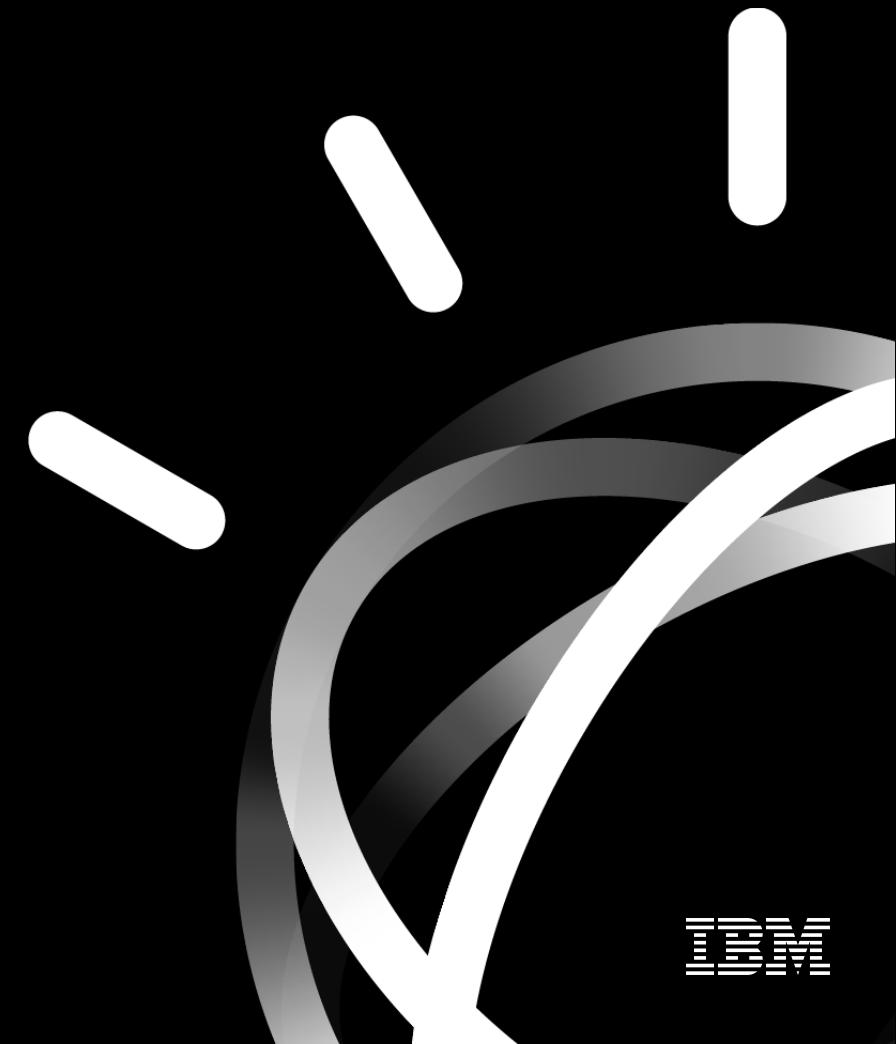


Drone Programming

Aerial Image Object Detection with Watson Visual Recognition

John Walicki
IBM Developer Advocate
CTO IoT / Edge Advocacy
@johnwalicki



Agenda

- Tello Intro
 - Drone Programming – Python / Node.js / Node-RED / Go / Swift
- Watson Studio – Drone Image Analysis
 - Training a Watson Visual Recognition Custom Object Detection Model
 - Save Lives!
- DroneAID - Object Detection with Tello Camera
 - Demo!
- Control your Tello Drone using Node-RED
 - Demo!
- Program your Drone with Python
 - Demo / Hands On !

<http://ibm.biz/TorontoDroneDrop>

Tello Programming

Python
Node.js
Node-RED
Go
Swift



Drone Aerial Image Detection with Watson Studio Visual Recognition

The screenshot shows the IBM Watson Studio interface. At the top, the navigation bar includes 'IBM Watson Studio', a bell icon, 'John Walicki's Account', and a user profile icon ('JW'). Below the bar, the breadcrumb navigation shows 'Projects / Flooding / Detect Flooded Neighborhoods'. To the right of the breadcrumb are several icons: a help icon, a refresh icon, a gear icon, a grid icon, and a map icon.

Detect Flooded Neighborhoods

Associated Service : [watson-vision-combined-dsx](#)

[Edit and Retrain](#)

The main content area has three tabs: 'Overview', 'Test' (which is selected), and 'Implementation'. On the left, there is a 'Filter' section with a 'Threshold' slider set at 0.0 and a 'Class' section containing a checkbox for 'Flooded_Neighborhood' which is checked. To the right is a large aerial photograph of a flooded neighborhood with houses and trees submerged in water. Below the image, a callout box indicates a detection result: 'Flooded_Neighborhood' with a confidence score of '0.92'.

Get the Code:

<https://github.com/IBM/drones-iot-visual-recognition>

<https://developer.ibm.com/tutorials/use-drones-after-floods-to-help-survivors-watson-visual-recognition/>

Detect Survivors on Rooftops

- Use drone aerial images, Watson Studio and Watson Visual Recognition to survey flood damaged neighborhoods, identify homes with survivors on rooftops and detect rescue boats
- A lifesaving scenario might be to use drones to identify flood survivors and direct rescue boats to the victims using GPS and visual recognition.
- Train a model to detect survivors waving their arms
 - Find drone images of people waving their arms for help
 - Zip the images (>10) / Upload to Cloud Object Store
 - Create a new class and retrain your model
- Train another Watson Visual Recognition Model to detect Wildfire Burned Neighborhoods using drone images
 - <https://developer.ibm.com/tutorials/detect-wildfire-damaged-homes-using-drone-images-watson-visual-recognition/>

rescueboat.jpg



grande.jpg



tokyoflood.jpg



Rescue Boat	0.62
Rooftop Survivors	0.57
Flooded Neighborhood	0.11

Flooded Neighborhood	0.89
Rescue Boat	0.02
Rooftop Survivors	0.00

Rooftop Survivors	0.91
Rescue Boat	0.00
Flooded Neighborhood	0.00

- Use drone aerial images, Watson Studio and Watson Visual Recognition to survey flood damaged neighborhoods, identify homes with survivors on rooftops and detect rescue boats

IBM Watson

[Log In](#)[Try it for Free](#)

Explore our apps

Use IBM Watson to collaborate and build smarter applications. Quickly visualize and discover insights from your data and collaborate across teams.



IBM Watson Studio

Democratize ML/DL to accelerate infusion of AI in your business.

[Learn more](#)

IBM Watson Knowledge Catalog

Securely discover, catalog, and govern enterprise data.

