TIC-TAC-LEGO: An Investigation into Coordinated Robotic Control

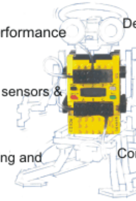
Ruben, Vuittonet, Jr.
University of Alabama At Birmingham
rvit@uab.edu

Jeff Gray
University of Alabama At Birmingham
gray@cis.uab.edu

Department of Computer and Information Sciences

Embedded Systems

Special-purpose computer systems encapsulated by the devices they control



Compromise:
size versus performance

Designed on one System
Deployed on Another
Firmware

Often deal with sensors & activators

Often deal in real-time

Information hiding and Safety Issues

Compiled, Interpreted, ML

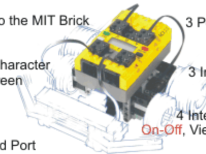
Loop control structures
for repetitious input

The goal?

To execute as quickly as possible in an asynchronous world using the smallest amount of code with the highest level of predictability

THE RCX® 2.0 BRICK

Hitachi H8300 Processor, 16MHz, 32K RAM
28K available for safe firmware use



Similar to the MIT Brick

3 Power Sources

1 Five-Character
LCD Screen

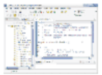
3 Input Sensors

4 Interface Buttons
On-Off, View, Run, Prgm

1 Infrared Port
(Uploading firmware, software, and communication)

Power Supply: 6 1.5 volt AA Batterie

The default firmware supports up to 5 programs in the RCX Brick

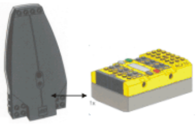


Eclipse Plug-in for leJOS

leJOS is a Tiny JavaTM runtime.
Replaces the Lego® Mindstorms firmware

Memory footstamp 17K
32K – 17K = 15K
(important later)

leJOS implements
- OO Language
- Preemptive Threads
- Multi-dimensional arrays
- Recursion
- Synchronization
- Exceptions
- Well-documented API



The leJOS firmware is installed through USB Tower to the RCX® via infrared

CLASS DECOMPOSITION

Tic-Tac-Lego:

a set of two embedded systems to coordinate the play of Tic-Tac-Toe

Robotics and Reusability

Robotics provides ample opportunity for component abstraction.
Robotics provides ample opportunity for component reuse.

The Classes

ArmRobot
Display
Locations
RollRotateRobot
RotationListener
ScanLocations
TicTacLocations
TicTacRcx
TicTacRcxMoverComm
TicTacRcxScanComm
TicTacRcxScannerComm

Board
Location
PieceLocations
WorkRobot
RotationAxel
Scanner
TicTacNegamax
TicTacRcxMover
TicTacRcxScan
TicTacRcxScanner

leJOS is an interpreter

Remember, the leJOS firmware must be installed = 17K footstamp Next, the application must be installed. The combined code and firmware exceeded available memory.

READING THE GAME BOARD

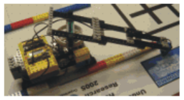
ScannerBot Software Components

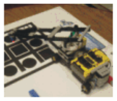
TicTacRcxScan
Location
RollRotateRobot
Board
RotationListener
ScanLocations
TicTacNegamax
TicTacRcxScanComm

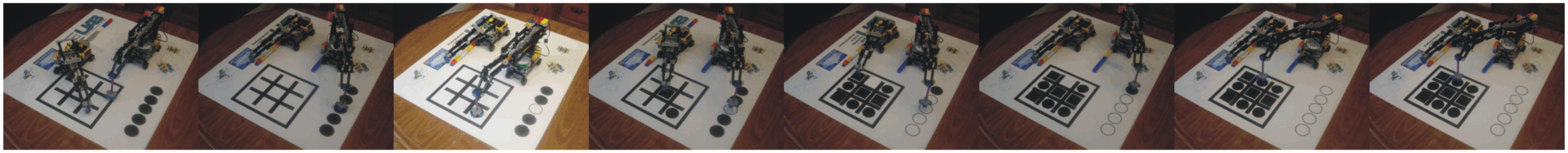
Display
Locations
RotationAxel
WorkRobot
TicTacRcxScanner
Scanner
TicTacRcxMover

The Scanning Process

On startup, ScannerBot enters a *loop*
ScannerBot Waits for *Prgm* to be pressed
When pressed, ScannerBot Scans the board
After the Scan, ScannerBot computes *Next Move*
Next move is transmitted to the MarkerBot







COMPUTATION OF THE NEXT MOVE

Negamax utilized to compute next move

1. Equivalent to Min-Max algorithm
2. Tic-Tac-Toe is monotonic
3. Recursive ply depth is ≤ 9
4. Not using α/β pruning allows exploration of parallelism

Parallelism

1. Occurs when the number of remaining moves is 7
2. ScannerBot plays 'X' or 'O' player
3. The board is transmitted to the MarkerBot
4. The MarkerBot and ScannerBot compute *Next Move* in parallel
5. MarkerBot transmits its result to ScannerBot
6. ScannerBot elects to use the appropriate result
7. ScannerBot transmits the selected *Next Move* to MarkerBot

NAVIGATION & BEHAVIOR

Navigation Components Heavily Reused

ScannerBot: rolls and rotates
MarkerBot: rolls, rotates, and places

Coordinated Component:

the Rotation Axel
Rotation Sensor
Motor

Encapsulation enables more precise navigation

The Nature of Embedded Systems

Embedded Systems behavior for Tic-Tac-Toe
Limited Interface to Embedded System
Sensors Brick Buttons
Specific Classes Drive System Behavior

TicTacRcx
Location
PieceLocations
TicTacLocations

TicTacRcxScan
Locations
ScanLocations

CONSTRAINTS & CLASS REFACTORING

The RCX 2.0s available memory is 32K

The leJOS Firmware is 17K

The Code + the Firmware was > 32K

Solution: refactor
- Make data members public
- Remove accessor and mutator methods
- Code exposure is a known aspect of embedded systems

ScannerBot's leJOS byte code is 13.2K
(17 * 1024 + 13572) = 30980 bytes < 32K

SUMMARY & CONCLUSION

Embedded Systems provide effective solutions in many critical real-time processing situations

In fact, the majority of processors are manufactured for embedded systems

The Lego RCX 2.0 Mindstorms Robotic Invention Systems provides an out of the box opportunity to explore the capabilities and limitations of embedded systems programming

The RCX provides the novice programmer entertaining opportunities to develop software engineering skills, including abstraction and encapsulation, while playing with a fun toy.