



LYU1401 - AndroidCopter

Supervised by Prof. LYU Rung Tsong Michael

Lam Ka Ho 1155018465

Wong Chor Man 1155018466

Recap - Architecture

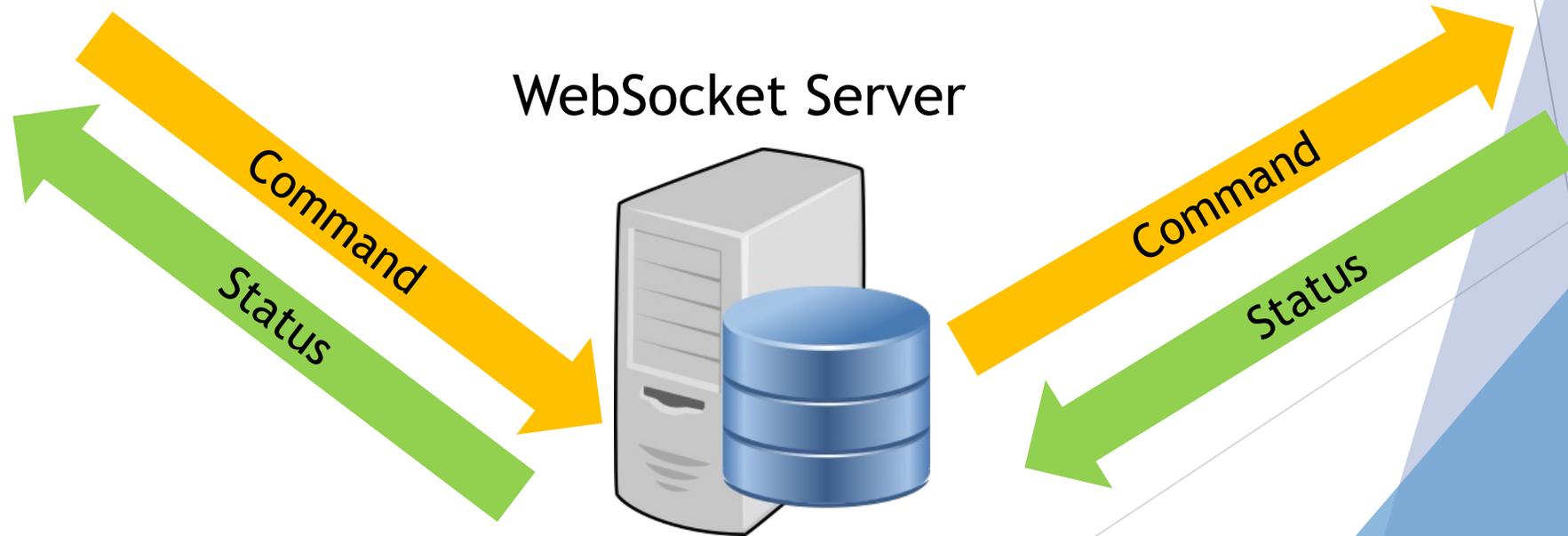
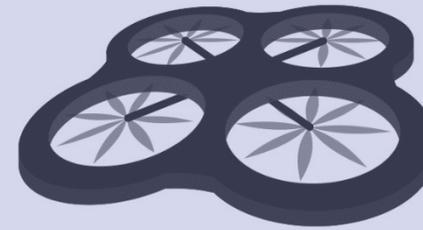
Remote Control Panel

