From Formalised State Machines to Implementations of Robotic Controllers

UNIVERSITY of York

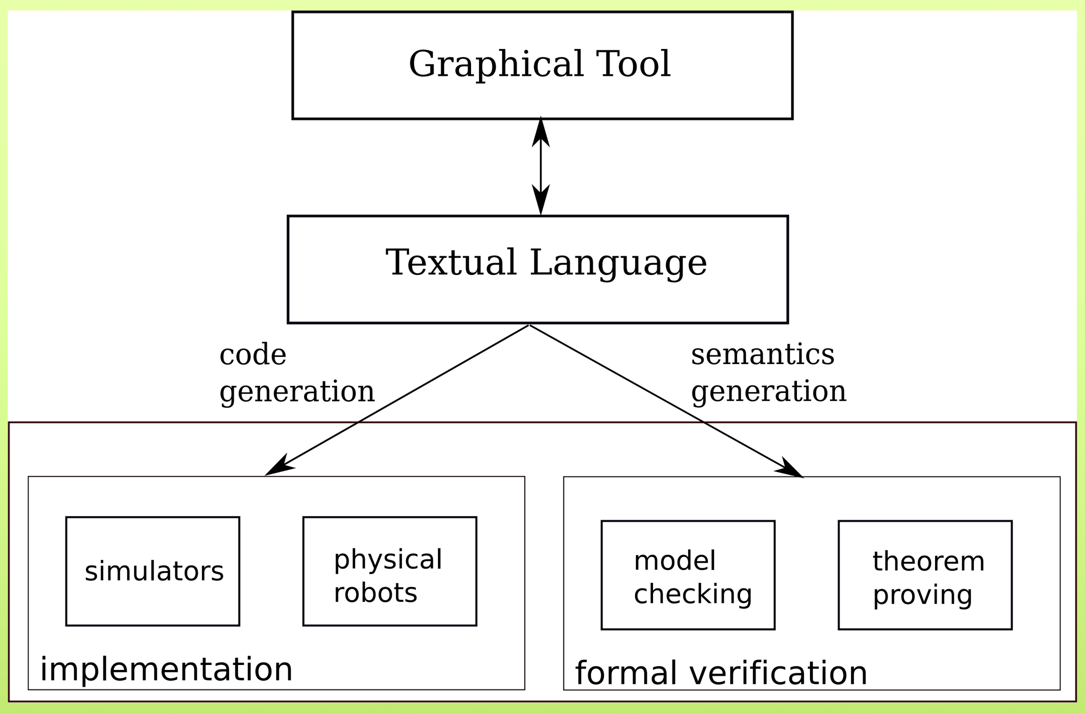
Wei Li[1], Alvaro Miyazawa[2], Pedro Ribeiro[2], Ana Cavalcanti[2], Jim Woodcock[2] and Jon Timmis[1]

[1]Department of Electronics, University of York, York, UK; [2]Department of Computer Science, University of York, York, UK

1. Abstract

Safety is a major concern for autonomous robots, and the ability to provide evidence that a robotic system is safe can be demanding. Robotic controllers can be specified using state machines. However, these are typically developed in an *ad hoc* manner without formal semantics, which makes it difficult to analyse the controller. We present a state-machine based notation, RoboChart, together with a tool to automatically create code from the state machines, establishing a rigorous connection between specification and implementation. RoboChart has a formal semantics that allows for formal verification [1]. We demonstrate our approach using two case studies (self-organized aggregation and swarm taxis) in swarm robotics. This paves the way for the verification of controller of individual robots in the swarm as well as their resulting emergent behaviours.

2. RoboChart framework



The RoboChart framework for combining formalised state machines and implementation of robotic controllers

Elements of RoboChart:

- Architecture for robotics
- State machine
- Interface
- Clock

Features of RoboChart:

- Graphical and textual modelling
- Formal semantics (CSP)
- Automatic code generation
- Platform independent

Automatic code generation:

- Model-View-Controller pattern
- Direct mapping from elements to entities in C++

RoboChart elements	State machine class
states clocks interfaces	attributes of enumerated type attributes of timer class inherit interface class

RoboChart elements	Interface class
events variables operations	attributes of enumerated type attributes methods

3. Case study one: Aggregation

Behaviour: Robots aggregate into a single compact cluster as fast as possible.

