Crazyswarm Documentation

Release 0.3

Wolfgang Hoenig, James A. Preiss

May 19, 2018

Contents

The Crazyswarm projects allows you to fly a swarm of quadcopters (using Bitcraze Crazyflie 2.0 directly or as control boards) in tight, synchronized formations. A motion capture system is required (VICON, OptiTrack, PhaseSpace are supported). We successfully flew 49 Crazyflies using three Crazyradios. An example video for what you can do is shown below:

Our contributed code is licensed under the permissive MIT license, however some of the parts (such as the firmware) are licensed under their respective license.

The Crazyswarm architecture, including some motivation for the design decisions, is described in our paper [pdf]. If you use our work in academic research, please cite us:

```
@inproceedings{crazyswarm,
  author    = {James A. Preiss* and
               Wolfgang  H\"onig* and
               Gaurav S. Sukhatme and
               Nora Ayanian},
  title     = {Crazyswarm: {A} large nano-quadcopter swarm},
  booktitle = {{IEEE} International Conference on Robotics and Automation ({ICRA})},
  pages     = {3299--3304},
  publisher = {{IEEE}},
  year      = {2017},
  url       = {https://doi.org/10.1109/ICRA.2017.7989376},
  doi       = {10.1109/ICRA.2017.7989376},
  note      = {Software available at \url{https://github.com/USC-ACTLab/crazyswarm}},
}
```

Contents:

CHAPTER 1

Changelog

1.1 April 22nd, 2018

1. More features have been merged into the official firmware. The high-level execution (takeoff, landing, trajectory execution) is now part of the official firmware. Custom firmware changes still include:

1. Kalman filter that support full-pose update
2. Improved handling after crashes (particularly important for big quads)
3. Custom LED-ring effect

2. The Crazyswarm now uses the latest official crazyflie_ros instead of its own fork. Crazyswarm specific code (e.g, crazyswarm_server) has been moved into `ros_ws/src/crazyswarm/src`.
3. Getting the code into the official code base unfortunately required some API changes:

1. Support for ellipsoids has been removed. If you need that feature please let us know via github issues.
2. Support for canned trajectories has been removed.
3. The new API allows to upload multiple trajectories and does not have a piece number limit (instead, it is limited by total memory size of all trajectories only).
4. `Hover` has been renamed to `GoTo`

1.2 March 2nd, 2018

1. Support for heterogeneous swarms, i.e., bigger quadrotors can be used with the CF as control board and the bigQuad deck. Types can be specified in `ros_ws/src/crazyswarm/launch/crazyflieTypes.yaml`. The list of all CFs has now a type field in `ros_ws/src/crazyswarm/launch/allCrazyflies.yaml` (previously `all49.yaml`). Each type can have its own marker configuration.

2. Firmware now re-based from the latest official firmware. Some parts are already in the official firmware (e.g., controller). The firmware now supports the use of our Kalman filter (KalmanUSC) or the official Kalman filter, although both have different feature sets. The prebuilt firmware uses our Kalman filter and comes with bigQuad-deck support enabled, i.e., it can be used for both standard Crazyflies and custom ones.
3. Removed support for avoid-target mode

CHAPTER 2

Getting Started

1. Read the Crazyswarm paper to understand the different components of the system. A preprint is available here.
2. Learn more about the Crazyflie platform, e.g. by reading sections 1 and 2 of "Flying Multiple UAVs Using ROS", Springer Book on Robot Operating System (ROS), 2017. A preprint is available here.
3. Learn about the Robot Operating System (ROS), e.g. by reading part 1 of "Programming Robots with ROS. A Practical Introduction to the Robot Operating System" by Morgan Quigley, Brian Gerkey, William D. Smart, O'Reilly, 2015.
4. Get to know your motion capture system (such as VICON or OptiTrack) by reading the respective user manuals.

CHAPTER 3

Installation

We assume that you have Ubuntu 16.04. Avoid using a virtual machine because this adds additional latency and might cause issues with the visualization tools.

