Using Finite State Automata in Robotics

Good Practice Reccomendation

Richard BALOGH and David OBDRŽÁLEK

Slovak University of Technology in Bratislava, Charles University in Prague

richard.balogh@stuba.sk, david.obdrzalek@mff.cuni.cz

Abstract

We demonstrate the use of the finite state machines (FSM, state automata) in robotics, especially for implementing entry-level control systems. Several examples of real tasks are presented and implementation sketch which uses FSM is shown. Our aim is to show FSM can be a good tool for implementing control system of a robot, powerful as well as easy to be used.

Example: Generic FSM

A FSM can be represented in many was, as well as implemented in many ways.

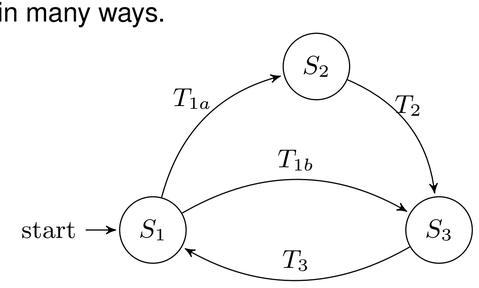


Figure 1: General example of the state machine diagram.

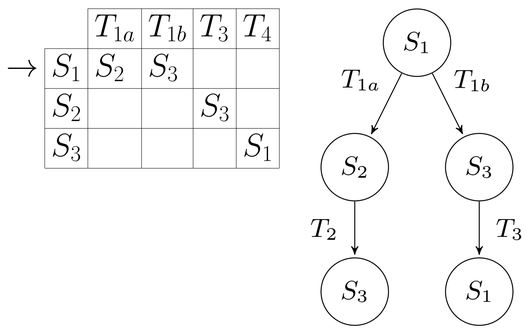


Figure 2: Alternative FSM representations of the same system.

```
enum states {S1, S2, S3};
enum states currentState = S1;

switch( currentState ) {
  case S1: if( T1a() ) currentState = S2;
           else if( T1b() ) currentState = S3;
           break;
  case S2: if( T2() ) currentState = S3;
           break;
  case S3: if( T3() ) currentState = S1;
           break;
}
```

Figure 3: Transition function implementation in C.

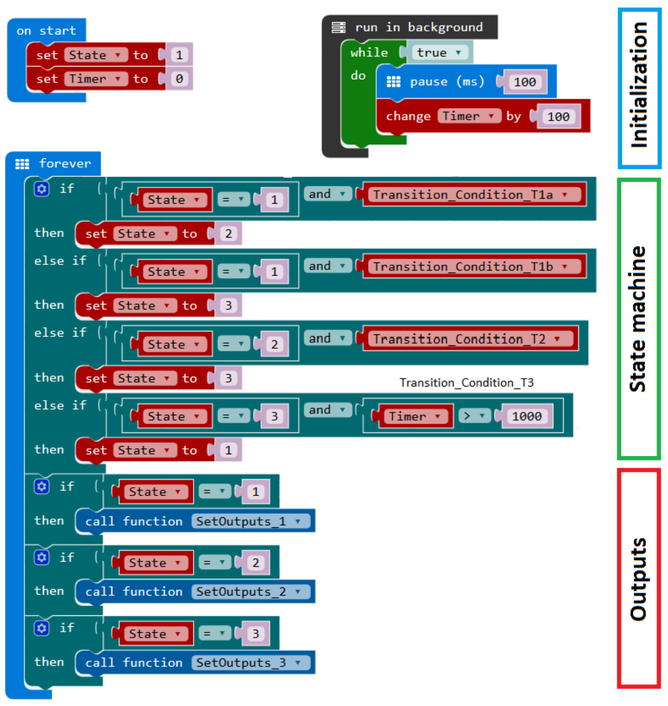


Figure 4: Microsoft Make Code Block implementation.

Example: ON-OFF switch

A simple push-button which toggles the state on every press.

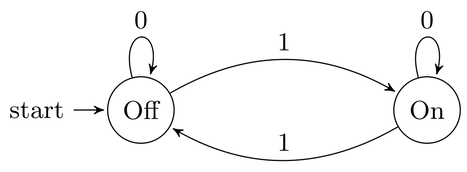


Figure 5: ON-OFF switch state machine diagram.

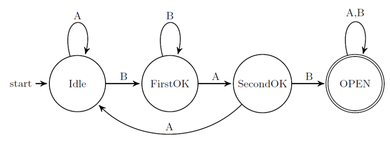
```
enum states {stateON, stateOFF};
enum states currentState = stateON;

while(1) {
  switch(currentState) {   // State machine
    case stateON:
      if(board.buttonA.isPressed())
        currentState = stateOFF;
      break;
    case stateOFF:
      if(board.buttonA.isPressed())
        currentState = stateON;
      break;
  }
  switch(currentState) {   // Actions
    case stateON:
      board.On();
      break;
    case stateOFF:
      board.Off();
      break;
  }
}
```

Figure 6: C implementation.

Example: Simple code lock

A simple code lock with two buttons (A, B) which responds to sequence B-A-B.



	A	B
Idle	Idle	FirstOK
FirstOK	SecondOK	FirstOK
SecondOK	Idle	Open
Open	Open	Open

Figure 7: Simple code lock state machine diagram.

```
enum states {Idle, FirstOK, SecondOK, Open};
static enum states stateTable[4][2]={ {Idle, FirstOK},
                                      {SecondOK, FirstOK},
                                      {Idle, Open},
                                      {Open ,Open} };

transition = readButtons();
nextState = stateTable[currentState][transition];
currentState = nextState;

switch(currentState) {
  case Idle:
  case FirstOK:
  case SecondOK:
    break;
  case Open:
    setOutput();
    break;
  default:
    break;
}
```

Figure 8: C implementation.

