensor Networks," Proc. of the 6th International Conference on Information Processing in Sensor Neworks (ACM IPSN'07), pp. 430-439, 2007.

[9] J. Benson, T. O'Donovan, U. Raoedig and C. J. Screenan, "Opportunistic Aggregate over Duty Cycled Communications in Wireless Sensor Networks," Proc. of the 7th International Conference on Information Processing in Sensor Neworks (ACM IPSN'08), pp. 307-318, 2008.

[10] A. Coman, J. Sander and M. A. Nascimento, "An Analysis of Spatio-Temporal Query Processing in Sensor Networks," Proc. of 21st International Conference on Data Engineering Workshops, pp. 1190-1196, 2005.

[11] P. Edara, A. Limaye and K. Ramamritham, "Asynchronous In-network Prediction: Efficient Aggregation in Sensor Networks," ACM Transacations on Sensor Network, vol. 4, no. 4, article 25, 2005.

Abstracts

Hypervolume: Calculation and Application in Multi-objective Optimisation

Lucas Bradstreet, Luigi Barone and Lyndon While
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
{lucas, luigi, lyndon}@csse.uwa.edu.au

ABSTRACT

A multi-objective optimisation problem (MOO) problem is one where potential solutions are assessed by their performance in more than one criterion, or objective [7][8]. The result of running an MOO algorithm is a set of solutions (called a front), with each solution representing a trade-off between objectives. Multi-objective Evolutionary Algorithms (MOEAs) are evolutionary algorithms applied to MOO problems. An example of an application where MOEAs excel can be found in Barone et al.[1] with their multi-objective EA for the design of rock crushers. Rock crushers are used to extract raw materials from rocks and are subject to various tunable parameters. Barone et al.'s MOEA attempts to simultaneously optimise two objectives: maximisation of the capacity of the rock crushers while minimising the size of the crushed product. MOEAs produce a range of solutions, each representing a different trade-off, that could not be captured ahead of time. As such MOEAs allow their users to respond effectively to change. As it may take a significant time to rerun an EA to respond to changing conditions, MOEAs represent a significant advantage for a problem solver who wishes to quickly adapt.

Unfortunately, the front of solutions produced by MOEAs can be infeasibly large when dealing with real valued objectives. As a result of space/computational limitations an MOEA is unable to retain all generated trade-off solutions and instead endeavours to keep the solutions that best cover the trade-off front. Several metrics have been used to determine which solutions should be discarded in lieu of another[9, 11]. One of the most popular is Hypervolume, otherwise known as the Lebesgue measure or S-metric[10]. Hypervolume is the n-dimensional space that is "contained" by the points (solutions) in a front. A front with a larger Hypervolume is likely to present a better set of trade-offs to a user than a front with a smaller Hypervolume.

My work involved an indepth look at the S-metric or Hypervolume algorithm. Existing Hypervolume algorithms have exponential time complexity, however they are still extremely popular in Multi-objective optimisation research as the measure has many favourable properties.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

My primary contributions are three-fold:

- Improved on performance of Hypervolume algorithms for performance metric use [12, 6].
- Created an incremental contribution Hypervolume algorithm, IHSO, and techniques to improve its performance when used for selection [5].
- Discovered and investigated techniques for use in selection. These included greedy front selection algorithms and a local search algorithm, each of which attempt to maximising the hypervolume of reduced size fronts [2, 3, 4].

I will describe these techniques and discuss how they evolved throughout my PhD.

1. REFERENCES

[1] L. Barone, L. While, and P. Hingston. Designing crushers with a multi-objective evolutionary algorithm. In W. B. Langdon *et al.*, editor, *GECCO-2002*, pages 995–1002. Morgan Kaufmann Publishers, 2002.

[2] L. Bradstreet, L. Barone, and L. While. Maximising hypervolume for selection in multi-objective evolutionary algorithms. In C. L. P. Chen, editor, *Congress on Evolutionary Computation*, pages 1744–1751. IEEE, 2006.

[3] L. Bradstreet, L. Barone, and L. While. Incrementally maximising hypervolume for selection in multi-objective evolutionary algorithms. In Arthur Tay, editor, *Congress on Evolutionary Computation*, pages 3203–3210. IEEE, 2007.

[4] Lucas Bradstreet, Luigi Barone, and Lyndon While. Updating Exclusive Hypervolume Contributions Cheaply. In *2009 IEEE Congress on Evolutionary Computation (CEC'2009)*, pages 538–544, Thronheim, Norway, May 2009. IEEE Press.

[5] Lucas Bradstreet, Lyndon While, and Luigi Barone. A Fast Incremental Hypervolume Algorithm. *IEEE Transactions on Evolutionary Computation*, 12(6):714–723, December 2008.