3.1 Simulation Only

You can install just the components required for the simulation by doing the following:

```
$ sudo apt install git swig libpython-dev python-numpy python-yaml python-matplotlib
$ git clone https://github.com/USC-ACTLab/crazyswarm.git
$ cd crazyswarm
$ ./buildSimOnly.sh
```

To test the installation, run one of the examples:

```
$ cd ros_ws/src/crazyswarm/scripts
$ python figure8_canned.py --sim
```

More details on the usage can be found in the usage section.

3.2 Simulation and Physical Robots

We assume that you have ROS Kinetic (desktop or desktop-full) installed (instructions).

Install the dependencies and clone the repository:

```
$ sudo apt install git swig libpython-dev python-numpy python-yaml python-matplotlib␣
↪gcc-arm-none-eabi libpcl-dev libusb-1.0-0-dev sdcc
$ git clone https://github.com/USC-ACTLab/crazyswarm.git
$ cd crazyswarm
```

For legal reasons we are not allowed to include the VICON DataStream SDK in this repository, which is used to capture data from a VICON motion capture system. Please download the SDK (version 1.7.1) from http://www.vicon.com and place the following files in *ros_ws/src/crazyflie_ros/externalDependencies/libmotioncapture/externalDependencies/vicon_sdk* (that is, from the Linux64-boost-1.58.0 folder):

```
├── include
│   └── vicon_sdk
│       └── DataStreamClient.h
└── lib64
    ├── libboost_system-mt.so.1.58.0
    ├── libboost_thread-mt.so.1.58.0
    └── libViconDataStreamSDK_CPP.so
```

You can now build everything by running our build script.:

```
$ ./build.sh
```

CHAPTER 4

Usage

4.1 Crazyflie Preparation

Since the Crazyflies are sharing radios and communication channels, they need to have a unique identifier/address. The convention in the Crazyswarm is to use the following address:

```
0xE7E7E7E7<X>
```

where <X> is the number of the Crazyflie in the hexadecimal system. For example cf1 will use address `0xE7E7E7E701` and cf10 uses address `0xE7E7E7E70A`. The easiest way to assign addresses is to use the official Crazyflie Python Client.

1. Label your Crazyflies
2. Assign addresses using the Crazyflie Python Client (use a USB cable for easiest handling)
3. Each radio can control about 15 Crazyflies. If you have more than 15 CFs you will need to assign different channels to the Crazyflies. For example, if you have 49 Crazyflies you'll need three unique channels. It is up to you which channels you assign to which CF, but a good way is to use the Crazyflie number modulo the number of channels. For example, cf1 is assigned to channel 80, cf2 is assigned to channel 90, cf3 is assigned to channel 100, cf4 is assigned to channel 80 and so on.
4. Upgrade the firmwares of your Crazyflies with the provided firmwares (both NRF51 and STM32 firmwares).

- Option 1: Upload the firmware via the command line using `make cload` as described here instead of using Bitcraze graphical app.
- Option 2: Upload the precompiled firmware by executing the following steps:
    1. Plug in a battery
    2. Turn your Crazyflie off by pressing the on/off button
    3. Set your Crazyflie into bootloader mode by holding the on/off button for 3 seconds (The blue M2 and M3 LEDs start to blink)
    4. `rosrun crazyflie_tools flash --target nrf51 --filename prebuilt/cf2_nrf.bin`

5. Turn your Crazyflie off by pressing the on/off button
6. Set your Crazyflie into bootloader mode by holding the on/off button for 3 seconds (The blue M2 and M3 LEDs start to blink)
7. `rosrun crazyflie_tools flash --target stm32 --filename prebuilt/cf2.bin`

1. Upgrade the firmware of you Crazyradios with the provided firmware.

- Option 1: follow the instructions in the `crazyradio-firmware` folder to install the self-compiled version.
- Option 2: Use the prebuilt binary:

1. `python crazyradio-firmware/usbtools/launchBootloader.py`
2. `python crazyradio-firmware/usbtools/nrfbootload.py flash prebuilt/cradio.bin`
3. Now unplug and re-plug the radio. You can check the version using `rosrun crazyflie_tools scan`, which should report `Found Crazyradio with version 99.55`.

Your Crazyflie needs to be rebooted after any change of the channel/address for the changes to take any effect.

