

FINAL REPORT

Project: Experts Can Finally Be Lazy: Lazy Approximation in Expert Systems

Student Researchers: Leha Blaney and Gina Castano

Advisor: Dr. Lynn Stauffer

Institution: Sonoma State University

The application of lazy evaluation to a fuzzy expert system proves to be one of considerable complexity, with some significant problem areas. In the course of our research, we isolated many of these problems. We can now offer ourselves and other researchers many directions for further research, but much is yet to be done before the real work of implementation, evaluation, and comparison with other models may begin in earnest. Our paper discusses these problem domains, the work we were able to do that was essential and preliminary to the implementation of a truly lazy system, and how we plan to continue our research from this point forward. This paper also outlines our experience with the research process, itself, and what we have learned from that experience.

Our research project began with the search for a suitable topic. We were interested in both expert systems and functional programming and wanted to see if there was a useful way to tie the two together. We discovered the Watt model of separation of control from calculation using lazy evaluation, and we found this to be an interesting concept to apply in the expert system arena. Gathering some preliminary papers and books on the two subjects, we were able to come up with several ideas for the application of the Watt model in the context of a fuzzy expert system. We made a thorough search of available information to determine that no one had already done the work we were planning. We then compiled our proposal based on this information, with the details remaining to be worked out in the course of the research. The proposal outlined several phases that were intrinsic to our plan of attack. Over the course of the year, we executed each of these phases to the best of our ability.

First, we set up an online conference for meetings, links, and discussions, and a WWW page to give the public access to our proposal and findings.

We then began a more thorough effort to gather papers and other documents on the subjects of lazy evaluation, expert systems, fuzzy logic, functional languages, logic languages, expert system shells, and various combinations of these.

Once we had gathered these materials we began the reading portion of study. The first topic to read about was functional programming. We looked at Scheme, Lazy ML, Haskell, and Miranda. Of these languages, Haskell seemed the most applicable to our goals. However, there was no fuzzy Haskell that we could find, and the task of creating a fuzzy Haskell expert system from scratch was beyond the scope of our time constraints. We do not consider this to be a bad concept, though, and hope that other researchers (or ourselves) will have the opportunity to investigate it. We are also interested at this point in creating a fuzzy expert system shell in Haskell. An expert system shell is a very high level language, written in a medium to high level language, specifically for the purposes of making the writing of expert systems as simple and streamlined as possible. A shell written in Haskell would make the use of lazy evaluation in the expert system natural and straightforward.

Our next topic of investigation was the mechanics of lazy evaluation, itself. We investigated the use of list comprehensions and lazy lists in the various functional languages we'd studied, and implemented various examples of the Watt model in these languages. We had regular brainstorming sessions wherein we expanded on our ideas for application of these tools in the expert system.

Then we read about fuzzy systems. None of us had had any prior experience working with fuzzy logic, so this portion of our research was both intensive and fascinating. We explored various areas in which fuzzy logic has successfully been applied, with the main focus, of course, on expert systems. Here, fuzzy logic is used to get an approximate answer when no exact answer is possible, and to act as a rule builder for expert systems that actually grow in expert knowledge over the course of use.

We learned the theory and organization of traditional expert systems. Much of our course work in databases, data structures, and software engineering had prepared us for more sophisticated models, such as the expert system, so we were ripe for this part of our study and the concepts were quick and easy to grasp. Expert systems are also generally written in logic, or relational programming languages, so part of our work here was to survey a few of these languages, learning more about how they work and looking for any that might already be capable of lazy evaluation, or of using fuzzy logic. We found many were capable of fuzziness, but only one had been altered to include lazy evaluation at all, and this only a partially lazy amendment to the existing language (Prolog). At this point, we determined that not only were we going to have to use cross platform tools to make our system lazy, but if we were going to be able to write a small expert system, to do so in one of these relational languages would be a formidable and time-consuming task. Wanting to move on to our chosen area of study as quickly as possible, we chose to look at expert system shells.