[Learn more](#)

Flooding

Associated Service : watson-vision-combined-dsx

[Edit and Retrain](#)[Overview](#) [Test](#) [Implementation](#)

Filter

Threshold

Class

- Flooded Neighborhood
- Rescue Boat
- Rooftop Survivors

[Clear results](#)

1_s.jpg



Rescue Boat	0.89
Rooftop Survivors	0.06
Flooded Neighborhood	0.02

rescueboat.jpg



Rescue Boat	0.62
Rooftop Survivors	0.57
Flooded Neighborhood	0.11

grande.jpg



Flooded Neighborhood	0.89
Rescue Boat	0.02
Rooftop Survivors	0.00

screen-shot-2016-04-20-at-105252.jpg



Flooded Neighborhood	0.89
Rescue Boat	0.08
Rooftop Survivors	0.00

tokyoflood.jpg



Rooftop Survivors	0.91
Rescue Boat	0.00
Flooded Neighborhood	0.00

sjm-floodalert-0226-04.jpg



Rescue Boat	0.83
Flooded Neighborhood	0.30
Rooftop Survivors	0.00

Homework - Learning Objectives

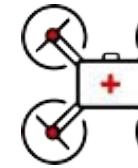
After completing this workshop you will be able to:

- Create a Visual Recognition model in Watson Studio running in IBM Cloud
- Capture images from a drone and zip them into a class (provided)
- Train a model to identify objects in the images
- Score the identified objects

<http://ibm.biz/TorontoDroneDrop>



Pedro Cruz
IBM Developer Advocate
Founder of DroneAID



DroneAid





Pedro lives here





No Power



No Cell Signal



No Food or Water

S.O.S
AGUA
COMIDA



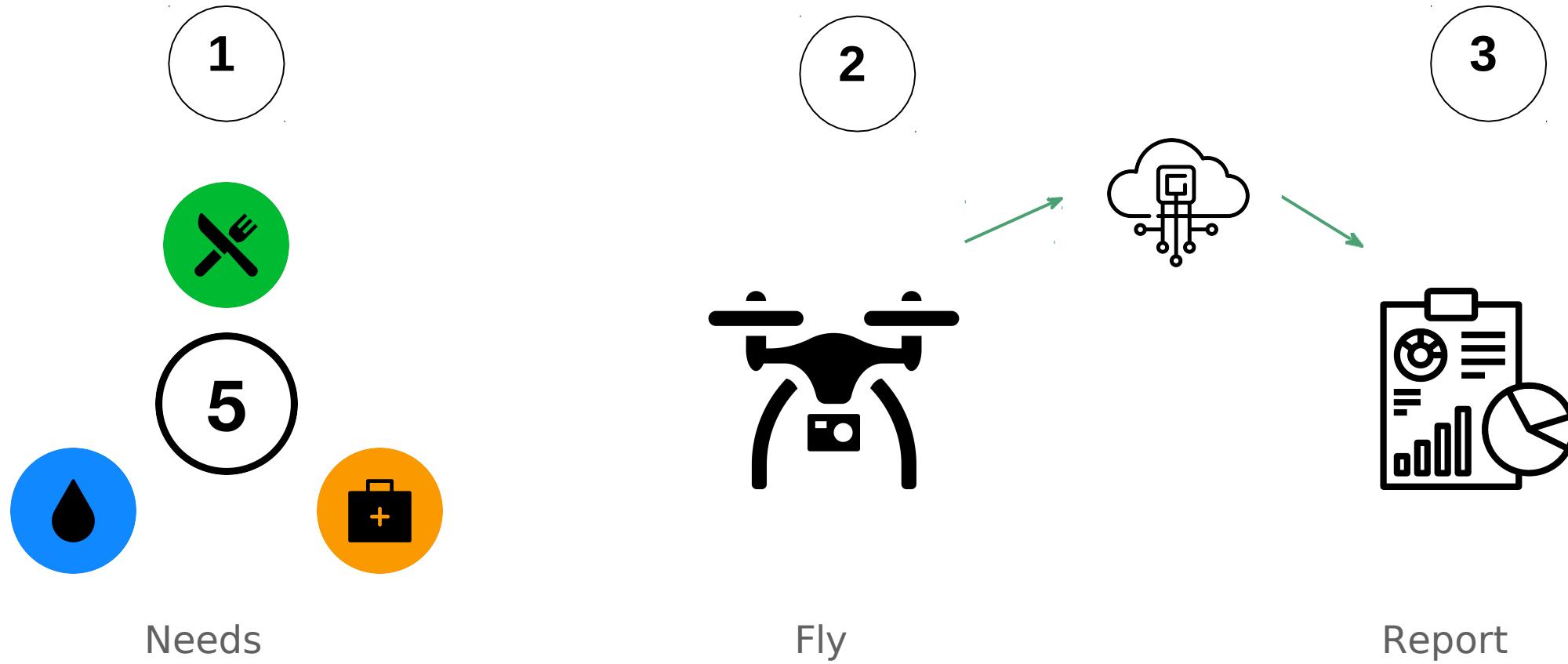
#MARIANA 2^{do}
SE LEVANTA



Aerial Scout for First Responders

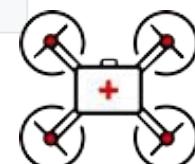


How it works



United Nations Office for the Coordination of Humanitarian Affairs

Symbol	Meaning	Symbol	Meaning
	Immediate Help Needed		Shelter Needed
	No Help Needed		First Aid Kit Needed
	Water Needed		Area with Children in Need
	Food Needed		Area with Elderly in Need



DroneAid

Map

Satellite



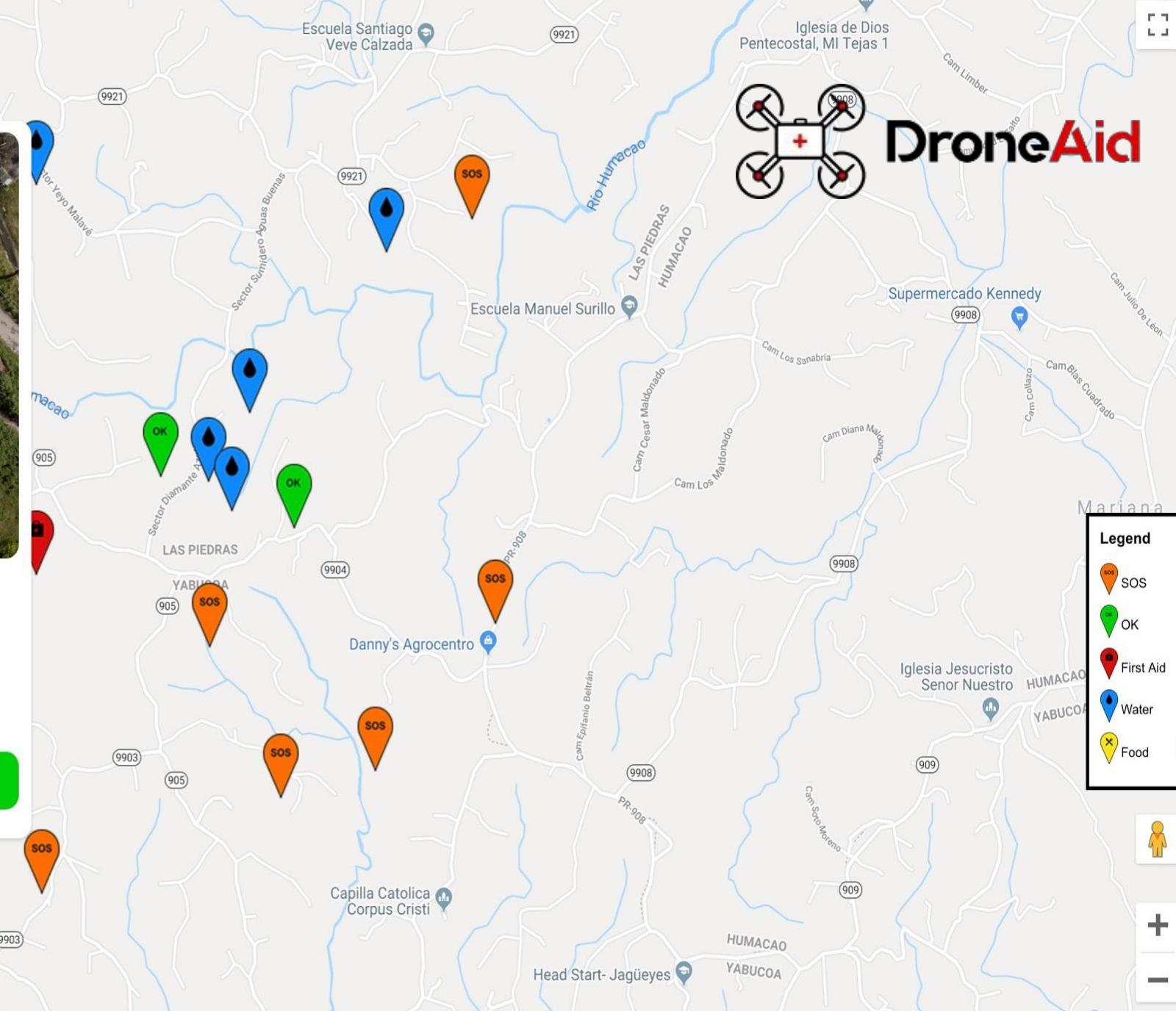
Detected: "SOS", "AGUA", "COMIDA"