4.2 Adjust Configuration Files

There are three major configuration files. First, we have a config file listing all available (but not necessarily active) CFs:

```
# ros_ws/src/crazyswarm/launch/allCrazyflies.yaml
crazyflies:
  - id: 1
    channel: 100
    initialPosition: [1.5, 1.5, 0.0]
    type: default
  - id: 2
    channel: 110
    initialPosition: [1.5, 1.0, 0.0]
    type: medium
```

The file assumes that the address of each CF is set as discussed earlier. The channel can be freely configured. The initial position needs to be known for the frame-by-frame tracking as initial guess. Positions are specified in meters, in the coordinate system of your motion capture device. It is not required that the CFs start exactly at those positions (a few centimeters variation is fine).

The second configuration file defines the possible types:

```
# ros_ws/src/crazyswarm/launch/crazyflieTypes.yaml
crazyflieTypes:
  default:
    bigQuad: False
    batteryVoltageWarning: 3.8  # V
    batteryVoltateCritical: 3.7 # V
    markerConfiguration: 0
    dynamicsConfiguration: 0
    firmwareParams:
      ...
  medium:
    bigQuad: True
```

(continues on next page)

(continued from previous page)

```
    batteryVoltageWarning: 7.6  # V
    batteryVoltateCritical: 7.4 # V
    markerConfiguration: 1
    dynamicsConfiguration: 0
    firmwareParams:
      ...
numMarkerConfigurations: 2
markerConfigurations:
  "0":  # for standard Crazyflie
    numPoints: 4
    offset: [0.0, -0.01, -0.04]
    points:
      "0": [0.0177184,0.0139654,0.0557585]
      "1": [-0.0262914,0.0509139,0.0402475]
      "2": [-0.0328889,-0.02757,0.0390601]
      "3": [0.0431307,-0.0331216,0.0388839]
  "1": # medium frame
    numPoints: 4
    offset: [0.0, 0.0, -0.03]
    points:
      "0": [-0.00896228,-0.000716753,0.0716129]
      "1": [-0.0156318,0.0997402,0.0508162]
      "2": [0.0461693,-0.0881012,0.0380672]
      "3": [-0.0789959,-0.0269793,0.0461144]
numDynamicsConfigurations: 1
dynamicsConfigurations:
  "0":
    maxXVelocity: 2.0
    maxYVelocity: 2.0
    maxZVelocity: 3.0
    maxPitchRate: 20.0
    maxRollRate: 20.0
    maxYawRate: 10.0
    maxRoll: 1.4
    maxPitch: 1.4
    maxFitnessScore: 0.001
```

The third configuration file is the ROS launch file (`ros_ws/src/crazyswarm/launch/hover_swarm.launch`). It contains settings on which motion capture system to use and the marker arrangement on the CFs.

4.2.1 Select Motion Capture System

Below are the relevant settings for the motion capture system:

```
# ros_ws/src/crazyswarm/launch/hover_swarm.launch
# tracking
motion_capture_type: "vicon" # one of vicon,optitrack
object_tracking_type: "libobjecttracker" # one of motionCapture,libobjecttracker
vicon_host_name: "vicon" # only needed if vicon is selected
optitrack_local_ip: "localhost" # only needed if optitrack is selected
optitrack_server_ip: "optitrack" # only needed if optitrack is selected
```

You can choose the motion capture type (currently `vicon` or `optitrack`). The application will connect the the motion capture system using the appropriate SDKs (DataStream SDK and NatNet, respectively). If you select `libobjecttracker` as `object_tracking_type`, the tracking will just use the raw marker cloud from the motion capture system and track the CFs frame-by-frame. If you select `motionCapture` as

`object_tracking_type`, the objects as tracked by the motion capture system will be used. In this case you will need unique marker arrangements and your objects need to be named `cf1`, `cf2`, `cf3`, and so on.

When using `libobjecttracker` it is important to disable tracking of Crazyflies in your motion capture system's control software. Some motion capture systems remove markers from the point cloud when they are matched to an object. Since `libobjecttracker` operates on the raw point cloud, it will not be able to track any Crazyflies that have already been "taken" by the motion capture system.