We found and evaluated several expert system shells, a few of which could build fuzzy expert systems. The most promising of these was FLOPS, a shell that comes in both fuzzy and non-fuzzy flavors. We attempted to procure a copy of FLOPS from its author, William Siler, but after several emails it was beginning to seem like it was just going to take too long to achieve good communication with Dr. Siler, so, pressed for time, we went ahead and began implementing our system in CLIPS, a shell that is freely available, well documented, and quick to download. CLIPS was developed by NASA, who at one point had considered implementing the shell in LISP, a partially functional language in which it is fairly straightforward to create the functions Delay and Force, which can be used to implement lazy evaluation. For various reasons, however, NASA ultimately chose to implement their shell in C, a procedural language. At the time we began creating our expert system, we thought it would be possible to either use cross platform tools to add laziness from functional code, or use the Delay and Force functions in the C environment. As best as we can tell at this time, neither of these is possible. Perhaps this is an area for further research, but articles we read on the semantics of functional programming and its differences from the procedural paradigm make it look less than promising.

Gina had the opportunity to attend the Symposium on Applied Computing 1999 in San Antonio, Texas, where she participated in the Artificial Intelligence and Fuzzy Logic tracks. She gained insight from the lectures on contemporary work in the field.

Leha also plans to more fully develop our small bird identification expert system. The system can currently identify a few immature birds of California. Leha hopes to add the ability to identify adult birds and fledglings, and to make the list of identifiable species more complete for a larger geographical base.

We all hope to continue our work by either finding or creating an expert system shell that is written in a functional (or partially functional) language. Ideally, that language would be Haskell or Miranda. Other areas to for possible research are, as mentioned above, the implementation of laziness in C, or some other procedural language; use of lazy lists in a purely functional environment for prototyping of expert system front ends; creation of a relational language that has delayed evaluation, fuzzy logic and fuzzy sets built in; and creation of a more fully lazy version of Prolog, Datalog, Likelog, or some other existing relational language.

The opportunity to do research as undergraduates has helped us in many ways. Through gathering and reading the contemporary information in our area of study, we gained greater skill in the important area of surveying and reviewing the work of peers and of computer science professionals. By implementing lazy lists in the Watt model, choosing an expert system shell, evaluating several languages, expanding our knowledge of artificial intelligence, encountering and coping with obstacles in the course of seeking solutions, and working collaboratively, we have gained a better understanding of the research process and of the difficulties and challenges inherent in the pursuit of advanced research topics. Finally, through choosing a research topic, writing a successful proposal, and collecting and processing the results of our study, attempting to produce an honest document that will be useful to others involved in research, we have gained an important skill set that will help us in our graduate level pursuits, and in our careers.

Mathematical Puzzles with Smart Objects Interfaces: Final Report

Student Researchers: Yuliya Dushkina and Shalva S. Landy

Advisor: Lori L. Scarlatos

Institution: Brooklyn University

by Shalva S. Landy

The Smart Objects project aims to enhance collaborative learning experiences by providing feedback to students working in a physical space. To do this, the system must "watch" as students manipulate physical objects, "decide" how close the students are to meeting their goals, and then "respond" appropriately. The specific application we were focusing on was a Tangram puzzle to be installed at the Goudreau Museum of Mathematics in Art and Science. The work would be general so it could be used for a wide range of other applications.

We are using a QuickCam as an "eye" to track the puzzle pieces. Image processing techniques applied to the camera's input determines which sides and/or corners of which pieces are adjacent to each other. Based on this information, we can determine the state of the puzzle. If the user has completed it, s/he receives a congratulatory animation; otherwise, we try to help the user if s/he seems to be having trouble. We give help by suggesting the user think about how two or more shapes may be used to construct one larger shape. For example, "how can you make a triangle out of two triangles and a square?" or "how can you make a larger triangle out of two smaller triangles?" Then, we play an animation showing how to do so. This is supposed to start the user thinking about how different shapes can fit together to ultimately form the complete solution.

Three people worked on the project this semester: Lori Scarlatos (mentor) and Shalva Landy and Yuliya Dushkina (students). We co-authored a paper titled, "TICLE: a Tangible Interface for Collaborative Learning Environments." This paper was accepted as a Late-Breaking Results paper for the ACM SIGCHI Conference on Human Factors in Computing Systems.

Although the project has not yet been completed, Dr. Scarlatos and I will continue working on it through the summer, so that it may be installed at the museum for the next school year.

