Towards a graphical language for quadrotor missions

Benjamin Schwartz, Ludwig Nägele, Andreas Angerer, Bruce A. MacDonald
Agenda

- Motivation & Vision
- Requirements
- Concept & Prototype
- Conclusion
Agenda

Motivation & Vision
Requirements
Concept & Prototype
Conclusion
Intuitive specification of quadrotor missions

Current situation

• Hard-coded UAV missions in general-purpose programming languages

• Increasing number of application domains (agriculture, archaeology, ...)

• Graphical end-user solutions with limited functionality
APM Planner

✓ Route defined by waypoints
✓ Commands specificable for each waypoint

✗ No parallel activities
✗ No branching
✗ No complex movements (e.g. obstacle avoidance)
Intuitive specification of quadrotor missions

Current situation

• Hard-coded UAV missions in general-purpose programming languages

• Increasing number of application domains (agriculture, archaeology, ...)

• Graphical end-user solutions with limited functionality

Vision

• Intuitive mission definitions by non-programmers

• Powerful interface for programmers

• Hardware-independent specification of missions (or even simulator-independent)
Agenda

Motivation & Vision

Requirements

Concept & Prototype

Conclusion
Requirements

- **Graphical language** for intuitive mission definition by non-programmers
- **Clear representation** of the main workflow
- **Sufficient expressiveness** needed for common use cases: Branches, loops, parallel actions
- **Concept of extensibility** of the language for programming experts
- **Extensions in turn reusable** by non-programmers
- **Hardware-independent**: Applicable to different target platforms or simulators (programming languages, frameworks, etc.)
Agenda

Motivation & Vision

Requirements

Concept & Prototype

Conclusion
First steps

• Specification of a graphical language for quadrotor missions

• Working editor in eclipse for the language (GMF)

• A code generator for MORSE simulator code
  Only partial support yet (evaluation purpose)
Concept

- **Separation between:**
  - \( \Rightarrow \textbf{Workflow}: \) Description of waypoints, branching (stable)
  - \( \Rightarrow \textbf{Actions}: \) Special behaviour (modifyable, extensible)

- **Workflow elements are defined by meta-model**

- **Action interfaces are defined by meta-model; Action implementations are part of model**
Routing elements

- Basic quadrotor functionality
- Describe the main workflow
- States and arrows

Examples:

TakeOff, FlyTo(GPS), HoldPosition, TouchDown
Concept – language elements

OneTimeActions

- Extended functionality
- Annotated to routing elements
- Executed once

Examples:

Look into direction, Take picture, ...
Concept – language elements

Filter elements and FilterActions

- Influence quadrotor flight commands
- Grouping of multiple FilterActions in Filter elements
- Filter elements are annotated to routing elements
- Running synchronously, internal priorisation

Examples: AvoidCollision, HoldHeight, ...
Concept – language elements

**IfBranch elements and ProcessingActions**

- **IfBranch elements** manipulate workflow (loops are possible)
- Reference to result of previous action, processed by **ProcessingAction**
- Comparison with condition

**Examples:**

Image recognition, …
Parallel elements

- Grouping of OneTimeActions and SimultaneousActions (i.e. time-triggered and constantly running actions)
- Annotatable to multiple routing elements
- May manipulate workflow
Concept – language elements

- **Parallel/Actions**
  - Actions that get triggered during flight (after a specified time)
    - **BatteryCheck**: 10 s
  - Actions that are running for the whole time
    - **StoreSensorData**
    - **ShootInfraredVideo**

- **ComplexFilter**
  - Actions that will alter the flight commands with respect to the old command
    - 1. **AvoidCollision**
    - 2. **HoldHeight**

- **TakeOff**
  - One time actions, executed after reaching GPS waypoint:

- **FlyToGPS**
  - 46.346° / 10.968° 50 m
  - One time actions, executed after reaching GPS waypoint:
    - 1. **Look into direction**
    - 2. **Take picture**

- **IfBranch 2**
  - **BatteryCheck** processed by Compare number
  - **Condition**: < 100 mAh

- **IfBranch 1**
  - **Take picture** processed by Image recognition
  - **Condition**: "Sign found"

- **IfBranch 2**
  - **BatteryCheck** processed by Compare number
  - **Condition**: < 100 mAh

- **FlyHome**
  - **0° / 0°** 50 m
  - One time actions, executed after reaching GPS waypoint:

- **TouchDown**
  - One time actions, executed after reaching GPS waypoint:

---

20.10.2014 Towards a graphical language for quadrotor missions
Extensibility by new actions
**In our scope:** Python-codegenerator for MORSE simulator (quadrotor simulation)
Agenda

- Motivation & Vision
- Requirements
- Concept & Prototype
- Conclusion
Conclusion

• Language and editor for graphically specifying quadrotor missions
• Extensible by new (reusable) actions
• Thus, appropriate for skilled programmers and non-programmers
• First code generator to MORSE simulator code showed proof of concept

Future work:

• Distinction between target-platform independent and dependend semantics (limitations on target platforms: no multi-threading etc.)
• Complete evaluation on (different) real hardware