Vicon

Vicon is fully supported and tested with Tracker 3.4.

OptiTrack

Warning:

The OptiTrack support currently has the following limitations:

- It is incompatible with Motive 2.0 because it uses NatNet 3.0.1 which has a different bit-steam syntax. Use an older version (Motive 1.10.3 is known to work).

Use the following settings for correct operation:

- Un-tick the rigid body in Motive so that the point cloud is streamed.
- Advanced network settings. Up axis: Z
- When specifying the marker locations in the config file you need to use the coordinates in Rviz and not Motive.

Instruction on how to use the rigid body option with Optitrack are available here.

4.2.2 Configure Marker Arrangement

If you select the `libobjecttracker` as `motion_capture_type`, you will need to provide the marker arrangement of your markers. All CFs must use the same marker configuration. An example marker configuration using four markers is shown below:

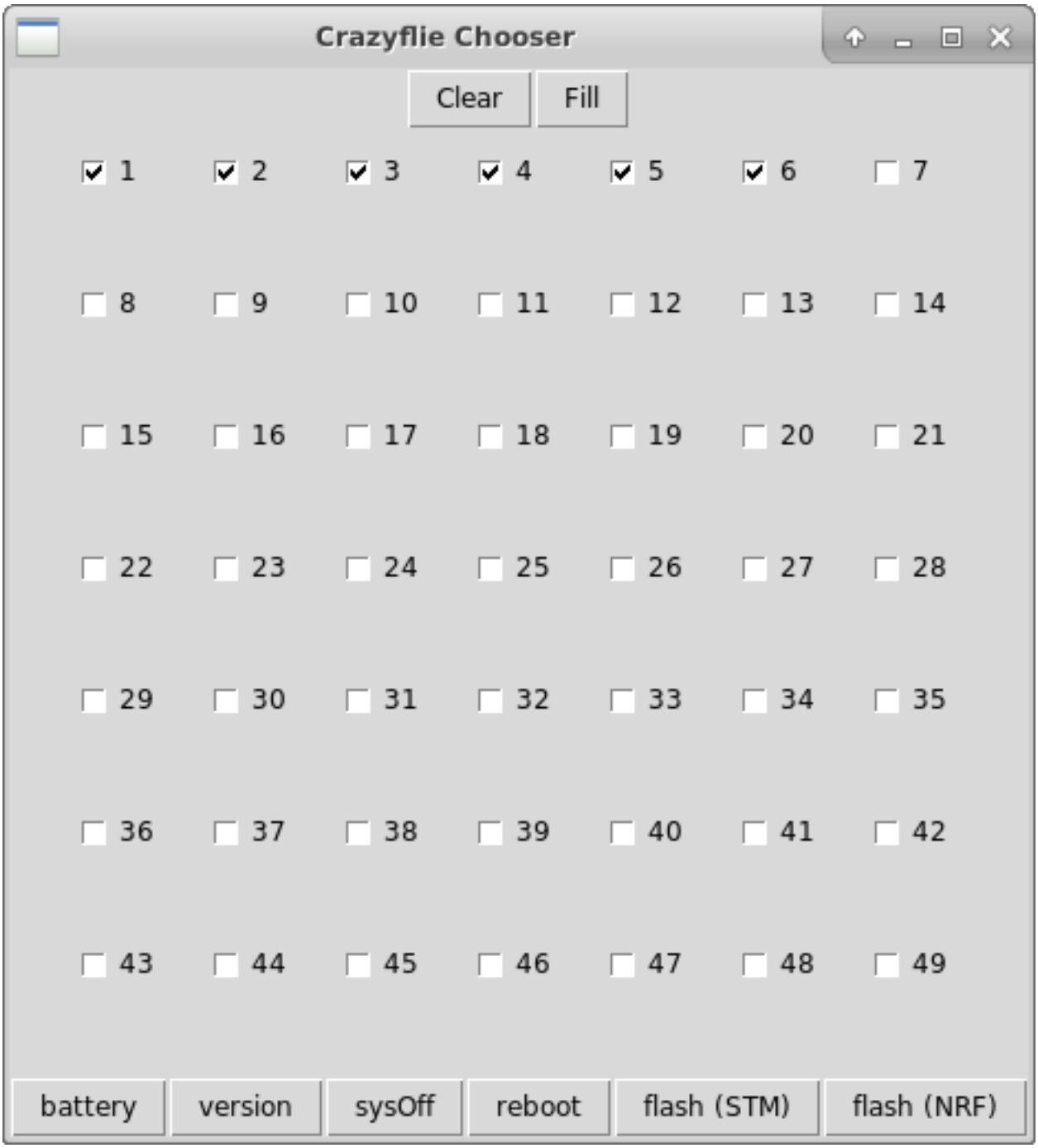
1. Place one CF with the desired arrangement at the origin of your motion capture space. The front of the Crazyflie should point in the `x` direction of the motion capture coordinate system.
2. Find the coordinates of the used markers, for example by using `roslaunch crazyswarm mocap_helper.launch`.
3. Update `crazyflieTypes.yaml`, see the example above.

