VisualHFSM 5: recent improvements in programming robots with automata in JdeRobot

Samuel Rey and José M. Cañas

samuel.rey.escudero@gmail.com josemaria.plaza@gmail.com

16 de junio de 2016

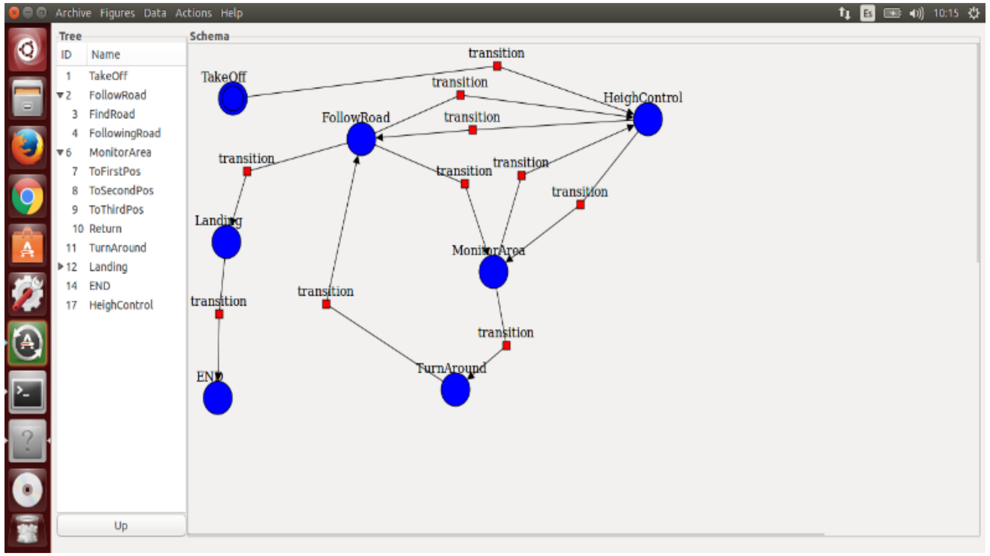
Objectives

The main objective is achieve a more mature, useful and flexible version of VisualHFSM.

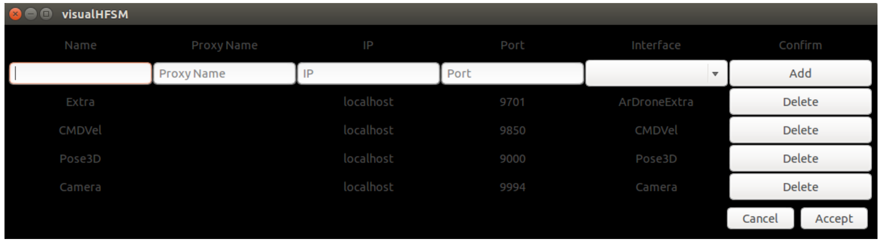
- Improve the usability and functionality of the graphical editor.
- Show a runtime GUI with the automata state.
- Generate Python components.
- Make visualHFSM compatible with the ArDrone.

Graphical Editor's Improvements

- Easier navigation between hierarchy levels.
- Function `shutDown()` for ending the execution.
- More flexibility for creating the ICE configuration file.
- Can be executed from any directory.



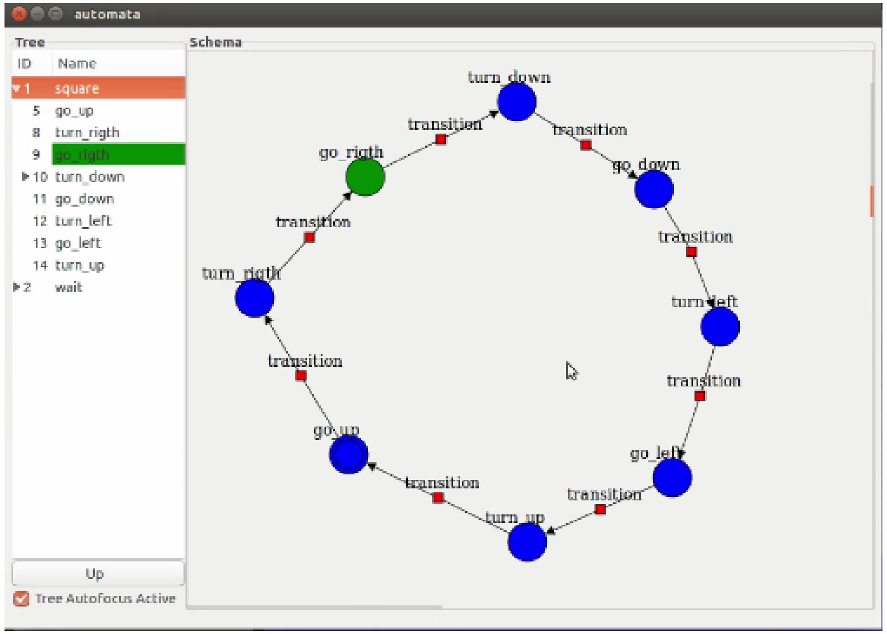
ID	Name
1	TakeOff
▾2	FollowRoad
3	FindRoad
4	FollowingRoad
▾6	MonitorArea
7	ToFirstPos
8	ToSecondPos
9	ToThirdPos
10	Return
11	TurnAround
▸12	Landing
14	END
17	HeighControl



