

This is a repository copy of *RoboCert: Property Specification for Robotics using Sequence Diagrams*.

White Rose Research Online URL for this paper:

<https://eprints.whiterose.ac.uk/193109/>

Version: Published Version

Conference or Workshop Item:

Windsor, Matthew orcid.org/0000-0003-1285-0080 and Cavalcanti, Ana Lucia Caneca orcid.org/0000-0002-0831-1976 (2022) RoboCert: Property Specification for Robotics using Sequence Diagrams. In: YorRobots and RoboStar Industry Exhibition, 11-12 Oct 2022, University of York.

Reuse

Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

RoboCert: Property Specification for Robotics using Sequence Diagrams

Why is property specification important?

Robot failures are costly and dangerous! We must ensure that robot software and hardware behaves according to the properties we need:

Universal property

'if the robot finds an obstacle, it will *always* turn around'

Existential property

'it is *possible* that the robot finds an obstacle then turns'

Negated property

'if the robot finds an obstacle, it *will not* go straight on'

Robot properties are often time-sensitive:

- deadlines ('the robot will *immediately* turn around')
- budgets ('the robot *can wait 1s* before turning')



Can't we just use English?

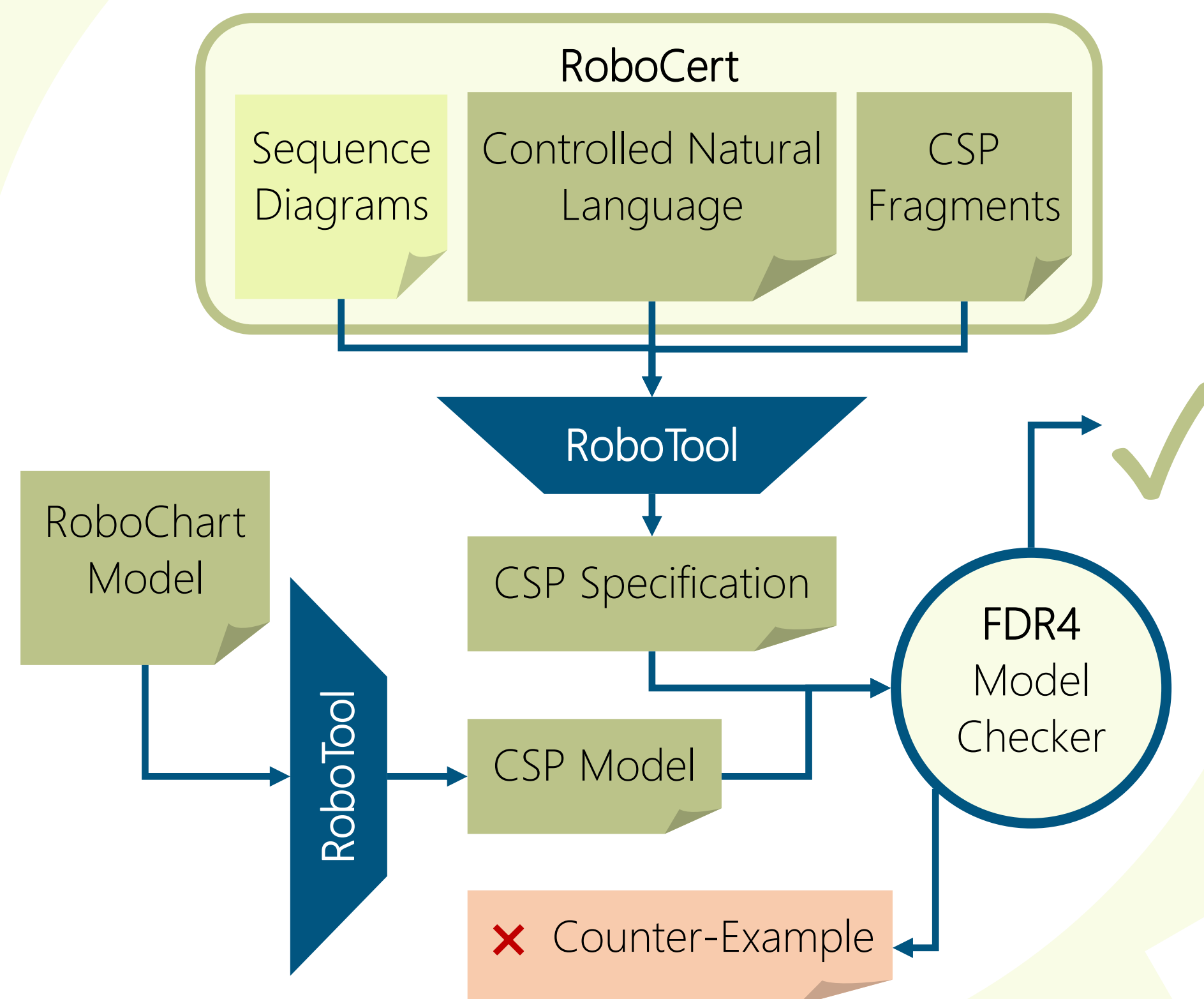
English is **ambiguous!** In the property 'when the robot finds an obstacle, it waits 4s then turns around', can the robot do other things while it waits? Can it turn around in that time?