I have learned from this project many interesting things regarding research that will hopefully help me when I do research in graduate school. First, there may be many different approaches to solve a problem, and the one that is taken may not necessarily be the best one. This calls for many unplanned changes during the course of the project. Also, research involves more than simply programming. Just because something sounds like a good idea does not mean that it will work. Another important thing is working with reliable people, but don't count on others; if you want the work to get done, do it yourself.

Visual Simulation Environments & Robot Drivers

Students:

Maralee La Barge
Emily Greenfest
Sarah Klaum

Faculty Supervisor:

Deepak Kumar

Department of Mathematics & Computer Science
Bryn Mawr College
101 N. Merion Ave
Bryn Mawr, PA 19010

Final Project Summary

The goal of our project was to write a Macintosh virtual simulation of the Khepera robot, based loosely upon work done by Olivier Michel, University of Nice, which was written for the UNIX environment. Completing the project involved three major steps:

- 1) developing an application for the Macintosh OS
- 2) modeling the physical Khepera robot accurately in a virtual environment
- 3) learning how to interface with a physical Khepera robot through the serial port.

First, developing an application for the Macintosh involved learning PowerPlant (part of the Metrowerks Code Warrior IDE), a collection of wrapper classes that hide the foundation level details of the operating system. We were unfamiliar with the package at the beginning of the project. Due to lack of tutorial resources on PowerPlant, we spent more time learning this development environment than we might have otherwise. By the end of the first semester, we had completed a smaller project similar to what we hoped to attain with the Khepera simulation. It brought to life the "Grid World" agent, an AI model of stimulus response behaviors defined in Nils Nilsson's text, *Artificial Intelligence: A New Synthesis*. Our GridWorld is freeware, available for downloading at

<http://mainline.brynmawr.edu/~mlabarge/darts/GridWorld.html>

Through GridWorld, we learned essential ingredients for the Khepera simulation, such as creating a single-window graphics interface and menu handling, as well as the more advanced task of thread management. While these ideas were applied to the more complex Khepera simulation, completing the full project included the additional task of learning how to implement a multi-window GUI, as well as the facility of saving, opening, and printing files.

The second step involved developing an accurate virtual model of the physical Khepera robot. This portion of the project initially relied heavily on Michel's

implementation. However, as our project progressed, it became increasingly clear that the mathematical representation of the Khepera robot used in the UNIX simulation was incorrect in some areas. Consequently, we found it necessary to undertake the challenging task of developing new mathematical models for collision detection, sensor readings, and robot motion. At this time, collision detection has yet to be perfected, but our simulation currently models most behaviors of the physical Khepera more accurately than the UNIX version.

Third, learning how to interface with the physical Khepera through the serial port proved to be the most challenging (and least successful) portion of our project. The task, which differed from other two as it involved unfamiliar hardware concepts, was made more difficult by a lack of current information. Consequently, the serial port interface to the physical Khepera is still incomplete.

In sum, we feel that we have accomplished the majority of the work we set out for ourselves. At this time, we have implemented a working Macintosh simulation for the Khepera robot. The simulation, which provides a fully functional GUI interface, allows users to create a virtual world with which the model robot may interact and to write a program controlling that interaction. User defined programs can utilize provided library functions that hide the inner workings of the simulation. We hope that future students may be willing to undertake the task of completing the project in full.

Trailblazing for the World-Wide Web

Student Researchers: Rachel Heck, Sarah Luebke, Weichao Ma, Hilary Mason

Advisors: Samuel A. Rebelsky

Institution: Grinnel College

This year we designed, built and began testing trailblazing tools for the World-Wide Web. These tools allow users to create and maintain ordered sequences of web pages that others can follow. We began our project by designing multiple templates for displaying such a trail. Most test subjects found some features of the designs useful, while others were more particular about other features. For instance, one template displayed the trail in a frame-based format. Some test subjects found this easy to use and aesthetically pleasing, while other subjects preferred a way of displaying the trails without frames.

After this testing phase, we determined that allowing users to specify some preferences would be desirable, but in the constraints of the school year, we decided to implement a default in such a way that we could add in user preferences later. For example, in our current default trail representation, we have a header and footer which give the trail information. Some people did not like having both because they felt it cluttered the page. With the use of preferences, we could allow the user to specify whether they want both the header and the footer or not.

