CSCI 1108

Dealing with Failure
Motivation

• The world is imperfect
  – Sensors give wrong readings
  – Motors turn too fast, too slow, too much, or too little
  – Wheels don’t grip the surface properly
  – Lighting conditions change

• This is normal
  – Humans deal with these kinds of problems all the time
  – We learn how to deal with failure

• How do we get robots to deal with them as well?
Dealing with Failure

• Need to do two things
  – Identify when a failure has occurred
  – Respond to the failure

• Example: Missing your exit on the highway
  – Identify that you have gone too far
  – Turn around and back track
Failure and Failure Cause

• **Def:** *Failure* is a state that is not anticipated by the design

• **Def:** *Failure cause* is the physical or functional reason for the failure
  – I.e., Why did failure occur?
  – Also known as *failure mode*

• **Examples:**
  – The furnace stopped working because it ran out of oil
  – We missed the exit because we did not see the sign
  – The robot missed the line because it drove over it too quickly

• **Key Observation:**
  – We can only deal with failures that we can foresee
  – I.e., What can go wrong?
Failure Manifestation

• **Def:** *Failure manifestation* is the detectable effect of the failure

• **Examples:**
  - The house is cold because the furnace is not working
  - We have driven too long because we missed the exit
  - Our arm hurts because we have broken it

• **Key Idea:** To identify failure, it must manifest itself in a detectable way
Failure Identification

• **Idea:** We can identify that a failure has occurred from its manifestation

• E.g., We identify that
  – The furnace must not be working because the house is cold
  – We must have missed the exit because we have driven too long
  – Our arm must be broken because it really hurts

• **Idea:** To identify a failure, we need to
  – Determine what can cause the failure
  – How the failure manifests
Enumerating Failures

• When designing a program we need to (attempt to) enumerate all relevant failures:
  – Assume things will go wrong
  – Ask “What can go wrong?”
  – Ask “How is failure manifested?”

• Narrow the enumeration to:
  – Failures we can deal with
  – Failure causes we understand
  – Failure manifestations we can identify

• Systems fail because designers fail to identify all relevant failure causes
Examples of Failures and Causes

• Ground proximity sensor fails to register dark / light
  – Sensor’s distance to ground changed
• Horizontal proximity sensor fails to register object
  – Object has an odd shape
  – Object has an odd surface
• Horizontal proximity sensor registers ghost objects
  – Other robots nearby emitting infra-red light
• Robot does not make sufficiently precise movement
  – Tires are not properly aligned
  – Motors are rotating too fast
  – Wheels don’t grip the surface properly

• In all cases the sensor or actuator may be broken
• How do we detect failures?
Mechanisms for Detecting Failure

• **Unexpected external events**
  – Sensors register an unexpected changes in environment
    • Sensors give false readings
    • Sensors give true readings of unexpected conditions
  – Actuators report status errors
    • Actuator fails to perform specified task
    • Actuator reports error where none has occurred

• **Lack of expected external events**
  – A timer expired while waiting for an expected event
    • Sensor fails to register the expected event
    • Expected event does not occur
  – Actuators fail to move the prescribed amount
    • Encounter unexpected resistance

• **Unexpected (or lack there of) internal events**
  – Programs run code they are not supposed to (bugs)

→ Unexpected: A difference between the anticipated and measured
Failure Response

• Once we determine that a failure has occurred, we need to respond to it
• **Def:** *Response mechanisms* are parts of the program that respond to the failure
• One approach is to put system in safe state and shut down
  – E.g., nuclear reactors
• This is not always possible if
  – System is inaccessible
    • E.g., rovers on Mars
  – System is mission critical
    • E.g., airplane
• In these cases the system must *recover* from the failures
Failure Recovery

• Recall: A failure occurs when a system enters an unexpected state
• **Def:** A *recovery mechanism* returns the system to a normal state

• Recovery mechanisms are specific to each failure
• Examples:
  – If an exit is missed, backtrack to the exit
  – If the furnace is broken, call landlord
  – If your arm is broken, see a doctor

• For our purposes: *return the robot to its last ``normal’’ state*
  – Find the line if it is lost
    • Recheck sensors
    • Retry actuator operation

• If the recovery mechanism fails, we need a recovery mechanism for the recovery mechanism...
Modeling Failure Identification and Recovery

• We need to model or represent how we
  – Identify failure
  – Respond to failure
  – Recover from failure

• What should we use?
State Transition Diagrams

- **Idea:** Use state transition diagrams to represent possible failures and recovery mechanisms
- **Example:** Crossing the Street

![State Transition Diagram for Crossing the Street]
A Missed Line in Follow the Line

- **Right state**
  - Sensor reports light
  - On left side of line
  - Moving to the right
  - Timer running

- **Left state**
  - Sensor reports dark
  - On the line
  - Moving to the left

- **Lost state**
  - Timer expired
  - Sensor reports light
  - On right side of the line
  - Moving to the left

- **Found state**
  - Sensor reports dark
  - On the line
  - Moving left
Observations

• Error identification and response can add much more complexity to your program
  – 80% of a typical application deals with error handling

• The error response itself may fail

• State transition diagrams are an easy way to reason about errors