Event-Processing in Autonomous Robot Programming

Pouyan Ziafati$^{1,3}$  Holger Voos$^1$  Mehdi Dastani$^3$  John-Jules Meyer$^3$  Leendert van der Torre$^{1,2}$

$^1$SnT, University of Luxembourg

$^2$CSC, University of Luxembourg

$^3$Intelligent Systems Group, Utrecht University

{pouyan.ziafati, holger.voos, leon.vandertorre}@uni.lu

{M.M.Dastani, J.J.C.Meyer}@uu.nl

May 3, 2013
Outline

- Introduction
- Event-processing in autonomous robot programming
- Systematic event-processing support
- Programming language support for implementing
  - event-processing components
  - interaction with event-processing components
- Conclusion
Introduction

Sensory events in autonomous robotics

Control Component

Agent Program

Deliberation Cycle
- Process events
- Update beliefs and goals
- Apply plan generating rules
- Execute plans

Sensory Events

Perception Components

Localization

Object Recognition

Face Recognition

Human Tracker (Gesture, Position)

t1
self(pos5)

obj(orange,pos2)
obj(key,pos1)

obj(milk,pos4)
obj(key,pos3)

t3
face(sara,70%)

t4
usr(sara,pos9)

usr(sara,pos7)
usr(alex,pos8)

t5
self(pos6)

t6
face(sara,80%)
Basic event-processing tasks

- Filtering
- Pattern detection
- Transformation
- Integrating domain knowledge

Usecase examples

- Reliable-faces: faces recognized two times with Conf > 60 within 2sec
- Select red objects and compute their absolute positions
Systematic Event-Processing Support

Current support

- Agent Programming Languages
  - Simple event-handling rules
- Robotics
  - Event-based communications
  - Application specific tools

We need programming language support for

High-level language abstraction

- Ease of programming
- Correct implementation

Efficient implementation

Event-driven and concurrent execution
Event-Processing Framework

Event-processing components and their interactions

- Etalis for programming event-processing components
  - formal semantics - expressive - efficient - knowledge processing
- Supporting adaptive interactions with EPCs
  - Runtime subscription
  - On-demand querying
- Prototype implementation for ROS
**Etalis: Rule-based Event-Processing language**

<table>
<thead>
<tr>
<th>Events: time-stamped ground atoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atomic event ( \text{face}(sara) ) (^{28})</td>
</tr>
<tr>
<td>Complex event ( \text{observed}(sara) ) (^{28,43})</td>
</tr>
</tbody>
</table>

- **Etalis program**: set of rules
  - Patterns of events, and domain knowledge

**E.g. Etalis rule for reliable faces**

\[
\text{reliable\_face}(F) \leftarrow ( \text{face}(F, C1) \ \text{seq} \ \text{face}(F, C2) ).2\text{sec} \\
\text{where}(C1 > 60, C2 > 60)
\]

**E.g. Etalis rule for selecting red objects**

\[
\text{redObj}(\text{AbsolutePos}) \leftarrow \text{obj}(O, OPos) \ \text{equals} \ \text{self}(SPos) \\
\text{where}(\text{color}(O, \text{red}), \ \text{AbsolutePos} = SPo s \times OPos)
\]
Subscribing to event-processing components for certain events

- Types of events, boolean conditions on their contents and their occurrence times

Usecase example

- To Keep looking at Sara: control component subscribes gaze control to position of Sara’s face
Querying Event-Processing Components On-Demand

To maintain certain histories of events in event-processing components to access on-demand

- Buffers: Maintain histories of events of certain types and conditions
- Various policies for managing histories
  - Fixed-buffer: events with occurrence times within a certain time period
  - Count-buffer: last N instances of events of interest
  - Time-buffer: events with occurrence times within last l seconds

**Usecase example**
- Keeping the last location of objects in the environment
- Keeping Sara’s locations in last 24 hours
Conclusion

A software framework to support event-processing and Adaptable Component Interactions in Autonomous Robotics

- Etalis language to implement event-processing components
- Runtime adaptable component interactions
  - Content-based filtering of flow of information
  - Maintaining necessary histories of events to query on-demand
- Interface to ROS
- Demo at http://robolab.gforge.uni.lu/
- Future work: further extension and evaluation of the library
This is the end of presentation

Thank you for your attention :-) 

Questions please?