Once we had developed a way to display trails we worked on two tools for creating them. We chose to appeal to users with varying experiences by implementing both a faster tool that requires computer expertise, and a slower, but more user-friendly tool that uses a WWW interface.

Overall our research project was a valuable experience, and it has encouraged each of us to continue doing research in Computer Science. We were only able to begin testing of the tools we built this year. Next year we hope to explore usability issues further and design the preferences mentioned above. Since we have found this experience rewarding and we believe it encourages women starting out in the field to continue with Computer Science, we intend to recruit first and second year women to work with our team. This will give the more senior research team members a chance to be mentors, as well.

One observation we made was that it became difficult to coordinate the research project with course work, and at times we had to adjust our schedules to fit these needs. For example, we were expecting to be able to test out tools for creating trails before the end of the semester but due to spring break, we were not able to completely meet this goal.

We found that meeting once a week or more as a group with the faculty mentor was effective. We also communicated by e-mail frequently.

Faculty Sponsor's Report on CREW Project

Project Title: Trail-blazing on the World-Wide Web

Faculty Sponsor: Samuel A. Rebelsky

Students: Rachel Heck, Sarah Luebke, Weichao Ma, Hilary Mason

Goals and Purpose

As is presumably the case in all CREW projects, there were both research goals and "personal" goals in the "Trail-blazers" research project. There was one primary research goal: to understand how people might understand, build, and use "trails", linked sequences of annotated pages, on the World-Wide Web. In support of that research goal, the students had a number of technical goals: to develop a file format for representing trails; to develop an interface or interfaces for presenting trails; to develop tools for building trails; and to do formative evaluation of the tools and interfaces.

On a more personal side, the students had goals of participating in a research project and learning not only research techniques, but also the joys and pitfalls of research.

Account of Process

Because the group had four student researchers, and the CREW grant only provided funding for three, an early order-of-business was to apply to the college for additional funds so that each student researcher could receive the recommended stipend of \$1000. The faculty mentor was responsible for the proposal, which was funded through normal college channels. The faculty member was also responsible for communication with the CREW administration when there were delays in receiving funds.

With the funding arranged, our first real orders-of-business were to develop a plan and schedule and to determine roles within the research team. The team decided that weekly whole-group meetings (including the faculty sponsor) were appropriate, with additional smaller-group or "sponsorless" meetings when needed. Ms. Mason agreed to serve as director of the group, and Ms. Luebke volunteered to serve as "rappateur".

As is appropriate in all projects, we then reviewed the proposal that funded the project and began a literature search to identify related projects. Although the student researchers did read some of the papers referenced in the original proposal, they were initially not able to identify related papers. (It may also be that they wanted to move ahead in the project and I was perhaps too willing to let them do so.)

Our first goal was to identify the interface most appropriate for presenting trails. We began with discussions of potential interfaces for showing trails. These interfaces included a framed version, with "table-of-contents" on the left; a simple "next/previous" interface; a slightly richer "numbered pages" interface, which included a list of pages running across the top and bottom of each page; and a multiple-window interface, in which the table-of-contents was presented in a separate window. In preparation for discussions with the research subjects, the student researchers created a number of sample trails,

and then manually developed each form of interface for each trail.

The student researchers then began some informal formative evaluation of the various interfaces using a variety of subjects culled from the college community. Rather than looking at particular numeric factors, such as which form of trail permitted readers to scan the material the fastest, the researchers decided to target more aesthetic qualities, asking subjects which they preferred, and determining which best indicate the "trail" quality to the clients. Before beginning these interviews (which are a form of research with human subjects), the researchers submitted a proposal to the Institutional Review Board which was approved. The faculty sponsor was responsible for much of this proposal, although the student researchers contributed to its development and provided the main questionnaire.

The evaluation revealed some surprising issues. For example, subjects who indicated that they didn't like frames in general nonetheless found the "framed with table-of-contents" interface appealing. In addition, many subjects preferred less information. In particular, few subjects wanted to see titles of all the pages in the trail. The research team developed a single interface, based on the information they had gathered from these subjects.