[6] Lucas Bradstreet, Lyndon While, and Luigi Barone. A Faster Many-objective Hypervolume Algorithm using Iterated Incremental Calculations. *to be submitted*.

[7] C.A. Coello Coello. *Evolutionary Algorithms for Solving Multi-objective Problems*. Kluwer, 2002.

[8] K. Deb. *Multi-objective Optimisation using Evolutionary Algorithms*. Wiley, 2001.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[10] M. Fleischer. The measure of Pareto optima: Applications to multi-objective metaheuristics. Technical Report ISR TR 2002-32, Institute for Systems Research, University of Maryland, 2002.

[11] S. Huband, P. Hingston, L. While, and L. Barone. An evolution strategy with probabilistic mutation for multi-objective optimization. In Hussein Abbass and Brijesh Verma, editors, *Congress on Evolutionary Computation*, pages 2284–2291. IEEE, 2003.

[12] L. While, L. Bradstreet, L. Barone, and P. Hingston. Heuristics for optimising the calculation of hypervolume for multi-objective optimisation problems. In B. McKay, editor, *Congress on Evolutionary Computation*, pages 2225–2232. IEEE, 2005.

Quantifying the Effect of Responses used in the Influenza H1N1 2009 Swine Flu Outbreak in Australia using an Individual-Based Model

Nilimesh Halder, Joel K Kelso and George J Milne

School of Computer Science and Software Engineering,

The University of Western Australia

Email: {nilimesh, joel, milne}@csse.uwa.edu.au

ABSTRACT

Background

Influenza H1N1 2009 swine flu has spread across the world causing a pandemic. This novel strain of virus results in mild symptoms; however public health measures have been activated to lessen its potential impact. According to pandemic preparedness plans of different countries, school closure and antiviral based strategies are implemented as the first available containment resources during the initial phase of pandemic due to lack of suitable vaccines and to lessen the probable risk of influenza spread. However the long term effectiveness of these intervention policies is difficult to predict in the early phase. In the absence of other evidence, computer modelling and simulation can provide guidance to public health policy makers about the possible impact of intervention policies.

Methods

A detailed, individual-based model of a real community (Albany, a small city in Western Australia) with a population of approximately 30,000 has been used to predict the impact of the initial intervention strategies used to contain the pandemic. We simulated the different containment strategies that were implemented in Australia, namely school closure and antiviral drugs used for treatment and prophylaxis (i.e. prevention). Each of the intervention measures is simulated in isolation and in combination with each other to form different plausible intervention scenarios. Using this model we have examined the mitigating effect of pharmaceutical based antiviral strategies and further considered their mitigating impact in the presence of social distancing measures such as school closure. We are thus able to quantify the impact of the interventions used and in addition, are also able to suggest improved intervention strategies.

Results

We have simulated epidemics with basic reproductive number R_0 range from 1.4 to 1.6. We summarized the results of epidemics of 1.5 (basic reproductive number) which aligns with recently estimated R_0 values (between 1.4~1.6) for the swine flu outbreak in Mexico. In epidemiology, the basic reproduction number R_0 (sometimes called basic reproductive rate or basic reproductive ratio) of an infection is the mean number of secondary cases a typical single infected case will cause in a population with no immunity to the disease in the absence of interventions to control the infection. Simulations show that school closure strategies of only 1 week have a negligible effect on reducing the final illness attack rate; it may reduce the overall final attack rate from 32.5% to 30%. However antiviral based strategies are shown to have greater impact on reducing influenza spread compared to school closure. Antiviral drug *treatment* of 50% symptomatic cases can reduce the attack rate by 6.5%, from 32.5% to 26%. Antiviral *treatment* of diagnosed individuals (again with a 50% case diagnosis rate) and *prophylaxis* to their household members can reduce the final attack rate to 19%. The addition of extended *prophylaxis* therapy to close contact groups in schools and workplaces of a diagnosed symptomatic case can further keep the final attack rate to 13% and reduce the peak daily attack rate to 22 per 10000 individuals from 120 per 10000. These simulations allow us to determine the size of antiviral stockpile required; the ratio of the required number of antiviral courses to population is 13% for antiviral *treatment* strategies, 25% for *treatment* and household *prophylaxis* and 40% for treatment and household and extended *prophylaxis*. In all these scenarios a diagnosis rate of 50% of infected individuals is assumed. School closure of up to 4 weeks together with different antiviral strategies is also simulated. Our simulations suggest that 2 weeks of school closure coupled with antiviral *treatment* and extended *prophylaxis* can reduce the final attack rate to 11% and the peak daily attack rate to 17 per 10,000 individuals with a ratio of required antiviral courses of 35%. On the other hand 2 weeks school closure with antiviral *treatment* and a household based *prophylaxis* strategy can reduce the final illness attack rate to 15% and the peak daily attack rate to 23 per 10,000 but with a cost of 20% of antiviral courses. Further school closure of up to 4 weeks is shown to have little reduction on both final and peak daily attack rate.