Send Control Command
Receive the status



AndroidCopter

Receive Control Command
Send the status



Recap - Milestone

► Stabilize



Recap - Milestone

- ▶ Control manually
- ▶ Take video and photo

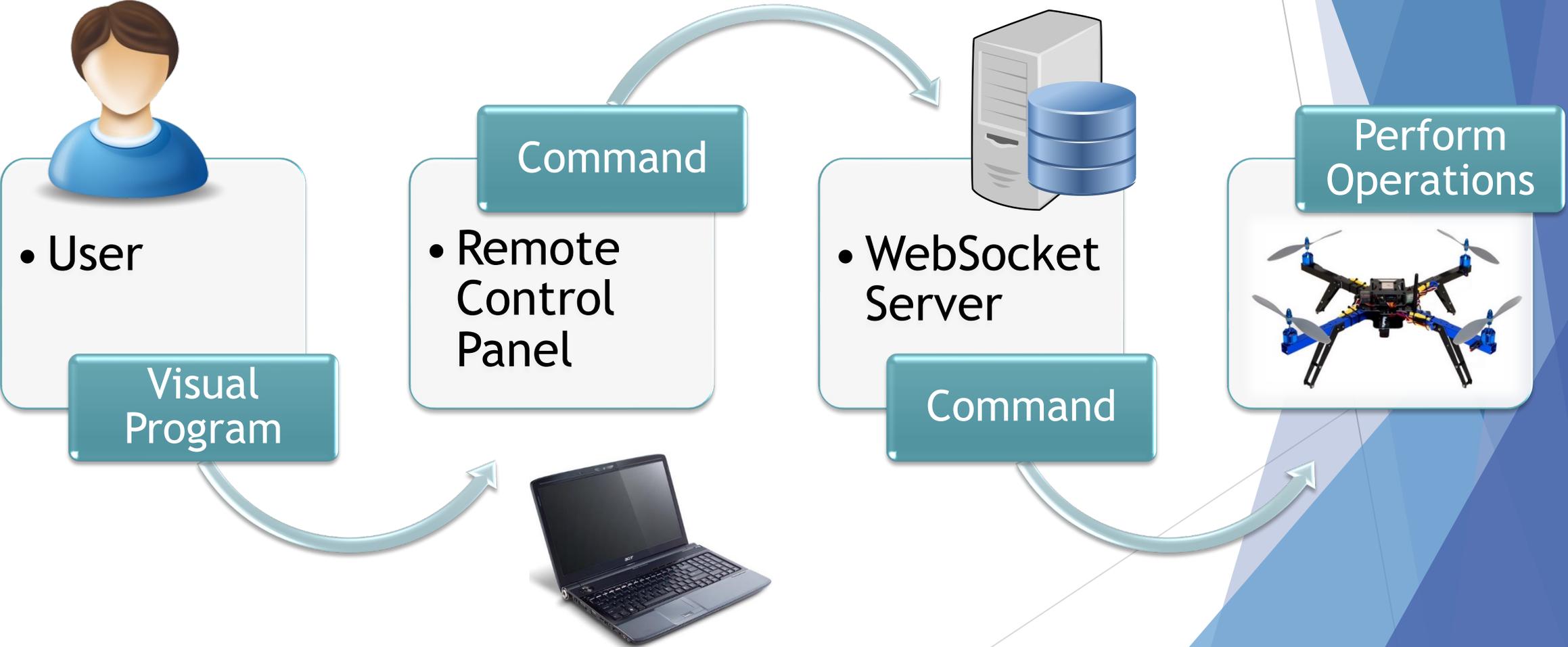


Objectives

1. Autopilot
 1. Indoor path planning using Visual Programming
 2. Push a program to AndroidCopter
2. Improve the stabilization with optical flow sensor



Data Flow Diagram



Remote Control Panel - Route Page

Control ⚙️ Route 🗺️ Map 📍 AndroidPhoneCopter - Remote Control P...