GPS Coordinates: [18.129938, -65.858775 \(Humacao, PR\)](#)

Timestamp: 11:24 AM, October 20, 2017

Cancel

Confirm



Prediction

 On

Confidence

 0.4Stop stream

children	0
elderly	0
firstaid	1
food	0
ok	0
shelter	0
sos	3
water	0

Get the Code:

<https://developer.ibm.com/exchanges/models/all/max-object-detector/>

The screenshot shows the IBM Developer website with a dark theme. The top navigation bar includes links for IBM Developer, Topics, Products & Services, Community, Open source at IBM, and a search icon. On the left, a sidebar under 'Artificial intelligence' lists categories like Articles, Courses, Models, Code Patterns, Open Projects, Series, Tutorials, and Videos. Below that are sections for Community, Blog Posts, and Events. A 'Related topics' section lists Conversation, Data science, Deep learning, Machine learning, and Speech & empathy. The main content area features a large blue button labeled 'Get this model' with a GitHub icon, followed by links to 'Try the API', 'Try the web app', and 'Try in a Node-RED flow'. The central part of the page is titled 'Object Detector' and describes it as a 'Model | Deployable, Trainable' that 'Localize and identify multiple objects in a single image.' It was created by 'IBM Developer Staff' on September 21, 2018, and published on March 20, 2018. An 'Overview' section details the model's function, mentioning it recognizes objects from 80 classes using the COCO Dataset and SSD Mobilenet V1 object detection model for TensorFlow. The right side of the page includes social sharing icons for Facebook, Twitter, LinkedIn, and others, along with sections for Technologies (Artificial intelligence, Deep learning, Visual recognition), Products & Services (Model Technologies), and Model Technologies.

IBM Developer Topics Products & Services Community Open source at IBM Q

Artificial intelligence

Model | Deployable, Trainable

Object Detector

Localize and identify multiple objects in a single image.

By IBM Developer Staff
Updated September 21, 2018 | Published March 20, 2018

Overview

This model recognizes the objects present in an image from the 80 different high-level classes of objects in the [COCO Dataset](#). The model consists of a deep convolutional net base model for image feature extraction, together with additional convolutional layers specialized for the task of object detection, that was trained on the COCO data set. The input to the model is an image, and the output is a list of estimated class probabilities for the objects detected in the image. The model is based on the [SSD Mobilenet V1 object detection model for TensorFlow](#).

Get this model →

Try the API →

Try the web app →

Try in a Node-RED flow →

f t in m o

Technologies (3) ▾

Artificial intelligence Deep learning

Visual recognition

Products & Services (3) ▾

Model Technologies (2) ▾

20

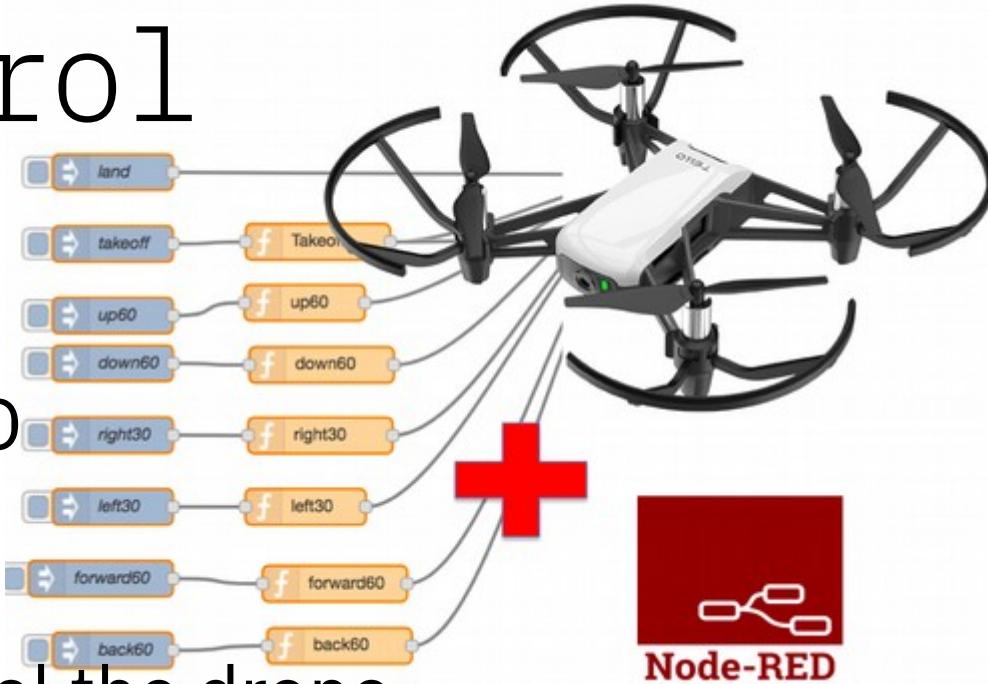
Get the Code:

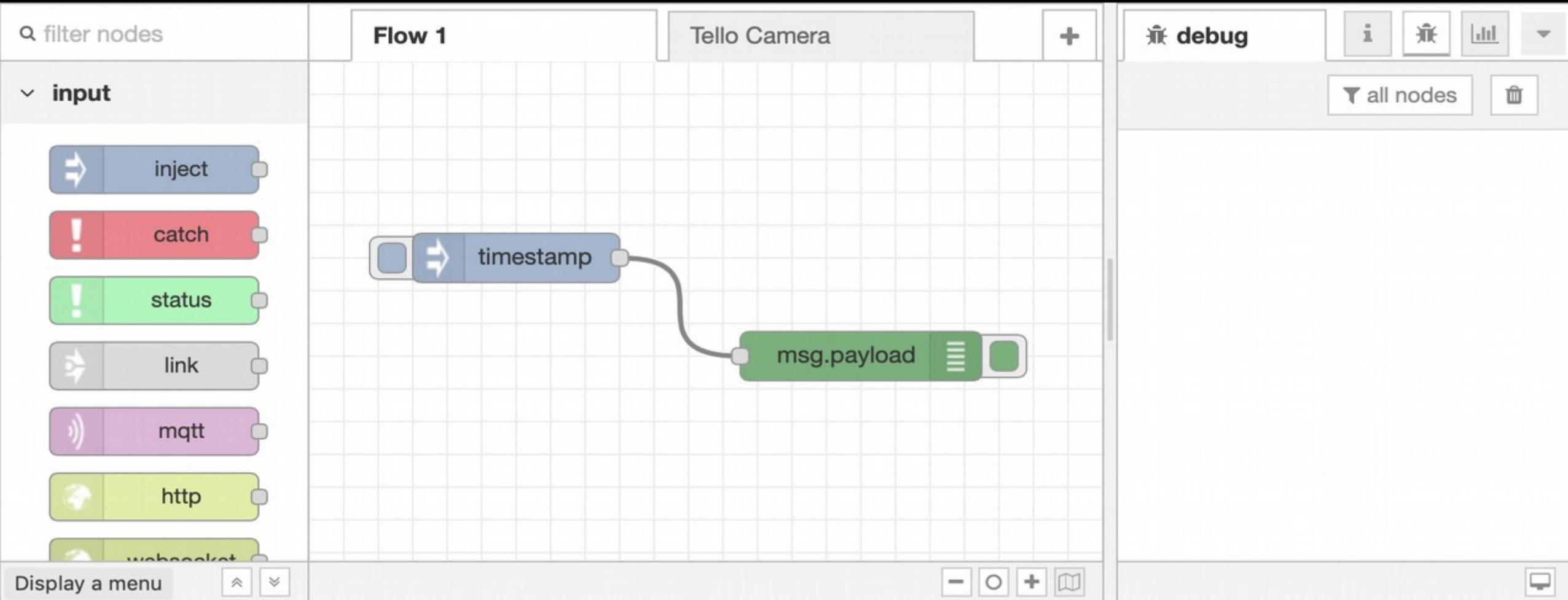
<https://github.com/Code-and-Response/DroneAID>



Node-RED Tello Control

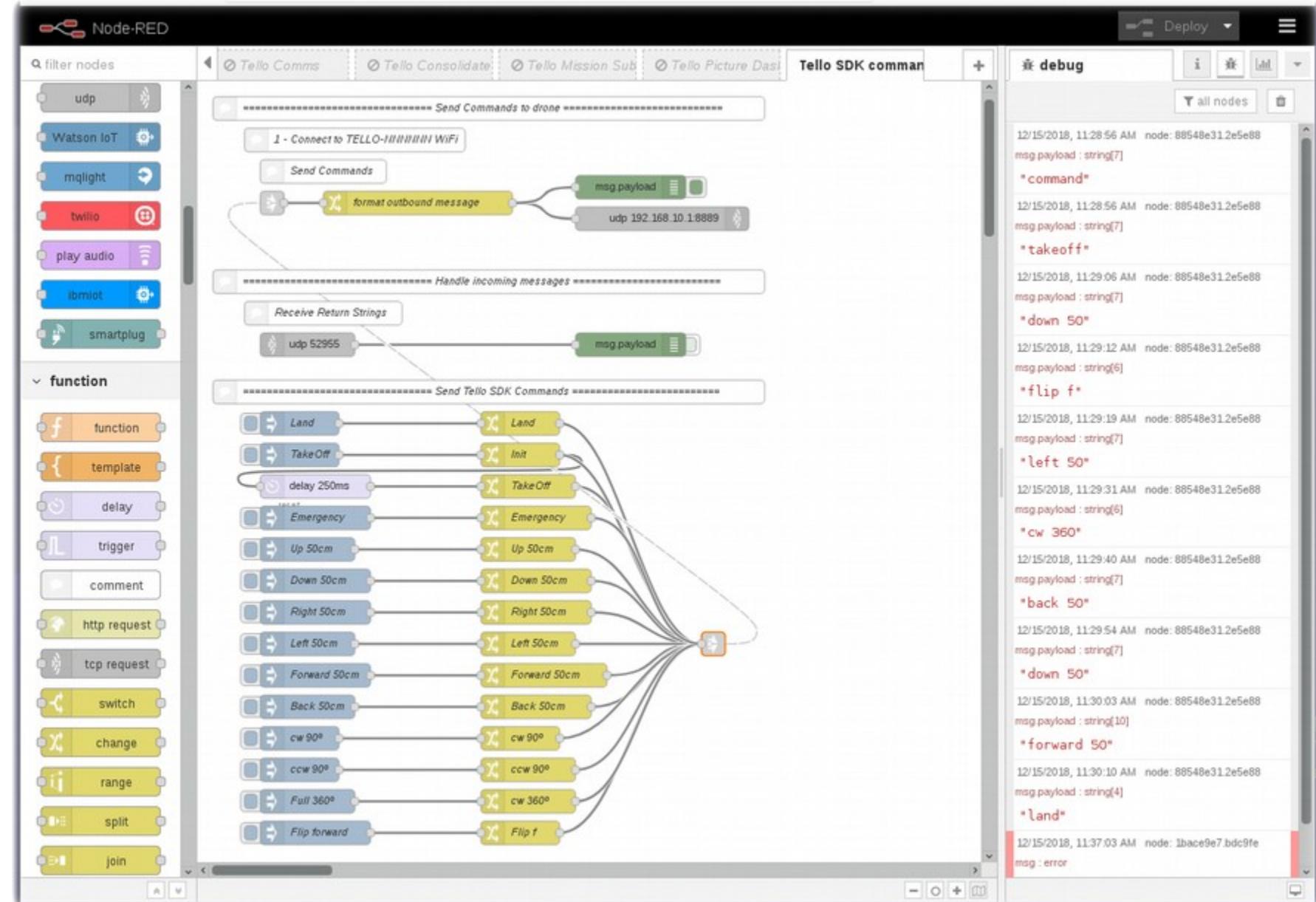
- Control your Tello Drone using Node-RED
 - Learn about Node-RED
 - Send SDK commands to the drone
 - Build a Node-RED Dashboard to control the drone
 - Receive telemetry data from the drone
 - Create missions for your drone
 - Take pictures using the drone camera
 - Watson Visual Recognition and Drone image analysis
 - Train a Visual Recognition Custom Classifier for drone images





Node-RED is an open source project and flow-based programming environment from the JS Foundation. It provides a palette of nodes that allow users to very quickly wire up IoT applications that can combine streams of both physical and digital events. The Node.JS runtime is easy to install on both devices and the cloud, and provides a framework for extending its capabilities.