Example: Line Follower

A line follower robot with two black-white sensors.

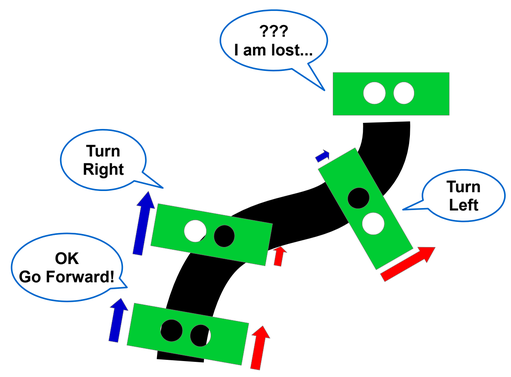


Figure 9: Linefollower robot with 2 sensors.

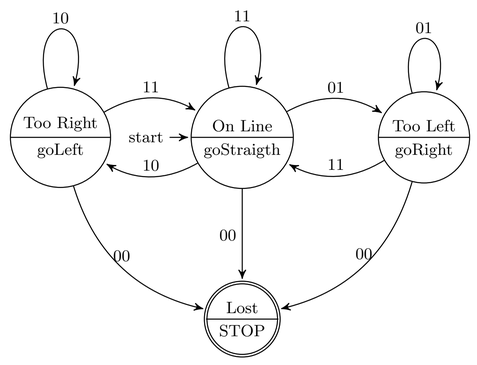


Figure 10: Line Follower state machine diagram.

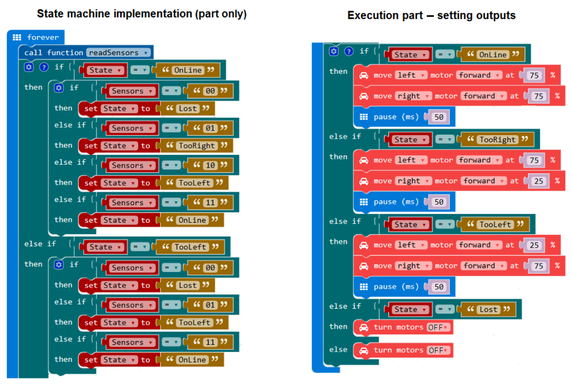


Figure 11: Microsoft Make Code Block implementation.

Example: Competition robot

A robot for a “Ketchup House” competition running on a grid 5x5 lines to collect cans of ketchup.

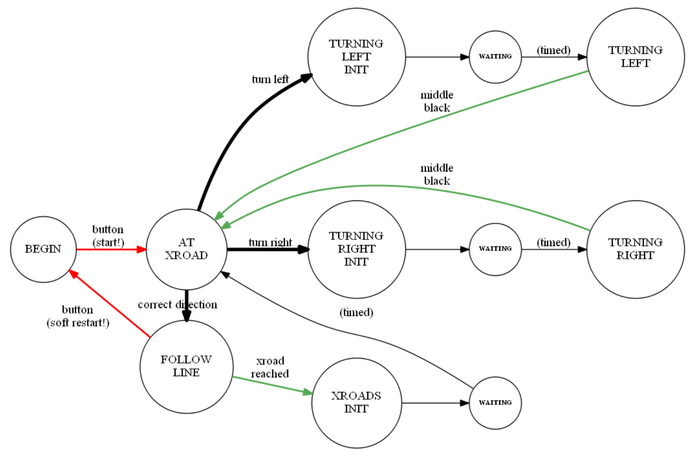


Figure 12: Main state machine for the MART Friday Bot

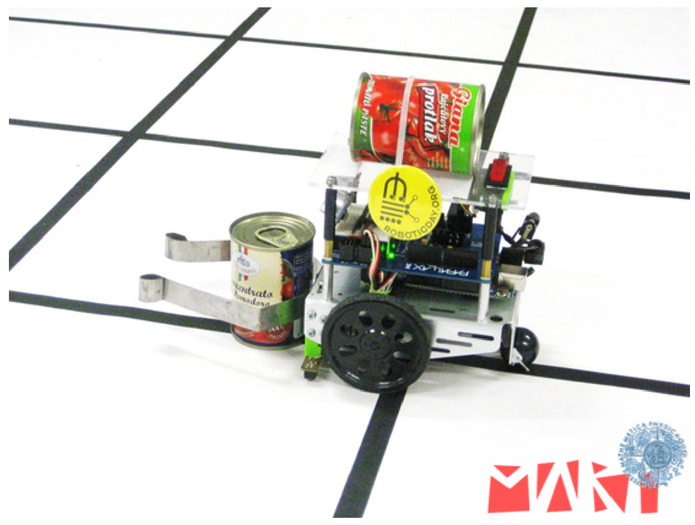


Figure 13: MART Friday Bot

Conclusions

Finite State Automata is a possible tool for implementing the core of a robot control. Our experiences show there are numerous task types where a FSM can significantly help at the implementation phase even if the task seems to be very simple at the first glance. Using the FSM can make the controlling code be readable, understandable and maintainable which is very important especially in education. Let us informally cite one student: “I have daubed everything somehow for the first exercise and it somehow more or less worked. Then I learned about FSMs and implemented the last exercise using it. Had I known it earlier, I would have used it straight on, it was so easy!”

Although there exist many implementations, libraries or systems based on FSMs, many of them are way too complex for beginners to understand and cannot be recommended as a starting point. Instead, we have shown how a simple FSM can be programmed from scratch in a simple way. That helps to understand the principles and allows for a fast progress towards real world usage.

References

[1] Full version of the paper and more details
https://goo.gl/3rswH2