Functions

- Actions
- Values

Forward ^

Backward v

Leftward <

Rightward >

Upward ↑

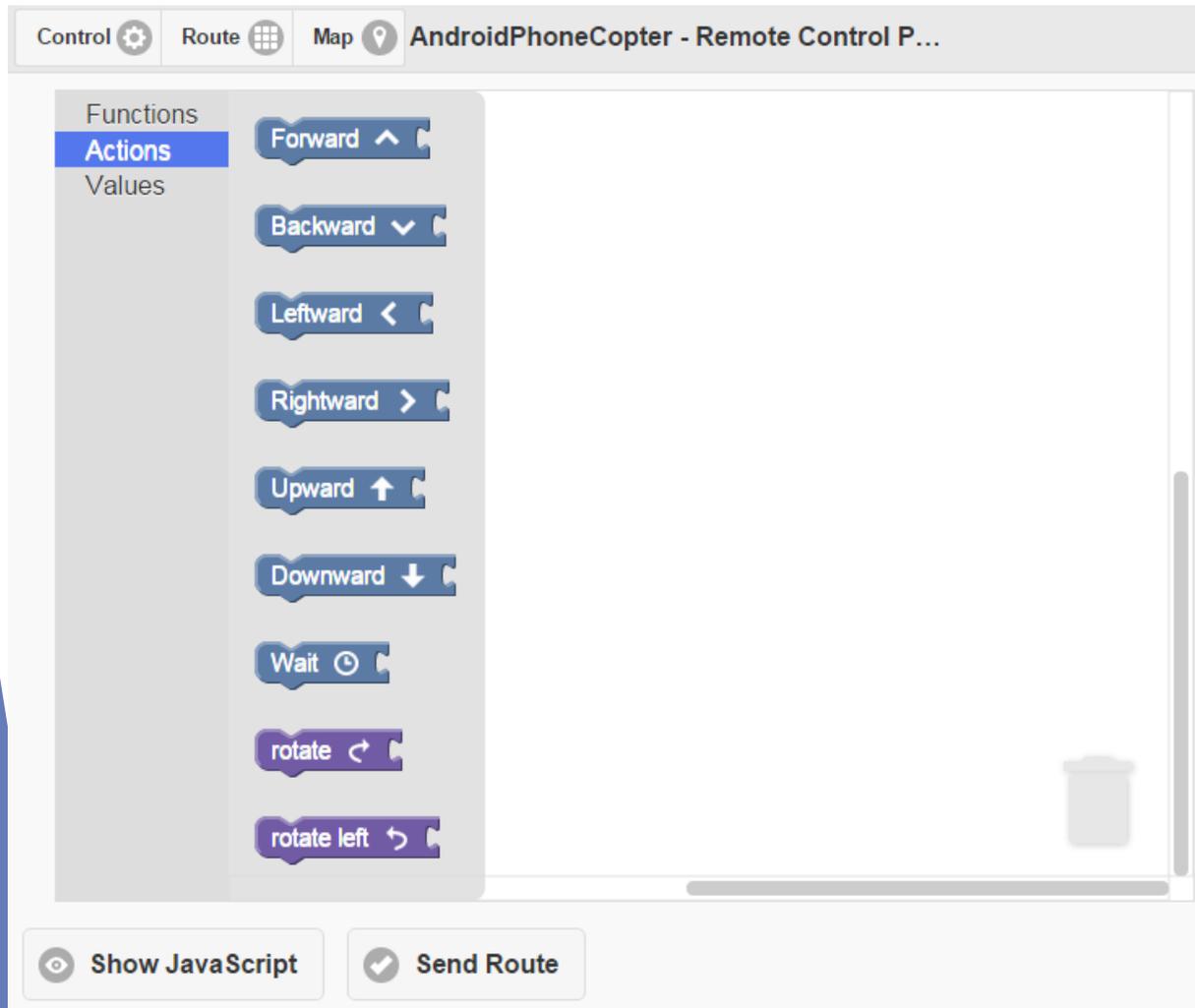
Downward ↓

Wait ⏸️

rotate ↶

rotate left ↷

Show JavaScript Send Route



Control ⚙️ Route 🗺️ Map 📍 AndroidPhoneCopter - Remote Control P...