Send Commands to the Drone using Node-RED



Tello Control Center Node-RED Dashboard

The screenshot displays two main components: the Node-RED editor on the left and the Tello Control Dashboard on the right.

Node-RED Editor: This section shows a complex flow for controlling a Tello drone. It includes nodes for sending commands to the drone (e.g., "Send Commands to drone", "Send Commands", "format outbound message", "msg payload", "udp 192.168.30.1:8889"), receiving return strings ("Receive Return String", "msg payload", "udp 52955"), and handling Tello SDK commands. The flow involves numerous nodes for various drone operations like takeoff, landing, and movement (Up, Down, Left, Right, Forward, Backward, Rotations, Flips).

Tello Control Dashboard: This section shows the user interface for controlling the Tello drone. It features a sidebar with categories such as "Control", "Tello Camera", "Results", and "Tello Control Dashboard". Under "Control", there is a large list of buttons for drone operations, each with an icon and text description. The buttons include:

- EMERGENCY STOP
- TAKEOFF
- LAND
- UP (50CM)
- DOWN (50CM)
- RIGHT (50CM)
- LEFT (50CM)
- FORWARD (50CM)
- BACKWARD (50CM)
- C CLOCKWISE (90°)
- ANTI-CLOCKWISE (90°)
- FULL ROTATION (360°)
- FLIP (FORWARD)

Tello Telemetry Node-RED Dashboard

The screenshot displays a Node-RED dashboard titled "Tello Telemetry". On the left, the Node-RED interface shows a flowchart for "Send Commands to drone" and "Handle incoming messages". The "Send Commands to drone" section includes nodes for "Connect to TELLO-MINIWIIFI WiFi", "Send Command", "Format outbound message" (using "msg.payload" and "udp 192.168.10.19999"), and "Parse Status into an object" (using "msg.payload" and "tcp 8890"). The "Handle incoming messages" section includes nodes for "Receive Return String", "Open port 8890 in Firewall", and "Parse Status into an object" (using "msg.payload" and "tcp 8890"). The "Send Tello SDK Commands" section contains numerous nodes for various drone commands like Connect, Land, Takeoff, Emergency Stop, and various movement commands (Up, Down, Left, Right, Forward, Backward, Clockwise, Counter Clockwise, Full Rotation, Flip). These commands are mapped to specific Tello Telemetry metrics on the right side of the dashboard.

Tello Telemetry

Stats

- Battery: 39
- Height: 1,000 (Line chart from 203.54 to 203.708)
- Temp: 66 °C
- Barometer: 39.79 mbars

Accel

- Accel X: 9 cm/sec²
- Accel Y: -5 cm/sec²
- Accel Z: -994 cm/sec²

Movement

- Speed X: 0 cm/sec
- Speed Y: 0 cm/sec
- Speed Z: 0 cm/sec

Control

- EMERGENCY STOP
- CONNECT
- TAKEOFF
- LAND
- UP (50CM)
- DOWN (50CM)
- RIGHT (50CM)
- LEFT (50CM)
- FORWARD (50CM)
- BACKWARD (50CM)
- CLOCKWISE (90°)
- ANTI-CLOCKWISE (90°)
- FULL ROTATION (360°)
- FLIP (FORWARD)

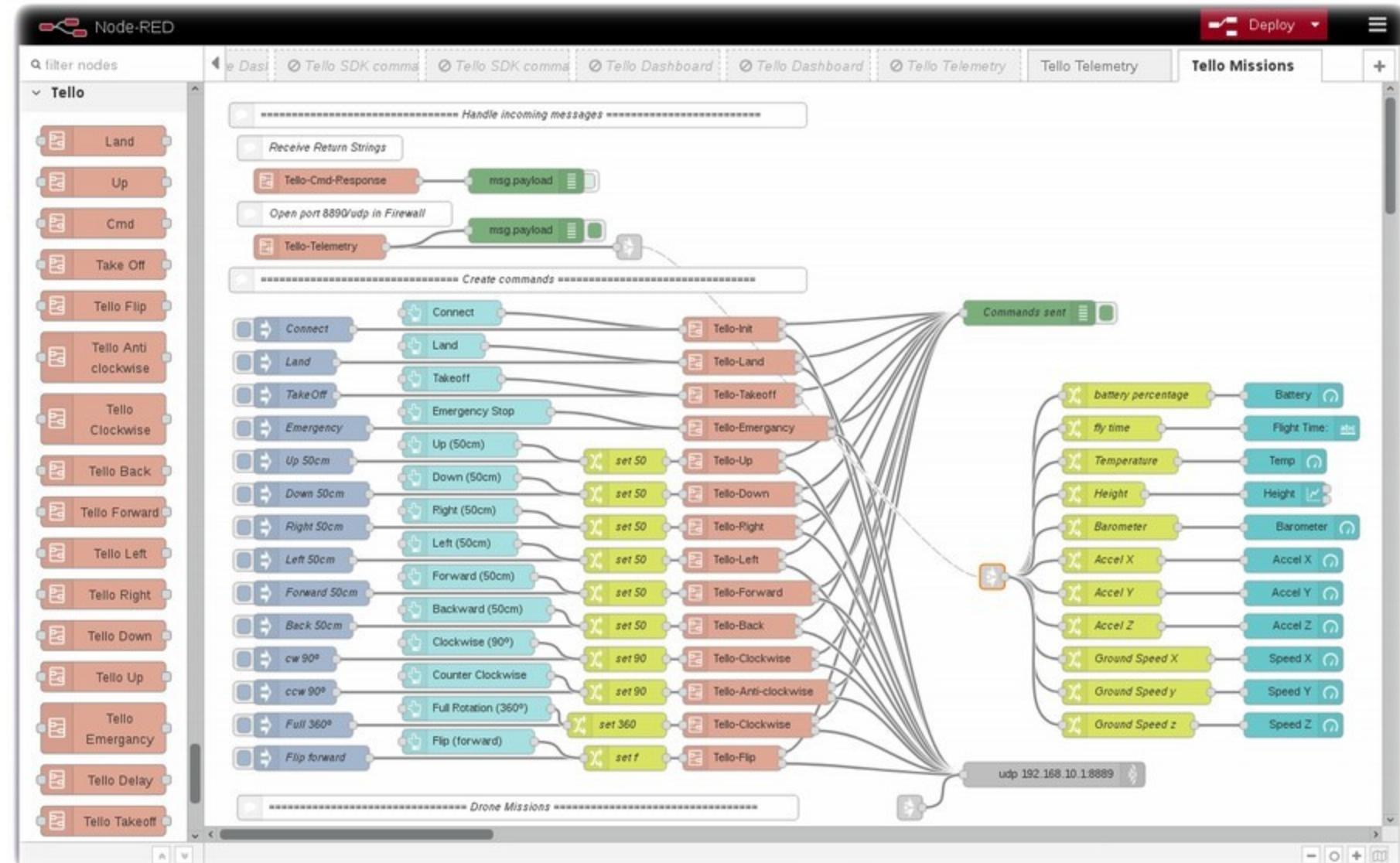
Tello Telemetry

- Battery
- Flight Time: 22
- Temp
- Barometer
- Height
- Speed X
- Speed Y
- Speed Z