4.3 Monitor Swarm

A simple GUI is available to enable/disable a subset of the CFs, check the battery voltage, reboot and more. The tool reads the `ros_ws/src/crazyswarm/launch/all49.yaml` file. You can execute it using:

```
ros_ws/src/crazyswarm/scripts
python chooser.py
```

An example screenshot is given below:



Clear Disables all CFs

Fill Enables all CFs

battery Retrieves battery voltage for enabled CFs. Only works if `crazyflie_server` is not running at the same time. Can be used while the CF is in power-safe mode.

version Retrieves STM32 firmware version of enabled CFs. Only works if `crazyflie_server` is not running at the same time. Can only be used if CF is fully powered on.

sysOff Puts enabled CFs in power-safe mode (NRF51 powered, but STM32 turned off). Only works if `crazyflie_server` is not running at the same time.

reboot Reboot enabled CFs (such that NRF51 and STM32 will be powered). Only works if `crazyflie_server` is not running at the same time.

flash (STM) Flashes STM32 firmware to enabled CFs. Only works if `crazyflie_server` is not running at the same time. Assumes that firmware is built.

flash (NRF) Flashes NRF51 firmware to enabled CFs. Only works if `crazyflie_server` is not running at the same time. Assumes that firmware is built.

4.4 Basic Flight

In order to fly the CFs, the `crazyflie_server` needs to be running. Execute it using:

```
source ros_ws/devel/setup.bash
roslaunch crazyswarm hover_swarm.launch
```

It should only take a few seconds to connect to the CFs. If you have the LED ring extension installed, you can see the connectivity by the color (green=good connectivity; red=bad connectivity). Furthermore, `rviz` will show the estimated pose of all CFs. If there is an error (such as a faulty configuration or a turned-off Crazyflie) an error message will be shown and the application exits. If there is a problem in the communication between the motion capture system and the Crazyswarm server, the application will not exit but the positions of the Crazyflies will not appear in rviz.

If you have an XBox360 joystick attached to your computer. You can issue a take-off command by pressing "Start" and a landing command by pressing "Back". All CFs should take-off/land in a synchronized fashion, holding the x/y position they were originally placed in.

4.5 Advanced Flight

The flight can be controlled by a python script. A few examples are in `ros_ws/src/crazyswarm/scripts/`.

1. Test the script in simulation first:

```
python figure8_csv.py --sim
```

(If you are asked to press a button, use the right shoulder on your joystick or press enter on the keyboard.)

1. Run the `crazyflie_server` (in another terminal window):

```
source ros_ws/devel/setup.bash
roslaunch crazyswarm hover_swarm.launch
```

2. Once the connection is successful, execute the script without `--sim`:

```
python figure8_csv.py
```

CHAPTER 5

Python API

You can use the Python API like this:

```
from pycrazyswarm import *
swarm = Crazyswarm()
timeHelper = swarm.timeHelper
allcfs = swarm.allcfs
```

The `swarm` object gives you access to a time helper (that works in both real execution and simulation) and the CrazyflieServer (`allcfs`).