What about mathematics?

Formal mathematical languages are good for proving properties, but can be **hard to understand** for roboticists.

Our approach

We use **sequence diagrams** (with a formal mathematical meaning) to capture properties in an *unambiguous but understandable* way.



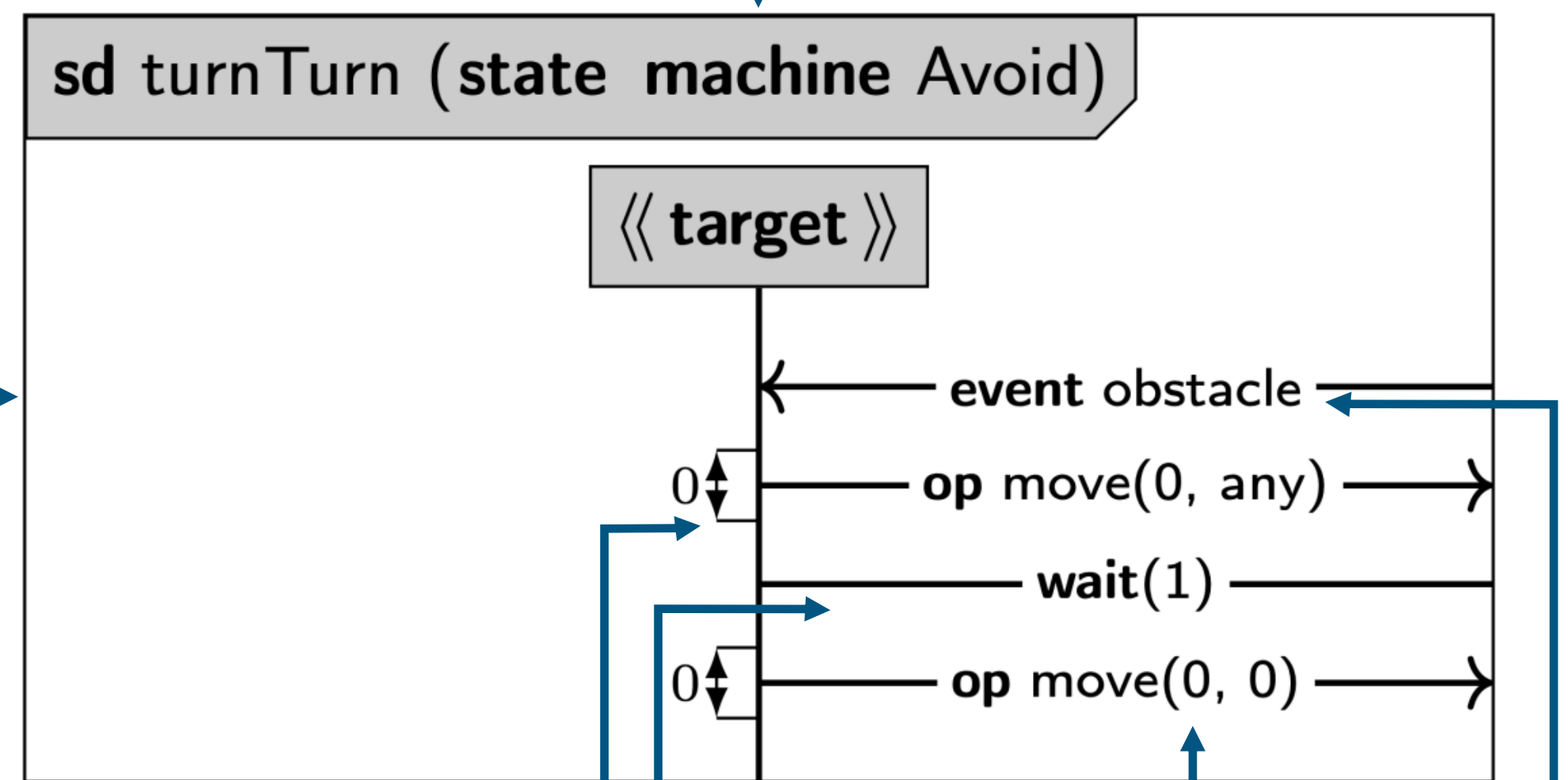
Features

- supports the RoboChart **component model**, with properties over single components or collections of components
- messages over RoboChart **events and operations**
- **deadlines and budgets** using RoboChart's discrete time model
- UML **control flow**: loops, alternative choice, and optional choice
- 'do anything in a set of given messages **until...**' control-flow
- **liveness reasoning** through 'hot/cold' (must occur/may occur) modality on messages, and an 'xalt' mandatory alternative choice
- automated generation of CSP processes for **verification** using the FDR4 model checker

What do our sequence diagrams look like?

Adaptation of UML2 sequence diagrams

Sequences target RoboChart state machines, controllers, modules, collections



Sequence diagrams mention RoboChart events, operations

Express deadlines and budgets in RoboChart's discrete-time model

Verifying Properties with RoboCert

Future directions

Hybrid properties over both software and hardware, dealing with both discrete and continuous time

System and scenario-level properties we can lower to properties over parts of RoboChart/RoboSim models

Graphical tools for viewing and creating sequence diagrams

