ROS2 and the Crazyflie

Aerial swarms and Autonomy with a tiny flying robot.

Kimberly McGuire - Bitcraze AB
Wolfgang Höning - IMRCLab, TU Berlin
IMRClab (TU Berlin, DE) x Bitcraze AB (Malmö, SE)

- https://imrclab.github.io/
- https://www.bitcraze.io/

Wolfgang Hönig
Intelelligent Multi Robot Coordination lab (TU Berlin)
Crazyflie 2.1

- Quadrotor/copter with brushed motors
- 27 grams, 7 min flight time
- Modular design with expansion decks
- Open source firm/software and open hardware
- Academic and industry researchers
Contributed work with Crazyflie (2019-2020)
ROS role in research

- 28 papers at IROS ‘21, ICRA ‘22 using Crazyflie
- 14 (50%) use Bitcraze’s python library
- 14 (50%) use ROS as communication framework

https://youtu.be/kjCP_hP30Zc

Crazyswarm ROS1

49 Crazyflies flying in total
Going to ROS2

1. ROS1 support will end
2. Much improved communication (TCPROS -> DDS)
3. Breaking change -> possibility to clean up interfaces/features
Autonomy support
- Multiple Positioning systems
- Support Single and Swarm flight
- Easy to connect external ROS2 packages

System design
- Modular Design
- Better ROS2 integration
- Easy to use and well tested

Simulation
- Easily switchable real <-> platform
- Different backends
- Hybrid software in the loop
Overview of the Crazyflie ROS2 architecture

Status: Under development

Crazyflie Server

- crazyflies.yaml
- Parameters
- Topics
- Services

Simulation

Crazyflie ROS2 Python API

Crazyflie Example Scripts

Teleop

URI 'radio://0/80/2M/E7E7E701
URI 'radio://0/80/2M/E7E7E702
URI 'radio://0/80/2M/E7E7E703

URI 'radio://0/80/2M/E7E7E700
Crazyflie Server

- The ROS2 node with Crazyradio PA
  - Crazyradio Dongle PA Long range
  - One Crazyradio can connect to multiple Crazyflies
  - One ROS2 node per Crazyflie not possible yet

- Different backends
  - C++ (Crazyflie-link-cpp)
  - CFlib (Crazyflie-lib-python)

- Handles the:
  - Setting up topics/params
  - Setting up flight services

Sharing 450 packets/s (26 Bytes)
Connecting with the Crazyflie (without ROS)

URI ‘radio://0/80/2M/E7E7E7E7E7

Open link

Request for log/param TOCs

Log and Params TOCS sent

Send Commands

Data streaming

“Fully connected”

Crazyflie Definitions

Logging: Data streaming of variables in real time.

Parameters: Reading and setting of in-firmware variables
CF param handling with ROS2 parameters

Fully connected

Set CF param

CF Param update

TOC and values of CF params received

Translate CF params to ROS2 params

Setup ROS2 param change callback

Update CF param from .yaml

Change ROS2 param CB: CF param is updated

Crazyflie Server

Init

Runtime
CF Log handling with ROS2 topics

**Logblock** (max 26B)
Name: ‘Attitude’
Frequency: 10 Hz
Variables:
- ‘stabilizer.roll’
- ‘stabilizer.pitch’
- ‘stabilizer.yaw’

---

**Fully connected**
- Request log block
- Start a log block
- Log streaming
- Stop a log block

---

**Init**
- TOC CF Logs received
- Setup Services
  - addLogging & removeLogging

**Runtime**
- Set CF log-blocks from .yaml
- Setup logblocks callbacks
  - CB: Transform CF Logs to ROS2 topic
- ROS2 srv
  - /add_logging
    - Sets up new logblock
  - /remove_logging
    - Stops existing logblock
Configure with crazyflies.yaml