In the next phase, the students developed a format for "trail files" and then each student researcher chose a different development project to support the construction of trails. One student was responsible for turning "trail files" into a sequence of Web pages. A second was responsible for building a Web-based tool for constructing trail files. A third was responsible for building a tool that extracted links from an existing page and turned those links into a trail file. The fourth was responsible for support architecture, particularly support for downloading and manipulating files. The development and tuning of the tools took significantly longer than the students or the faculty sponsor expected.

During this development time, the student researchers also began to consider a number of other issues. In particular, they began to decide upon the next round of testing, which they decided would involve testing of the trail-construction tools. Towards the end of the development phase, they wrote a second proposal for the institutional review board. This time, the students were able to write the complete proposal, with little input from the faculty sponsor. Because they expected that the second study would involve more time and effort from the subjects, the researchers also began to look at mechanisms for compensating participants. They talked to a number of possible subjects, trying to identify appropriate compensation (enough to encourage subjects to participate; not too much, so that there were sufficient funds to support a number of subjects). They discovered that the compensation was more difficult than in many such experiments because of the odd funding of the CREW program. Typically, the institution would receive the funds and distribute them to the subjects when the researchers requested them to do so. Unfortunately, the CREW funds were handled from a separate institution, an arrangement not amenable to such small-scale funding. As a compromise, they settled on gift certificates from the college bookstore. This permitted mass purchase but individual distribution.

The tools neared completion at an awkward time; just before Spring break. It was clear that the college's long Spring break (two weeks) impeded progress. It was not possible to gather subjects before Spring break, and the

fine-tuning that was necessary right before Spring break took much longer than expected. In fact, assorted delays resulted in the interviews occurring too close to the end of the semester for all the recruited subjects to be available. Nonetheless, they gathered some interesting information, and the students expect to write a paper addressing their results in the near future.

The last half of the second semester also raised a number of related issues. The group gave an impromptu presentation to a visiting human factors scholar. The group identified some related research (Walden's Trails) and began to compare their own efforts and results to that related research. The group began to consider the form of the final report, and encountered the difficulties of developing a research report in the midst of other activities (e. g., classes, finals, work, the research itself). The group also began to develop a second CREW proposal, one intended to share their own good experiences by bringing less-experienced women into the research progress, and discovered the difficulty of coordinating grant writing and submissions within a group. The group also attempted to arrange funding from the Conference Experiences for Women program for travel to a conference on educational hypermedia, but were saddened to discover that that program no longer has funds. In the last week of the semester, the team also found an unexpected downside to their work: one of the trails that their subjects had constructed involved pages from another site, and the owners of the other site discovered the trail and objected to what the owners considered "thievery". Fortunately, a short explanation of the research project repaired the confusion.

Conclusions and Results

There are three main research outcomes from this project. First, we have evidence to show that both expert and novice Web users can understand the concept of trails, and can envision building and using trails. Second, we have some information on the interfaces that readers seem to prefer when using trails. Third, we have a number of tools that can be used for building and presenting trails on the World-Wide Web.

We do not consider our project complete. We clearly need more experience with "blazers" building new trails and readers following those trails. We also hope to do some comparative testing of the interfaces we and others have developed. We have applied to CREW for additional funding, but expect to continue the project in any case.

Lessons Learned

The students learned a number of important lessons about research and doing research. They worked on many of the technical skills necessary to be successful in research: writing grants, writing reports, coordinating subjects, meeting formal requirements (such as those of the Institutional Review Board), developing surveys, and otherwise building the infrastructure for research. Obviously, they have not yet mastered all of these skills, but they are on their way.

Many of the informal lessons will also stick with them. In particular, they

learned that many things take longer than anyone expects, that it's difficult to coordinate many people and many parts of research, that it is important to work harder at identifying background materials, and that the rhythms of research do not always correspond conveniently to the rhythms of the academic year.

Other Outcomes

Happily, this was a good experience for all of the student researchers (and for the faculty mentor). All four indicate that they plan to go on to graduate school in computer science when they complete their undergraduate degrees. This summer, two of the students (Heck and Luebke) will be working on a related research project, one student (Ma) will be working on another research project at the college, and the fourth student (Mason) will be participating in a research internship at AT&T labs.