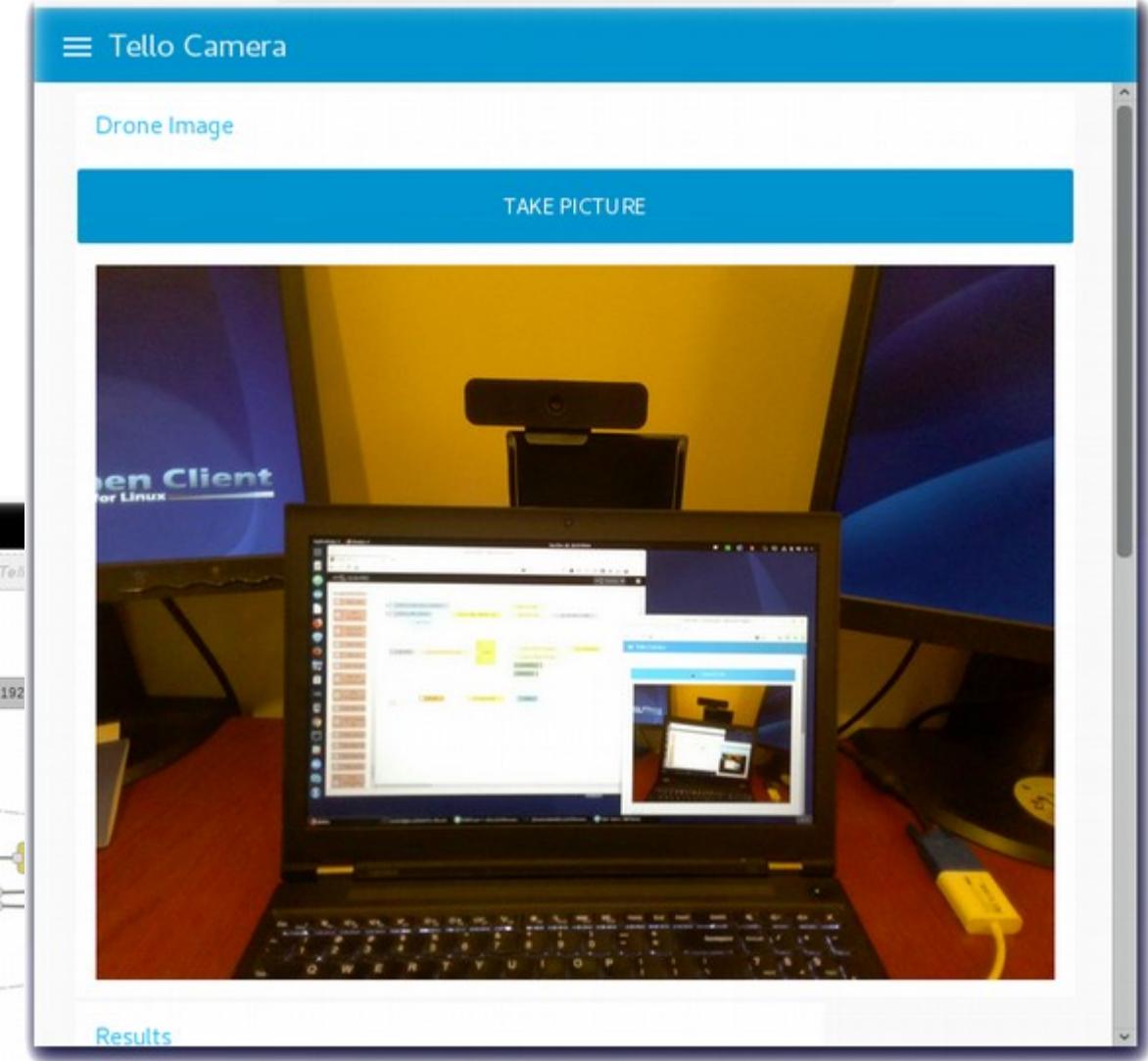
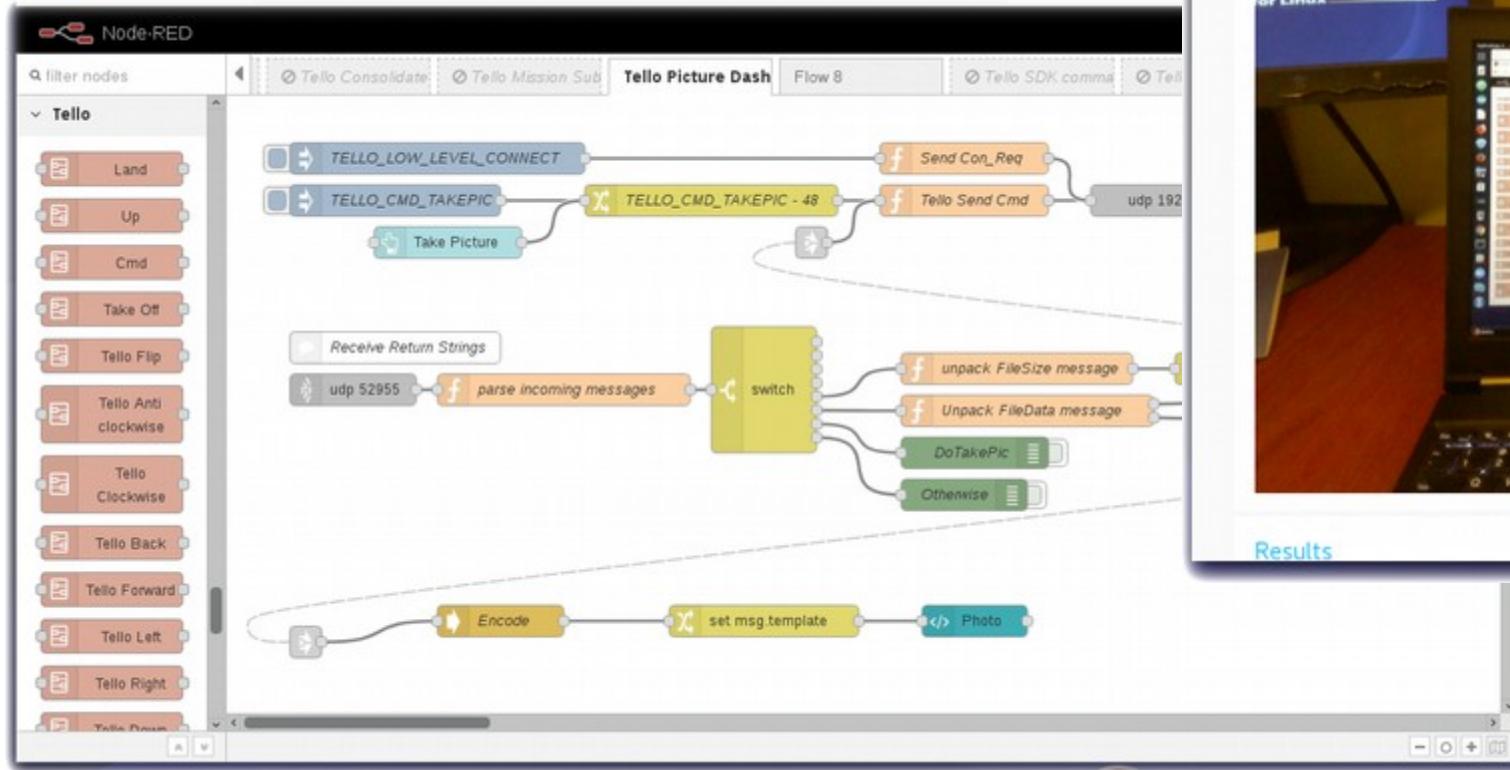
Nodes