---

```yaml
robots:
  cf2:
    enabled: true
    uri: radio://0/20/2M/E7E7E7E702
    initial_position: [0, 0, 0]
    type: cf21 # see robot_types
  cf5:
    enabled: false
    uri: radio://0/80/2M/E7E7E7E705
    initial_position: [0, -0.5, 0]
    type: cf21 # see robot_types

robot_types:
  cf21:
    motion_capture:
      enabled: true
      marker: default_single_marker
      dynamics: default
    battery:
      voltage_warning: 3.8 # V
      voltage_critical: 3.7 # V
  cf21_mocap_deck:
    motion_capture:
      enabled: true
      only if enabled; see motion_capture.yaml
      marker: mocap_deck
      dynamics: default
    battery:
      voltage_warning: 3.8 # V
      voltage_critical: 3.7 # V

all:
  firmware_logging:
    enabled: false
  custom_topics:
    topic_name1:
      frequency: 10 # Hz
      vars: ["stateEstimate.x", "stateEstimate.y", "stateEstimate.z", "pm.vbat"]
  firmware_params:
    commander:
      estimator: 1: complementary, 2: kalman
      controller: 2: PID, 2: mellinger
    stabilizer:
      estimator: 1: complementary, 2: kalman
      controller: 2: PID, 2: mellinger
```

---

Current existing Logs and Params

```
https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/api/logs/
https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/api/params/
```
Autonomy and swarms
Positioning systems

- Different types:
  - MoCap
  - UWB
  - Lighthouse (SteamVR)

- Flowdeck
Default logging: Pose and Transforms

- Default Logging callbacks
  - Specialized topics
  - Predefined types and transforms
- Example: Pose
  - Topic PoseStamped
  - Transform
    - TransformStamped
- RVIZ2
  - Visualization of all crazyflies in the system
Flying with the Crazyflie

- ROS2 Topics:
  - Attitude control
  - Velocity/Position control

- ROS2 Services
  - Takeoff
  - Land
  - Go to
  - Upload/Start trajectory
  - Emergency

- Individual or Swarm
  - /cf1/takeoff
  - /all/takeoff

Take off!
Fly a figure!
Multiple Crazyflies Take off

- 5 Crazyflies
- Lighthouse positioning system
- Python wrapper Crazyflie_py
- Services:
  - /all/takeoff
  - /all/land
Mapping with the Multiranger

- ROS2 course
- Try on a Crazyflie
  - Flow deck: optical flow and range
  - Multi-ranger: 4 horizontal 1D ToF sensors
- Summer Hack project

https://youtu.be/j3qNuV6ieGQ
https://www.manning.com/liveprojectseries/build-mobile-robots-with-ROS2
Mat Sadowski
Connecting with External Packages

- Default topics:
  - Scan: LaserScanStamped
  - Odometry: TF odom > cf_name + Odometry topic
- Connecting SLAM toolbox
  - Scan matching had to be turned off
  - Solely on flow-deck odometry
- Nav2 Bringup Package
- Velocity handling node

https://youtu.be/-NfKnlJMAHQ
https://youtu.be/1BKLPkQ6Gz8
Simulation node?

- Summer hack project
- Crazyflie in Webots 2022b
- Next steps:
  - Integrate webots in Crazyflie ROS2
  - Start implementing other simulations

https://youtu.be/pwSQBwguT-I
What’s next?

- Implementation simulation node with more backends
- Crazyradio 2 development
- Finalizing all documentation
- Tweak out smaller issues
- Getting ready for first release! (Humble)

Interested or want to contribute with the development?

https://imrclab.github.io/crazyswarm2/
ROS2 take aways

- New communication (DDS) is a big plus (stability, consistency, lower latency)
- No central master makes it easier to use
- Update params in real-time
- Excited about MicroROS
Documentation & Contact

Crazyswarm2 Project: https://imrclab.github.io/crazyswarm2/.

Contact

Bitcraze AB: contact@bitcraze.io, www.bitcraze.io

IMRCLab: hoenig@tu-berlin.de, https://imrclab.github.io/