Final Report, Collaborative Research Experience for Women in Undergraduate
Computer Science and Engineering

Students: Michele R. Cofield and Kamilah K. Walker

Faculty: Dr. Albert C. Esterline

Institution: Department of Computer Science, North Carolina A&T State
University, Greensboro, NC 27411

Our one year of research addressed multiagent motion planning. We distinguish high- and low-level aspects of motion planning. The low-level aspects involve laying out paths with explicit metric properties. The high-level aspects involve task structures, agent intentions, and generally more abstract features. Our goals were to extend the low-level implementation we had begun, translate C code to Java, address high-level planning, and integrate high- and low-level aspects as much as possible. We held weekly meetings to formulate algorithms and to review code. Standard sources on planning were consulted. Starting with the low-level aspects, path planning is handled differently in static and dynamic environments. When we began, we had already implemented in C a significant part of a visibility-graph approach to static path planning. This code has been completed and most of it has been translated into Java. We also completed Java code for dynamic environments where objects have constant linear motions. This code used an accessibility-graph approach. A start has been made on a graphical user interface for the dynamic case.

We have developed algorithms for special cases of multiagent path planning, which apply in the first instance to static environments. We assume that agents are associated into groups. All the agents in a group are assumed to have start and destination points near to each other and to move in roughly the same directions (generally linear) with roughly the same speed. We also assume that the environment is sparse, that agents are prioritized within groups, and that the groups are prioritized among themselves. We approach such a problem hierarchically. For each group, we find a rectangular envelope with a constant linear motion that encloses all the agents throughout. This envelope should be as small as possible. After the paths of the individual agents are computed (disregarding the other agents), collisions among agents within a group are found. A collision between two agents is resolved by delaying the agent with lower priority just enough to avoid the collision. Once all collisions within groups are resolved, the possibility of collisions between agents in different groups is eliminated by resolving collisions between pairs of envelopes. When two envelopes collide, the envelope corresponding to the group with the lower priority is delayed just enough to avoid the collision. This delay is added to the delays already imposed on its members. The generalization of this approach to dynamic environments can be done rather straightforwardly by treating obstacles as highest-priority agents.

For high-level aspects of motion planning, we have worked with Statechart representations of multiagent plans that are being used in Dr. Esterline's research. We have written Java code that interprets the concurrent behavior of a system of agents which have been assigned roles in a plan Statechart. The code exploits Java's monitors.

Not much progress was made on integrating high- and low-level aspects of motion planning. The key to this integration would appear to involve developing ways for high-level decisions to impose constraints on the low-level planning algorithms. It would also be helpful to have some way to classify low-level

configurations in ways that are meaningful to a high-level planner. We feel fortunate that we took an incremental approach with a topic with which we already had some experience. Motion planning was found to be an excellent topic for advanced undergraduate research. It is manageable yet offers unlimited opportunities for developing sophisticated techniques. We presented our research at the North Carolina Alliance for Minority Participation Conference (North Carolina Central University, Raleigh) in April. We also have a paper in the Proceedings of the NASA ACE/PURSUE Student Conference, also held in April.

Report to the Computing Research Association Committee on the Status of Women in Computing Research Collaborative Research Experience for Women

Project Title: Processor Allocation to Independent Tasks in Low Dimensional Mesh Systems.

Collaborators: Erin B. Greening, student
Jessica (Tze-Yun) Lin, student
Bonnie E. Melhart, advisor
Craig A. Morgenstern, advisor

Our research project has been concerned with the allocation of processors to independent tasks in a two-dimensional mesh system. We have focused on heuristics to manage the tradeoff between the competing goals of optimizing for high processor utilization versus optimizing for low inter-task bandwidth contention.

Goals and purpose for the project

Our project goal was four-fold:

- * to examine the performance of existing allocation strategies,
- * to propose, develop, and test new allocation strategies to try to raise processor utilization while lowering task completion time,
- * to determine how well new and existing strategies scale as the mesh and/or communication requirements increase, and
- * to provide a realistic and successful research experience for undergraduate students in Computer Science.

The purpose of the project was to better understand the problems of such mesh systems and to contribute new strategies for allocation and scheduling that improve the overall system efficiency in the presence of communication between processors allocated to independent tasks.