The CrazyflieServer object allows you to

- Send broadcasts to a group of Crazyflies
- Access individual Crazyflies

5.1 Broadcasts

- **`allcfs.emergency(self)`** Cut power to motors of all CFs
- **`allcfs.takeoff(self, targetHeight, duration, groupMask = 0)`** Take-off of given Crazyflie group to specified hight within the specified duration.
- **`allcfs.land(self, targetHeight, duration, groupMask = 0)`** Land given Crazyflie group to specified hight within the specified duration.
- **`allcfs.stop(self, groupMask = 0)`** Stops (i.e., turns off motors) for given Crazyflie group.
- **`goTo(self, goal, yaw, duration, groupMask = 0)`** Moves each Crazyflie relative to its current position/yaw by the specified goal/yaw offset and reaches that location after the specified duration.
- **`startTrajectory(self, trajectoryId, timescale = 1.0, reverse = False, relative = True,`** Starts executing the specified trajectory. Trajectory can be scaled in time (larger number = slower), or executed in reverse.

5.2 Access Crazyflies

- `cf = allcfs.crazyflies[idx]` Array that contains all Crazyflies
- `cf = allcfs.crazyfliesById["42"]` Dictionary that contains crazyflies by their ids

5.3 Individual Crazyflie

- **`cf.id`** ID of the crazyflie
- **`cf.initialPosition`** Initial position (as defined in `crazyflies.yaml`)
- **`cf.setGroupMask(self, groupMask)`** Assign this Crazyflies to be part of the given groups (all CFs are part of group 0)
- `cf.takeoff(self, targetHeight, duration, groupMask = 0)`
- `cf.land(self, targetHeight, duration, groupMask = 0)`
- `cf.stop(self, groupMask = 0)`
- **`cf.goTo(self, goal, yaw, duration, relative = False, groupMask = 0)`** Move to the specified goal (position) and yaw angle in the specified time.
- `cf.uploadTrajectory(self, trajectoryId, pieceOffset, trajectory)`
- `cf.startTrajectory(self, trajectoryId, timescale = 1.0, reverse = False, relative = True, groupMask = 0)`
- **`cf.position(self)`** Returns the current position of the Crazyflie
- `cf.getParam(self, name)`
- `cf.setParam(self, name, value)`
- `cf.setParams(self, params)`

5.4 Examples

5.4.1 Nice Hover

niceHover.py:

```
import numpy as np
from pycrazyswarm import *

swarm = Crazyswarm()
timeHelper = swarm.timeHelper
allcfs = swarm.allcfs

# takeoff
allcfs.takeoff(targetHeight=1.0, duration=2.0)

# move to the initially assigned positions facing forward
timeHelper.sleep(2.0)
for cf in allcfs.crazyflies:
    pos = np.array(cf.initialPosition) + np.array([0, 0, 1.0])
```

(continues on next page)

(continued from previous page)

```
    cf.goTo(pos, 0, 1.0)

# Wait 5 seconds
timeHelper.sleep(5)

# Land
allcfs.land(targetHeight=0.02, duration=2.0)
timeHelper.sleep(2.0)
```

CHAPTER 6

Hardware

If you are planning to built a Crazyswarm yourself, below is the list of components we use.