- Clockwise (90°)
- Counter Clockwise
- Full Rotation (360°)
- Flip (forward)
- Emergency Stop
- Stats
- Battery
- Temp
- Barometer
- Height
- Accel
- Movement
- Tello Telemetry

Send Tello Drone on Autonomous Missions



Tello Camera



Tello Camera with Watson Visual Recognition

The screenshot displays two windows side-by-side. The left window is a 'Tello Camera Dashboard' in Mozilla Firefox, showing controls for a Tello drone and the results of a Watson Visual Recognition analysis of a living room scene. The right window is a 'Node-RED Dashboard - Mozilla Firefox' showing a flow diagram for processing Tello camera data.

Tello Camera Dashboard (Left Window):

- Controls:** Includes buttons for Emergency Stop, Takeoff, Land, and various movement commands (Up, Down, Left, Right, Forward, Backward, Clockwise, Anti-clockwise, Full Rotation, Flip).
- Camera:** A live video feed from the Tello drone's camera showing a living room with a brown sofa, a television, and some decorative items.
- Results:** Watson's default classifier thinks the picture contains a sage green color family room.

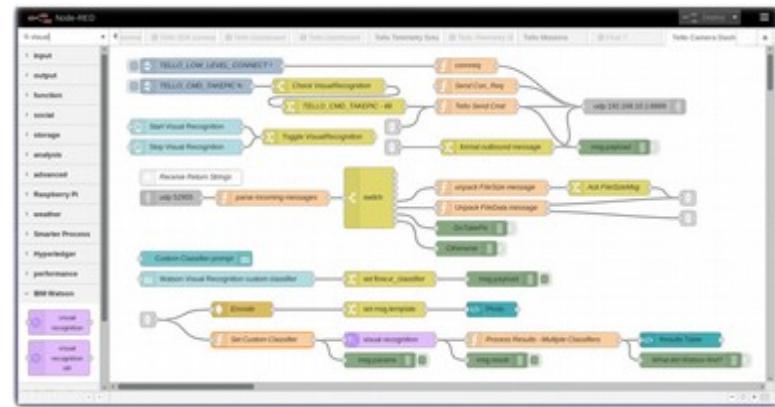
Class	Confidence
family room	0.652
indoors	0.88
living room	0.633
lounge	0.553
basement	0.522
support	0.522
supporting structure	0.522
den	0.5
sage green color	0.855
beige color	0.622

Node-RED Dashboard (Right Window):

The Node-RED flow diagram is titled 'Tello Picture Dash' and consists of the following nodes:

- Input nodes: TELLO_SLOW_LEVEL_CONNECT, TELLO_CMD_TWICFG, TELLO_CMD_TWICFG - AF, TELLO_CMD_TWICFG - RF.
- Process nodes: New Picture, Parse incoming messages, Watson Visual Recognition, Process Results, Set msg template, and a custom node labeled 'Watson Visual Recognition custom classifier'.
- Output nodes: Send Cmd_Pic, Send Cmd_Cfg, and a custom node labeled 'Tello Picture Dash'.
- Control nodes: a switch node and a function node labeled 'Update Picture Message'.
- Other nodes: a function node labeled 'Impact FileData message', a decision node, and a function node labeled 'Custom file extension'.

Tello Camera with Watson Visual Recognition Custom Classifier



Tello Camera Dashboard

Controls

- EMERGENCY STOP
- TAKEOFF
- LAND
- UP (50CM)
- DOWN (50CM)
- RIGHT (50CM)
- LEFT (50CM)
- FORWARD (50CM)
- BACKWARD (50CM)
- CLOCKWISE (90°)
- ANTI-CLOCKWISE (90°)
- FULL ROTATION (360°)
- FLIP (FORWARD)

Camera

START VISUAL RECOGNITION EVERY 10 SECONDS

STOP VISUAL RECOGNITION

Results

Watson's Bird Nests in Cell Towers
classifier thinks this picture contains a Nest.

Class	Confidence
Nest	0.893

Watson's default classifier thinks this picture contains a steel blue color barbed wire.

Class	Confidence
barbed wire	0.744
television antenna	0.715
antenna	0.743
electrical device	0.743
supporting tower	0.5
tower	0.501
wire	0.601
steel blue color	0.909
gray color	0.871

If you have created a Watson Visual Recognition custom classifier for drone image detection, enter the classifier ID here:
Watson Visual Recognition custom classifier
BirdNestsinCellTowers_276989758

Get the Code:

<https://github.com/johnwalicki/Node-RED-Tello-Control>



TelloPy

<https://github.com/hanyazou/TelloPy>

```
$ pip install tellropy
```



What does the code look like?

```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```



Standard setup – we need to access the DJI drone library and the standard time library from python

***from time import sleep
import tellopy***

What does the code look like?

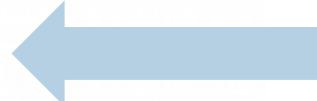
```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)
```

```
def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```



Simple Event Handler routine, we'll see it used later, but when the drone raises an event, this is called

Simple “handler” function to print out diagnostic information from your “in-flight” drone. More information can be found in your red folder

What does the code look like?

```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```

drone = tellopy.Tello()
•
•
drone.subscribe()

What does the code look like?

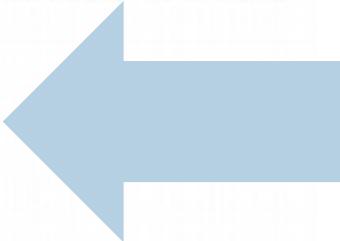
```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```



Standard stuff. We need to have your laptop connect to the drone – be sure your laptop is on the same SSID (wireless network as your drone)

drone.connect()
drone.wait_for_connection()
drone.takeoff()
sleep(5)

What does the code look like?