Conclusions

Our results show the critical role of antiviral drug strategies in the potential control of an influenza pandemic with the characteristics of influenza H1N1 2009 swine flu and indicate that coupling antiviral interventions with school closure may significantly reduce the risk of disease progression in community. They can significantly contain the rate of influenza epidemic development and provide sufficient time for development and distribution of new vaccines.

Heuristic RNA Pseudoknot Detection in Long Sequences

Jana Sperschneider, Amitava Datta and Michael Wise
School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
email: {janaspe, datta}@csse.uwa.edu.au

1. BACKGROUND

There are two types of nucleic acids in the living cell: deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). RNA is a versatile macromolecule which is no longer seen as the passive intermediate between DNA and proteins. Numerous functional RNA molecules with an astonishing variety have been uncovered in the last decade. Macromolecule function is closely connected to its three-dimensional folding and therefore, structure prediction from the base sequence is of great importance. The most popular approach for RNA secondary structure prediction is free energy minimization using dynamic programming. The key concept which allows for dynamic programming is that all structure elements are non-crossing and self-contained in terms of their free energy. Using dynamic programming, the secondary structure with minimum free energy (MFE) can be predicted in $O(n^3)$ time and $O(n^2)$ space. However, the inability to predict crossing structure elements, so-called pseudoknots, is a major drawback.

Pseudoknots are functional structure elements which occur in most classes of RNA and in many viruses. Pseudoknots play key roles in viral genome replication and regulation of protein synthesis. Therefore, prediction of these crossing structures in RNA molecules has important implications in antiviral drug design. Pseudoknots are also found in the cell where they participate in processes such as splicing, ribosomal frameshifting, telomerase activity and ribosome function. In many cases they assist in the overall three-dimensional folding and should not be excluded from computational structure prediction.

From a theoretical point of view, RNA secondary structure prediction including arbitrary pseudoknots is an NP-complete problem. Restricted classes of pseudoknots can be included in the dynamic programming algorithm for prediction of the MFE structure. However, these algorithms are computationally very expensive and their accuracy deteriorates if the input sequence containing the pseudoknot is too long. Due to the computational complexity of dynamic programming for pseudoknot prediction, heuristic approaches were developed as an alternative. Heuristic methods do not necessarily return the MFE structure, however they can include a wide class of pseudoknots in reasonable runtime.

17th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 9–10 September 2009.

2. PROPOSED STUDY

A different algorithmic framework proposed in this study is heuristic pseudoknot detection. Here, one attempts to find promising pseudoknot candidates in a sequence as a preliminary step for structure prediction. These potential pseudoknots are subsequently analyzed and verified. After pseudoknot detection, the remaining sequence can be folded using the mathematical exact method of free energy minimization in $O(n^3)$ time and $O(n^2)$ space. Pseudoknot detection has two main advantages over dynamic programming for pseudoknot prediction. First, it is computationally much more efficient. Second, the underlying framework is less restrictive and allows for easy incorporation of sophisticated energy rules for pseudoknots and comparative information.

This study will furthermore focus on the pseudoknot energy parameters which are widely used in the literature. For pseudoknot prediction, algorithmic runtime has been improved in many cases whereas predictive accuracy is always limited by the underlying folding model. For prediction of non-crossing RNA secondary structures, additive energy models have been used successfully. However, for pseudoknots this principle fails as there is strong interference between opposite loops and stems. A sophisticated pseudoknot energy model has to take into account stem-loop correlations. This is crucial for successful prediction, yet there is no method in the literature which efficiently employs such a model. It is the main aim of this study to investigate and incorporate an advanced pseudoknot energy model, resulting in a rapid and reliable pseudoknot prediction method.

From a practical point of view, the goal of this project is to look for pseudoknots in a wide range of viral genomes and, where found, note their locations. As structure is strongly related to function, computational prediction can deliver the basis for laboratory experiments on detected pseudoknots. Most methods in the literature are tested only on short sequence stretches containing known pseudoknots. In contrast, the benefit of the pseudoknot detection approach will be demonstrated by applying it to long sequences and ultimately viral genomes.

Keywords: RNA structure prediction, Pseudoknots, Heuristic Algorithms, Minimum Weight Independent Set.
CR Classifications: J.3 [Computer Applications]: LIFE AND MEDICAL SCIENCES.