Controller: two states; two events

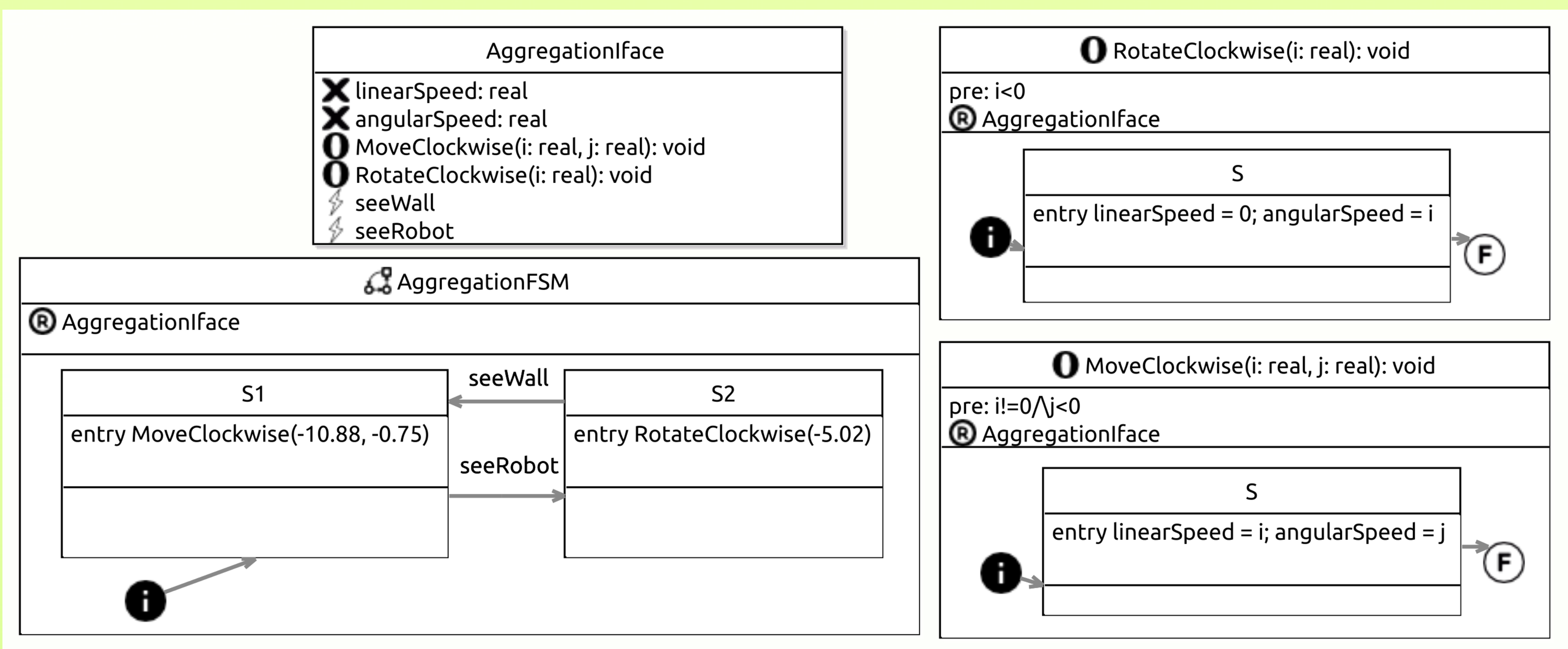


Diagram of the aggregation controller modelled in RoboChart

Elements:

state machine: *AggregationFSM*; interface: *AggregationIface*; state: *S1, S2*; operation: *MoveClockwise, RotateClockwise*

Textual description:

```
stm AggregationFSM {
    requires AggregationIface
    initial I
    state S1 {
        entry MoveClockwise(-10.88, -0.75)
    }
    state S2 {
        entry RotateClockwise(-5.02)
    }
    transition T1 {
        from I to S1
    }
    transition T2 {
        from S2 to S1
        trigger seeWall
    }
    transition T3 {
        from S1 to S2
        trigger seeRobot
    }
}
```

controller

```
operation MoveClockwise(i: real, j:real) : void {
    precondition i != 0 /\ j < 0

    requires AggregationIface

    initial I
    final F
    state S {
        entry linearSpeed = i; angularSpeed = j
    }

    transition T1 {
        from I
        to S
    }

    transition T2 {
        from S
        to F
    }
}
```

operation

initial | after 20s | after 40s | after 60s

Snapshots of the aggregation behaviour of 20 e-puck robots in Enki simulation, using the automatically generated controller code from RoboChart