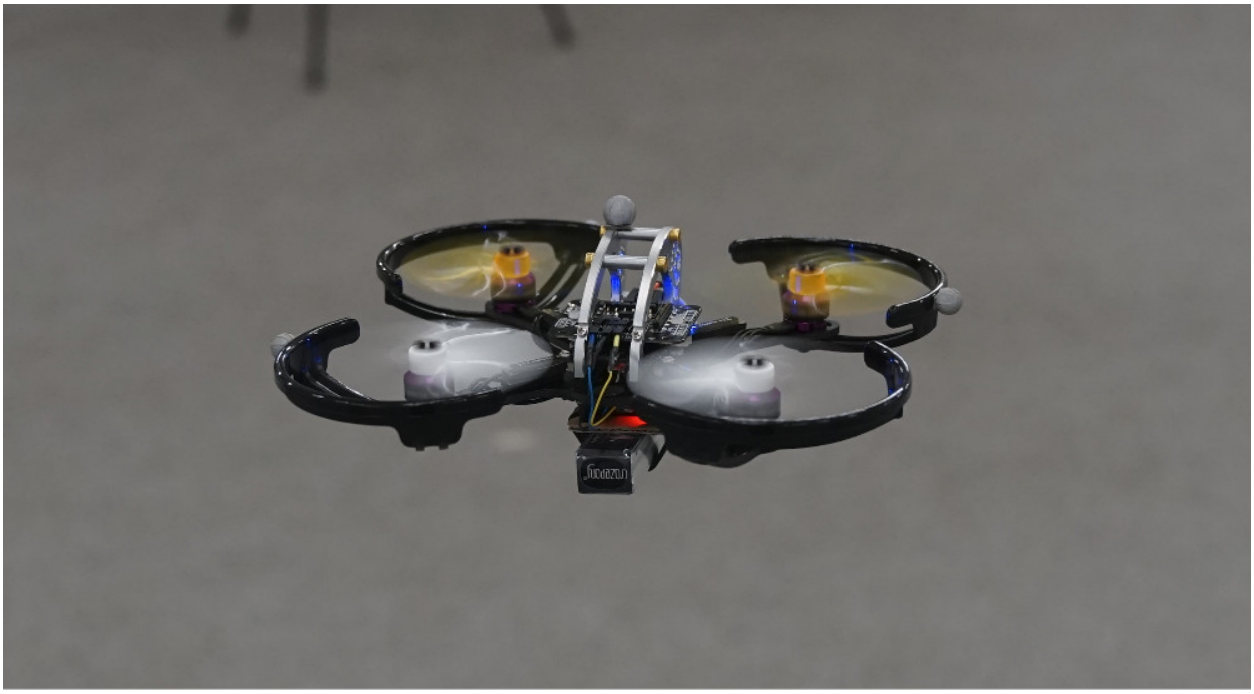
6.1 Components

Prices and links as of summer 2017.

Table 1: Part List

Name	Price (USD)	Notes	Distributor	Link
Bitcraze Crazyflie 2.0	180		Bitcraze/Seeed	https://store.bitcraze.io/collections/kits/products/crazyflie-2-0
Crazyradio PA	30	1 suffices for 15 Crazyflies. Also available as kit with Crazyflie.	Bitcraze/Seeed	https://store.bitcraze.io/products/crazyradio-pa
LED-ring deck	20	Optional for light effects	Bitcraze/Seeed	https://store.bitcraze.io/collections/decks/products/led-ring-deck
Propeller pack	5	Breaks most often; buy ~ 0.5 packs per CF	Bitcraze/Seeed	https://store.bitcraze.io/collections/spare-parts-crazyflie-2-0/products/propeller-pack
Spare motor mounts	5	Breaks second most often; buy ~0.5 packs per CF	Bitcraze/Seeed	https://store.bitcraze.io/collections/spare-parts-crazyflie-2-0/products/crazyflie-2-0-4-x-spare-7-mm-motor-mounts
Turnigy nano-tech 260mAh 1S	2.45	Buy 2 or 3 per CF	Hobbyking	https://hobbyking.com/en_us/turnigy-nano-tech-260mah-1s-35-70c-lipo-pack-qr-ladybird-genius.html?___store=en_us
Turnigy Micro-6 Lipoly Battery Charger	13.34	Buy 1 for 6 Cfs	Hobbyking	https://hobbyking.com/en_us/turnigy-micro-6-lipoly-battery-charger.html?___store=en_us
7.9mm reflective markers	4	Buy 4 per CF (and a few spares)	B&L Engineering	http://www.bleng.com/7-9mm-reflective-markers-set-of-10.html
Double-sided foam tape (e.g. 3M VHB)	10	Command Poster Strips work well, too		
Debug adapter	30	1 suffices for swarm; only needed if firmware development is intended	Bitcraze/Seeed	https://store.bitcraze.io/collections/accessories/products/debug-adapter
J-LINK EDU	60	1 suffices for swarm; only needed if firmware development is intended	Segger	https://shop-us.segger.com/J_Link_EDU_p/8.08.90.htm

6.2 Medium Quadrotor



Prices and links as of March 2018.