Name	Proxy Name	IP	Port	Interface	Confirm
	Proxy Name	IP	Port		Add
Extra		localhost	9701	ArDroneExtra	Delete
CMDVel		localhost	9850	CMDVel	Delete
Pose3D		localhost	9000	Pose3D	Delete
Camera		localhost	9994	Camera	Delete

Runtime GUI in C++

- Shows dynamically the actives states in execution time.
- Similar appearance to the graphical editor.
- Deactivated by default. Can be activated with the argument *–displaygui=true*.
- Do not depend of the XML file.
- *Autofocus* option.



automata

Tree

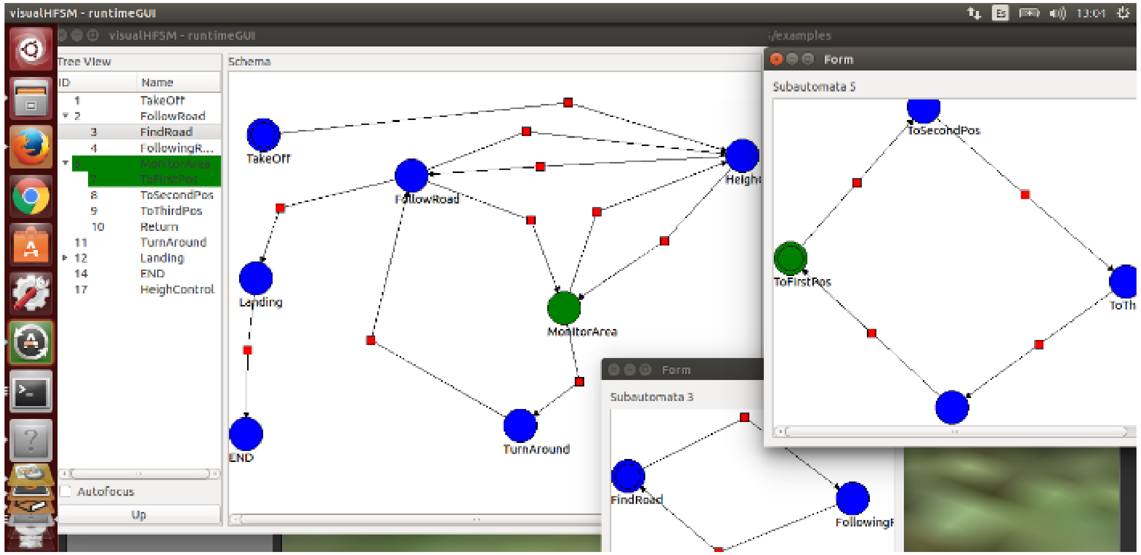
ID	Name
▾ 1	square
5	go_up
8	turn_rigth
9	go_rigth
▸ 10	turn_down
11	go_down
12	turn_left
13	go_left
14	turn_up
▸ 2	wait

Up

☑ Tree Autofocus Active

Python components with runtime GUI

- More flexibility for the tool.
- Template organization a OOP model.
- The runtime works like in the C++ components.
- Allows visualizing more than one level at the same time.