Process used in completing the research

We began weekly meetings last fall semester. The advisors distributed a few essential papers and we discussed those together at first and made plans for the next weeks activities. We acquired the Proximity simulator, developed at the University of Oregon, and began running experiments to duplicate the findings we had read about. This served to acquaint us with the workings of the simulator and helped us to understand the technical papers.

As we ran more experiments, we examined the simulation results and strategized about new allocation algorithms that might take advantage of the strong points of previous ones without the weak points they exhibited. For each new strategy, a series of experiments was designed to compare it to old strategies. As we began to consider the added problems of communication between the processors, we added experimental measurements for the effects of this to our simulation studies. We considered various messaging patterns and ways to simulate that would allow us to run the experiments efficiently. (Some experiments use several days of machine time to finish.)

Conclusions and results achieved

Our studies included comparisons of existing allocation strategies and a new strategy we developed that is a variation of the contiguous Fast Frame Sliding (FFS). Our Closest Shape (CS) strategy allows allocations the FFS would not by considering submeshes of a different size and shape than the request. For

example, suppose a task requests a 3x5 submesh of 15 processors when there are no such submesh blocks available. If there are free submeshes of 2x8 or 4x4, FFS will not allocate, but CS will. None of the contiguous strategies can match the utilization when scattered (non-contiguous) allocation is allowed, however. Intuitively, utilization will be favored with scattered allocations if there is no communication between the processors.

Our further studies looked at the same strategies when communication is considered among the processors. Scattered allocations we studied included random and multiple-buddy allocation strategies. The Proximity simulator allowed us to simulate various communication levels between the processors allocated to a specific task. Intuitively, high levels of communication will be handled better with contiguous allocations. It seems reasonable that one would want to at least limit the length of the path between communicating processors, though it is certainly possible to pass messages through as many other processors as it takes to deliver the message, including those allocated to other tasks. For our experiments we used wormhole routing, an XY-routing scheme that routes a message in the horizontal direction first and then in the vertical direction to its destination. The whole message follows along from processor to processor in a pipeline fashion.

We have simulated these strategies extensively for 16x16 and 32x32 meshes and have preliminary results for 64x64 mesh systems. In all cases, the utilization is better for the scattered allocation multiple-buddy. As the mesh size grows larger, the contiguous strategies improve their utilization for heavy traffic, while the scattered allocation maintains the same high utilization. Closest Shape has the highest of the contiguous strategies. The response time and finish time of contiguous vs. non-contiguous fluctuated depending on the severity of messaging traffic and the mesh size. The observed trend is that as the mesh system gets larger, there is little difference in performance between the best of the contiguous methods and the best of the non-contiguous methods. That is, contiguous methods can be competitive with non-contiguous methods. However, more studies are needed to determine the benefits and trade-offs of each type of allocation strategy. We hope to continue the studies with these strategies for the largest mesh systems that we can simulate. There is still much to explore for ways to improve the contiguous vs. non-contiguous allocations when processors require communication.

Our most useful outcomes are the new ideas for areas to consider with further heuristics and experimentation. The funding from the CREW program made it possible to get other funding from our university for equipment that we will be able to use solely for this project next year.

Jessica Lin is starting graduate school in Computer Science at the University of Texas at Arlington, while Erin Greening is starting her senior year of studies at TCU and plans to continue her involvement with this research project.

Crew research project - final report:

- goals and purpose of the project

Humans are very good at picking out composite features to use as landmarks in unstructured environments and to use those landmarks to navigate or identify objects around themselves. The goal of this work was to decide what features in a typical unstructured environment an automated navigation system could use as landmarks and how properties of those features could be combined to provide localization superior to that obtained by using beacons or point features.

- account of the process used in completing the research

Most of the previous research using landmarks in outdoor environments has used point features, such as mountain peaks, as opposed to composite features. Point features are easier to identify but produce more error when used for localization than do composite features. To identify a composite land feature, a precise definition must be known. Since no clear, formal definitions existed for any of these features, they had to be formulated. By interpreting previous work and consulting with researchers in the area, the terms for such features, such as ridge line, saddle, circ and bowl, and their definitions were developed. The features were separated into two categories: Primitives and Formations. The Primitives are themselves composite while the Formations are groupings of Primitive composite features. It was decided to focus this work on ridge lines since the La Crosse area provided numerous examples with which to work.