Table 2: Part List

Part	Item	Weight (g)	Price (USD)	Notes	Distributor	Link
Flight Controller	Bitcraze Crazyflie 2.0		180		Bitcraze/Seeed	https://store.bitcraze.io/collections/kits/products/crazyflie-2-0
Flight Controller	Bitcraze BigQuad Deck	4	7		Bitcraze/Seeed	https://store.bitcraze.io/collections/decks/products/bigquad-deck
Frame + Prop Guards	Crazypony KingKong Flyegg 130mm	30	22.99		Amazon	https://www.amazon.com/gp/product/B073LBNJCR/ref=oh_aui_detailpage_o06_s02?ie=UTF8&psc=1
Power + ESC combo	DYS F18A 4-in-1 ESC	8	35.99		Amazon	https://www.amazon.com/gp/product/B077G26CTX/ref=oh_aui_detailpage_o06_s01?ie=UTF8&psc=1
Motors x4	Crazypony 1104 7500KV x4	26	52.99		Amazon	https://www.amazon.com/gp/product/B06XNJ8XPP/ref=oh_aui_detailpage_o06_s01?ie=UTF8&psc=1
Propellers	Crazypony 2840 3-blade x8	5.5	7.99		Amazon	https://www.amazon.com/gp/product/B071FN9V78/ref=oh_aui_detailpage_o06_s00?ie=UTF8&psc=1
Battery	Crazypony 450mAh 2S x2	29	18.99		Amazon	https://www.amazon.com/gp/product/B07351J77T/ref=oh_aui_detailpage_o02_s00?ie=UTF8&psc=1

6.3 Large Quadrotor

Prices and links as of March 2018.

Table 3: Part List

Part	Item	Weight (g)	Price (USD)	Notes	Distributor	Link
Flight Controller	Bitcraze Crazyflie 2.0		180		Bitcraze/Seeed	https://store.bitcraze.io/collections/kits/products/crazyflie-2-0
Flight Controller	Bitcraze BigQuad Deck	4	7		Bitcraze/Seeed	https://store.bitcraze.io/collections/decks/products/bigquad-deck
Frame	iFlight RACER iX5 V2 210mm	24.2	36.99		Amazon	https://www.amazon.com/gp/product/B06XYRPQCD/ref=oh_aui_detailpage_o05_s01?ie=UTF8&psc=1
Power Distribution	Thriverline Matek PDB & Dual BEC	13	10		Amazon	https://www.amazon.com/gp/product/B06XB9K9S8/ref=oh_aui_detailpage_o05_s02?ie=UTF8&psc=1
ESCs x4	Makerfire BLHeli 20A x4	25	34.99		Amazon	https://www.amazon.com/Makerfire-BLHeli-Brushless-Controller-QAV250/dp/B07869QR66/ref=sr_1_1
Motors	DLFPV 2205 2300KV x4	115	39.99		Amazon	https://www.amazon.com/gp/product/B01JM1C9GK/ref=od_aui_detailpages02?ie=UTF8&psc=1
Propellers	RAYCORP 5040 3-Blade x8	6.2	14.99		Amazon	https://www.amazon.com/gp/product/B01N3R7RCI/ref=oh_aui_detailpage_o05_s02?ie=UTF8&psc=1
Battery	Tattu 1800mAh 45C 3	200	19.99		Amazon	https://www.amazon.com/gp/product/B013I9SAHO/ref=oh_aui_detailpage_o05_s00?ie=UTF8&psc=1
Propeller Guards	5-inch Prop guard for 250mm Class	18	4.5	Optional	Hobbyking	https://hobbyking.com/en_us/propeller-guards-for-the-250-class-racer-5inch-set-of-4.html
Alternative Prop Guards	KingKong 5 Inch Propeller Guard		4.9	Optional	Helipal	http://www.helipal.com/kingkong-5-inch-propeller-guard-black.html
SD Card Camera	EOVAS 1080P Small Cam	40	25	Optional	Amazon	https://www.amazon.com/gp/product/B074QM3JDP/ref=oh_aui_detailpage_o04_s00?ie=UTF8&psc=1

CHAPTER 7

Indices and tables

- genindex
- modindex
- search