4. Case study two: Swarm taxis

Behaviour: Robots move towards a beacon while maintaining a coherent group

Controller: three states; one event; two conditions

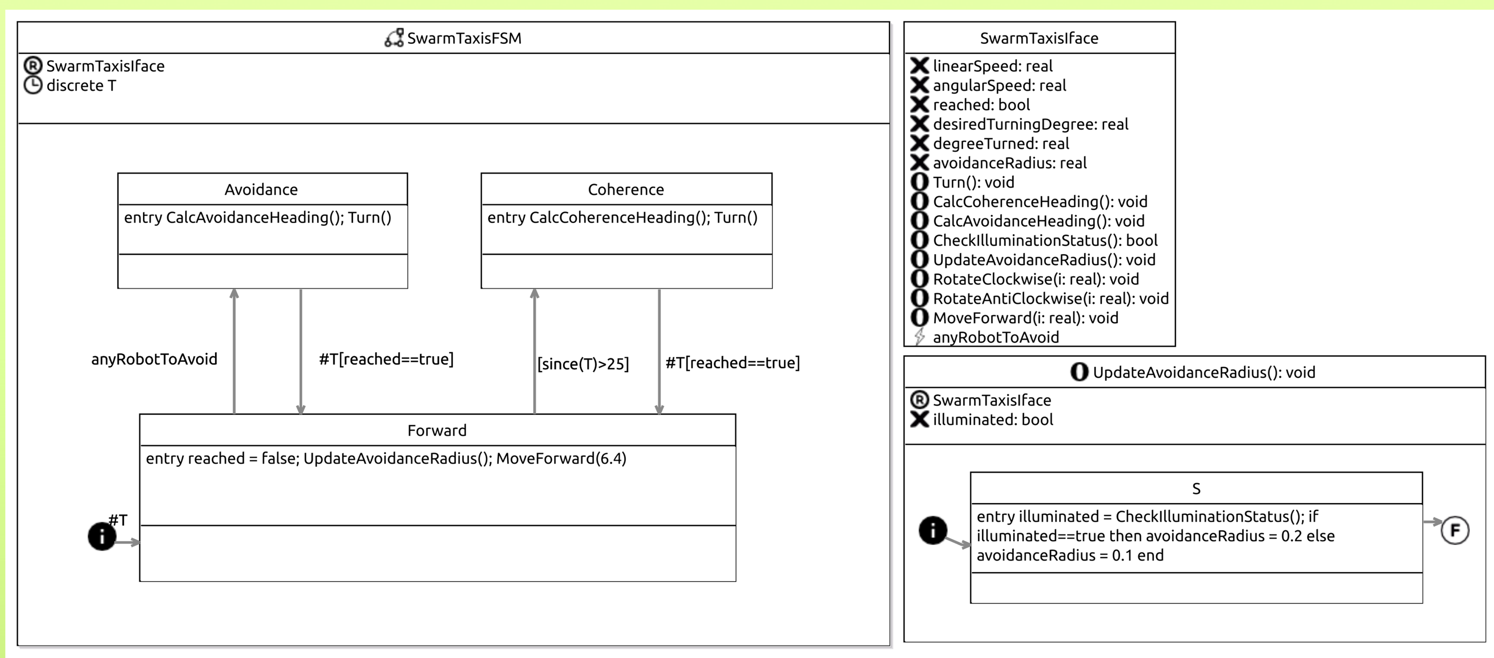
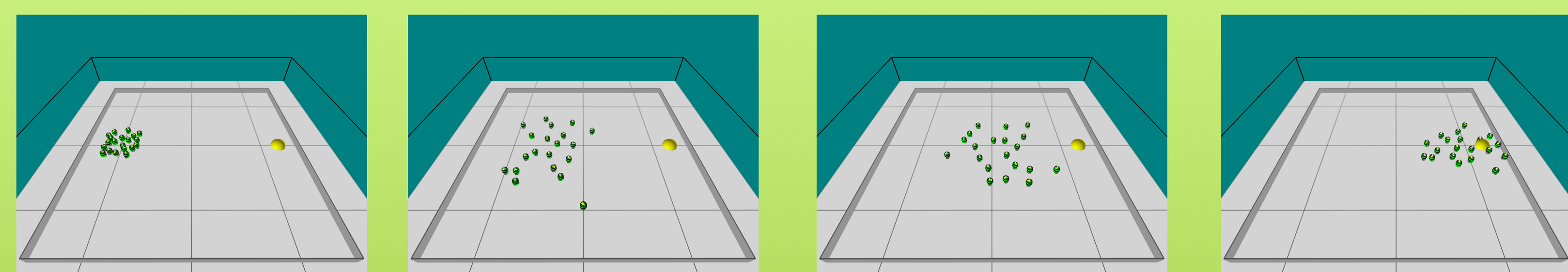


Diagram of the swarm taxis controller modelled in RoboChart



initial | after 120s | after 240s | after 400s

Snapshots of the swarm taxis behaviour of 20 e-puck robots in ARGoS simulation

5. Conclusions and future work

Summary:

- State-machine based robotic controllers can be modelled in RoboChart.
- Controller code can be automatically generated and integrated into different robotic platforms.
- Formal CSP semantics allows for the application of formal verification techniques[1].
- Gap between high-level reasoning and low-level implementation of robotic controller is reduced.

Future work:

- Verify individual controllers as well as their emergent swarm behaviour.
- Model probability and environmental stimuli.
- Generate code for implementation in physical robots.

Reference:

[1] Miyazawa et al. "RoboChart: A state-machine notation for modelling and verification of mobile and autonomous robots". Technical report, University of York.

EPSRC
Engineering and Physical Sciences Research Council