visualHFSM - runtimeGUI

Tree View

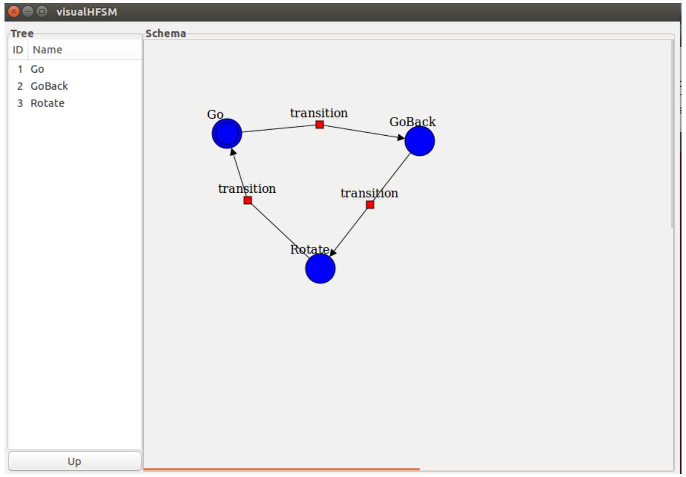
ID	Name
1	TakeOff
2	FollowRoad
3	FindRoad
4	FollowingR...
6	MonitorArea
7	ToFirstPos
8	ToSecondPos
9	ToThirdPos
10	Return
11	TurnAround
12	Landing
14	END
17	HeighControl

□ Autofocus

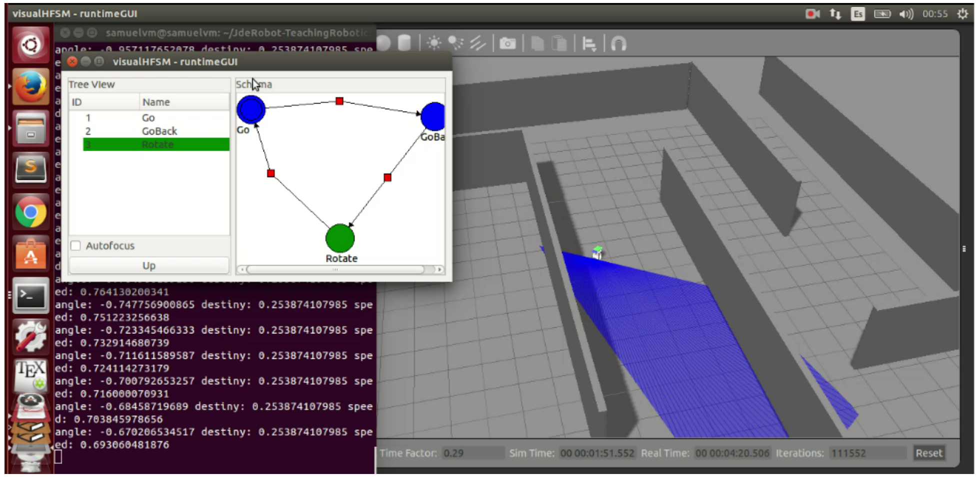
Experiments

Bump & Go

- Simple application with a mono-level automata.



ID	Name
1	Go
2	GoBack
3	Rotate



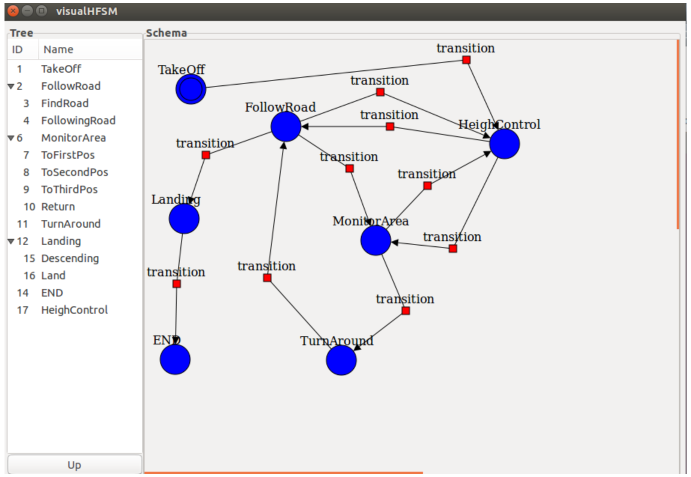
ID	Name
1	Go
2	GoBack
3	Rotate

```
angle: -0.957117652078 destiny: 0.253874107985 spe
ed: 0.764130200341
angle: -0.747756900865 destiny: 0.253874107985 spe
ed: 0.751223256638
angle: -0.723345466333 destiny: 0.253874107985 spe
ed: 0.732914680739
angle: -0.711611589587 destiny: 0.253874107985 spe
ed: 0.724114273179
angle: -0.700792653257 destiny: 0.253874107985 spe
ed: 0.716000070931
angle: -0.68458719689 destiny: 0.253874107985 spee
d: 0.703845978656
angle: -0.670206534517 destiny: 0.253874107985 spe
ed: 0.693060481876
```

