CSCI 1106
Lecture 9

Using State Transition Diagrams

Announcements

• Today’s Topics
  – States, transitions, and program blocks
  – Mapping state transition diagrams to programs
  – Discussion
Recall

States and Transitions

• State
  – Unique set of conditions
  – Describes a step of a task
  – Represented by a circle and a label

• Transition
  – Change of one or more conditions
  – Describes a change from one state to another
  – Represented by a labeled arc

Program Blocks

• Sense blocks

• Decide blocks
  – wait
  – loop

• Action blocks

Program Blocks

Translating State Transition Diagrams

• Problem:
  – We design our solution by creating a state transition diagram (STD)
  – We need to translate the STD into a program

• Idea: Divide and Conquer
  – Identify standard patterns in STD
  – Implement each pattern using one or more program blocks

• How do we identify the patterns?
Standard Patterns

• Idea: We consider all the blocks and ask, what patterns do they implement?
• We start with simplest: Action Blocks
  – Synchronous Move: 
  – Asynchronous Move: 
  – Any other action will yield the same patterns

Example: Move in a Square

• The “stop” state is implicit in the program
• Transitions are triggered by internal events (action completions)
• The program can be shortened by using a loop
The Wait Block (Sense-Decide)

- The Wait block is used to represent a state and a transition due to an external event
- Use it after an asynchronous action
- Example: Move forward until an object is encountered
- Note: Only one transition can occur from an “action + wait” state.
  - Why?
  - Why would this be inadequate?
  - How do we get around this?

The Loop Block

- The Loop Block is used to implement loop structures in a STD
- The loop allows us to repeat a state
- Easy if you have only one cycle
  - Even if more than two states
- Questions:
  - What if you have multiple cycles?
  - What if you have multiple transitions from states?
- Answer: Bring out the “big guns”
  - The Switch (if) Block
The Switch (If) Block

• Recall the definition of a state ...
• Idea: Use a loop and switch blocks to implement multiple states and cycles
• Example: Follow the line
• Questions:
  – Where is the “wait”? 

The True Meaning of State

• Observations:
  – A state is defined by a set of conditions
  – A switch (if) block can be used to test for conditions
• Idea: Use switch blocks to test what state the system is in each time through the loop
Example: Follow but Stop

- Follow the line but pause for objects in the way.

![Diagram showing a robot following a line and stopping for objects]

More Complexity

- Suppose we wanted to add another constraint:
  - Stop and quit if rear touch sensor is activated.
- Our STD gets another state.
- The program becomes even more complex.
- How do we change the program?
Options for Implementing the Changes

- Add another Switch Statement
  - Standard, but adds complexity
- Change loop condition
  - STD is still complex
- Use multiple threads and simplify the STD
  - Two simple STDs are easier to implement than one complex STD

Simplifying the STD

- Break one STD into two
- Use one thread per STD
The Resulting Program

Discussion

• Can an STD or program always be split into multiple threads?
  – Why or why not?
• How can we ensure that the threads do not interfere with each other?
• Can “Wait” blocks be used inside Switch blocks?
• What are sensor blocks for?
Why Do We Need Sensor Blocks?

- Viewing the measurements that a sensor is reporting when debugging a program
- Keeping track of previous measurements to compare against future measurements
  - E.g., Finding a minimum/maximum measurement
- Creating multiconditional Wait blocks

Multiconditional Wait Blocks

- A Wait block polls a sensor until a threshold is reached
- Idea: Combine measurements from multiple sensors
  - Polling
  - A sensor is below threshold
  - All sensors are above threshold