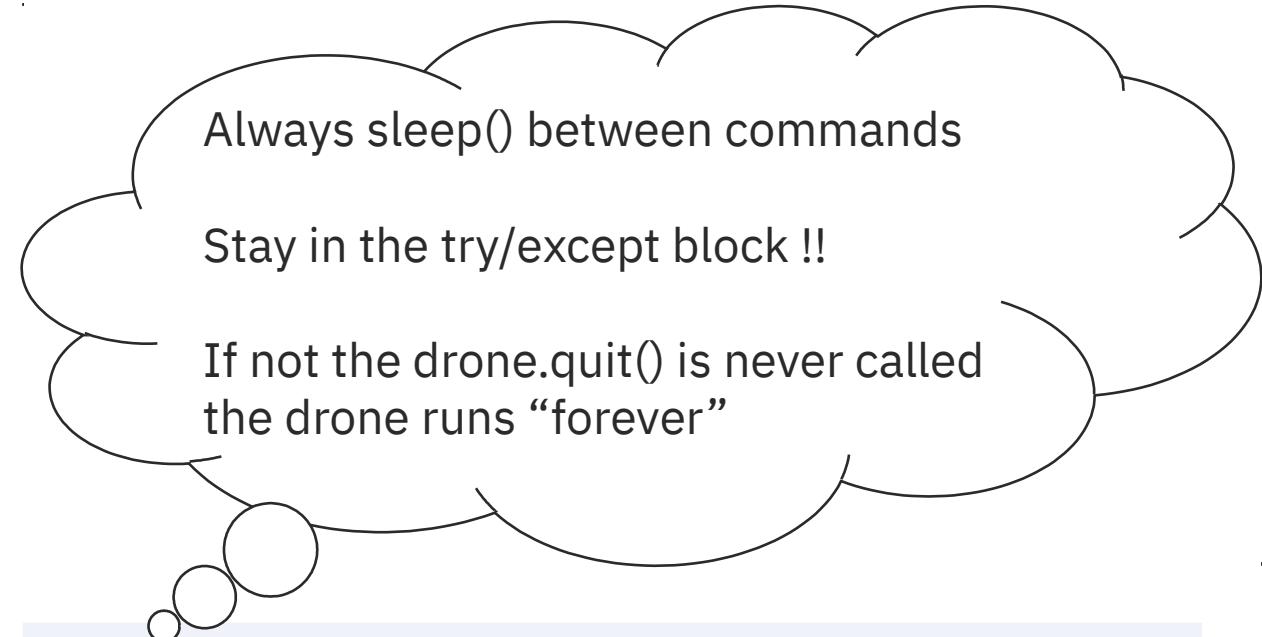
```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```



Drone control statements:

takeoff() – the drone starts up and hovers at about 2'
down(x) – moves down a speed of 'x' (0-100)
land() – drone immediately lands

***This is the typical flow: takeoff
set in motion
sleep (while it moves)***

What does the code look like?

```
from time import sleep
import tellopy

def handler(event, sender, data, **args):
    drone = sender
    if event is drone.EVENT_FLIGHT_DATA:
        print(data)

def test():
    drone = tellopy.Tello()
    try:
        drone.subscribe(drone.EVENT_FLIGHT_DATA, handler)

        drone.connect()
        drone.wait_for_connection(60.0)
        drone.takeoff()
        sleep(5)
        drone.down(50)
        sleep(5)
        drone.land()
        sleep(5)
    except Exception as ex:
        print(ex)
    finally:
        drone.quit()

if __name__ == '__main__':
    test()
```

One last thing

If you issue a:

- drone.up(50) and nothing else

The drone will rise up (50 cm/sec) forever (or until 20 feet or so)

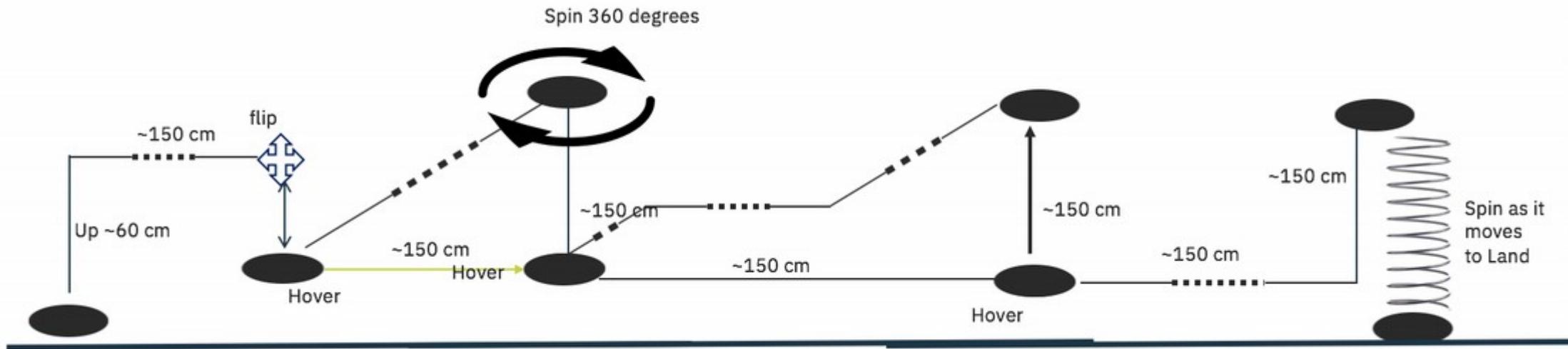
If you issue:

- Drone.up(50)
- Drone.forward(50)

It will go BOTH up and forward at 50 cm/sec – forever

To stop the motion, do a **drone.up(0)** (go up a zero speed – ie, hover) and **drone.forward(0)**

Create a Tello Flight Plan



Thank you

-  twitter.com/johnwalicki
-  github.com/johnwalicki
-  developer.ibm.com/profiles/walicki
-  linkedin.com/in/jowalicki



John Walicki

CTO, IoT / Edge Developer
Advocacy

IBM Academy of Technology
Leadership Team

Twitch TV | IBM Developer Channel

The screenshot shows the Twitch TV interface for the IBM Developer channel. At the top, there's a navigation bar with the channel name "IBMDeveloper", a video count of "Videos 48", a "Follow" button, and other menu options. To the left is a sidebar with various profile icons. The main content area displays a video player showing a webpage from the IBM Developer website. The page features a headline "1,500 drones are ready to fly." and an image of a DJI Tello drone. Below the video player, there's a caption "Highlight: #IBMDroneDrop - Drone Giveaway Winner..." and a category "Category: Science & Technology". The video player also shows a timestamp of 00:01:07 and a "Settings" button. On the right side of the screen, there are two smaller video preview boxes for other users: one for "@nibalizer" and another for "johnwalicki". The bottom right corner of the main video player has a "Share" button and a three-dot menu icon.

IBMDeveloper

Videos 48

Clips

Events

...

Follow

IBMDeveloper

1,500 drones are ready to fly.

Enter to win one of 1,500 DJI Tello drones. Then get started on the Drone Programming Experience and develop with Watson Visual Recognition, Watson IoT, IBM Cloud, and IBM Data and Analytics. We'll give you all the tools you need to program your drone.

ENTER TO WIN

U.S./Canada only

00:01:07

Free code patterns to program your drone.

Highlight: #IBMDroneDrop - Drone Giveaway Winner...

Category: Science & Technology

413 Share

@nibalizer

@johnwalicki

IBMDeveloper 00:14:49

Settings



Please note

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice and at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.

The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Notices and disclaimers

© 2019 International Business Machines Corporation. No part of this document may be reproduced or transmitted in any form without written permission from IBM.

U.S. Government Users Restricted Rights – use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed “as is” without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.”

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimers continued

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.