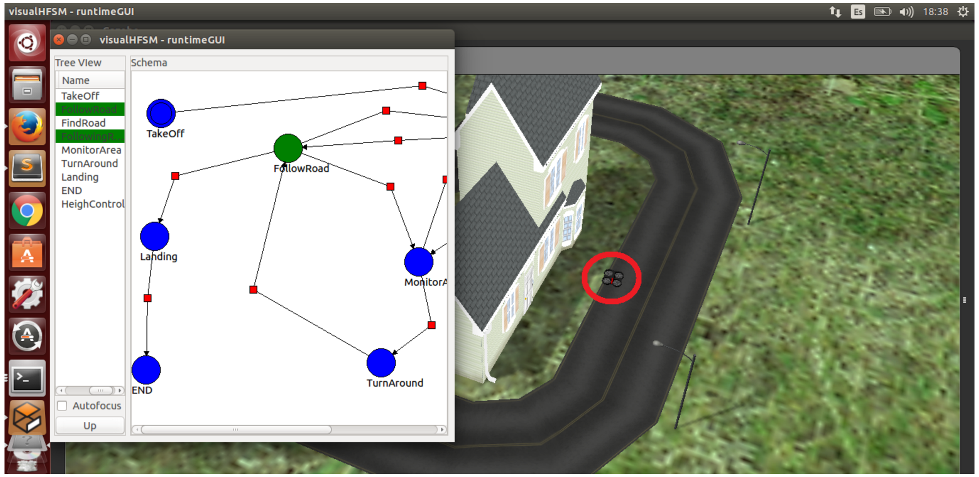
Time Factor: 0.29 Sim Time: 00 00:01:51.552 Real Time: 00 00:04:20.506 Iterations: 111552

A drone monitors an area

- More complex behaviour.
- Multilevel automata.
- Shows the compatibility with the ArDrone's interface in VisualHFSM.



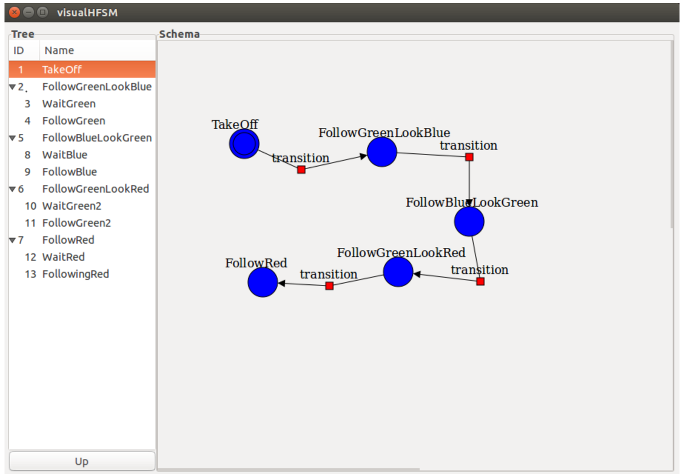
ID	Name
1	TakeOff
▾2	FollowRoad
3	FindRoad
4	FollowingRoad
▾6	MonitorArea
7	ToFirstPos
8	ToSecondPos
9	ToThirdPos
10	Return
11	TurnAround
▾12	Landing
15	Descending
16	Land
14	END
17	HeighControl



Name
TakeOff
FollowRoad
FindRoad
FollowingRoad
MonitorArea
TurnAround
Landing
END
HeighControl

A real drone follows the colors

- Generated components are compatible with real robots.
- Automata vs pure reactive systems.



visualHFSM

Tree

ID	Name
1	TakeOff
▾2	FollowGreenLookBlue
3	WaitGreen
4	FollowGreen
▾5	FollowBlueLookGreen
8	WaitBlue
9	FollowBlue
▾6	FollowGreenLookRed
10	WaitGreen2
11	FollowGreen2
▾7	FollowRed
12	WaitRed
13	FollowingRed