Digital images of the bluffs surrounding La Crosse were taken. An image library was created consisting of images taken at multiple locations and from different viewpoints at each location in order to clearly capture distinguishable land features within the bluffs.

USGS (United States Geological Survey) elevation data for the La Crosse area was obtained and used to render a terrain map of the area. The data used was 7.5 minute DEM (Digital Elevation Model) with readings taken every 30 meters. This allowed for simulated terrain to be rendered covering the same areas as the images in the library and for the elevation at any given point to be electronically available.

The digital images were then converted into a form that allowed for easy access to the actual pixel values. They were cropped so that extraneous features in the image were eliminated and ridge lines were featured. The cropped images were then converted from color to grey scale, then into a matrix of numerical values with each value ranging from 0, representing black, to 255, representing white. These pixel values were then written into a text file. Visual patterns of pixel values were visible in this text file. An algorithm was developed to color different ranges of pixel values different colors. It was found that there was a unique number pattern that represented the ridge line. The unique pattern consisted of a contrasting pattern of lighter number values located in close proximity to darker number values. Although ridge lines along the sky line were easy to find, these patterns were also apparent along ridge lines that did not border the sky.

Finally, pattern recognition algorithms were developed in an attempt to identify this unique pattern that represented the ridge line. The goal was to

produce from the pixel intensity values a single entity which could be labeled ``Ridge line''. Although ridge lines on the map as well as on the scene rendered from the map data are shown as lines, they appear as blobs in the actual images. Once the blobs are identified, they must be compared in some way to the linear map data in order to match the view to the map.

The first algorithm pulled out pixels with values exceeding a given threshold. There were two problems with this approach:

1. When images are taken in an outdoor setting, there is no control over the light intensity. Images existed in the library with ridge lines visible to the human eye which had no pixel values over the threshold.
2. It was then necessary to add another level of processing to the algorithm in order to decide which of these thresholded pixels were part of the ridge line and which were outliers and to connect the legitimate pixels into a single entity.

At this point, it was decided that, to deal with the light intensity problem, instead of an absolute threshold, a difference measure should be used, pulling out pixels which differed from their surroundings by a percentage of the total range of pixel values. The second problem was dealt with by grouping pixels into a blob, using a standard region growing algorithm. A ``Ridge line signature'' was then developed for each connected blob for matching with the map ridge lines.

- conclusions and results achieved

A set of rigorous definitions for composite features in outdoor environments was developed.

It was found that there is a unique pixel value pattern when a ridge line appears in a digital picture and this pattern can be extracted using a pattern recognition algorithm.

After examining the U.S. Geological Survey data available, it was clear that the data is smoothed to the point that many land features commonly recognized by a human observer, such as cliffs and outcroppings of rock are lost in the smoothing. Although the U.S. Geological Survey data might be of more value if it included more data using smaller intervals than 30 meters, the additional data would then add to any required processing time.

It should therefore be noted that using the USGS data for localization on a map will require that large scale features be utilized rather than some of the smaller distinguishable features so often used by humans.

Weather conditions can change the view significantly. For example, when the sky is overcast, the image is dark, leading to a smaller range of pixel values and less likelihood of picking up differences that signify ridge lines. A scaling of the pixel values can often widen the spread, but only if there are not too many outliers to skew the distribution. As an example, an image taken with an overcast sky might contain pixels in the range of 0 to 150. By scaling the pixel values, the range can be changed to 0 to 255, providing a larger difference in values along the ridge lines. However, a patch of white in the image, such as a piece of pavement in the foreground, produces outliers and will skew the distribution and mitigate the effect of the scaling. Vegetative growth can also change the look of the surroundings. A tree line may be

mistaken as a ridge line when analyzing the digital image. However, it should be noted that human navigators also occasionally mistake tree lines for ridge lines.

- students and mentor(s) involved:

Students: Jennifer Davison, Kelly Hasler
Faculty Mentor: Karen T. Sutherland Ph. D.

Department of Computer Science
University of Wisconsin - La Crosse
La Crosse, WI 54601