Functions

- Actions
- Values

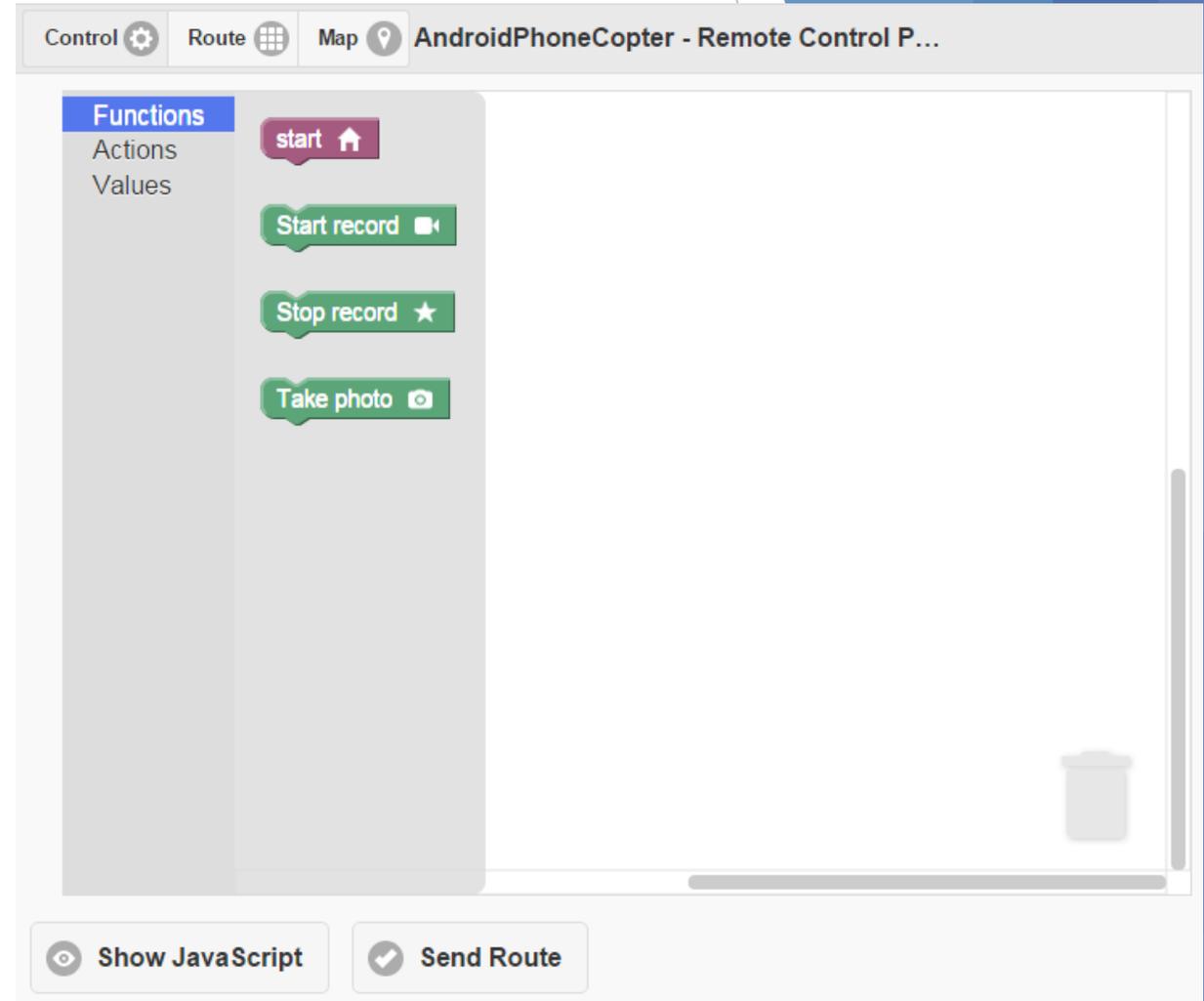
start 🏠

Start record 🎥

Stop record ★

Take photo 📷

Show JavaScript Send Route



Remote Control Panel - Puzzle

Basic
Operation

Advanced
Operation

Value
Puzzle

Forward ^

Rightward >

Wait ⌚

Backward v

Upward ↑

rotate ↻

Leftward <

Downward ↓

rotate left ↶

start 🏠

Start record 📹

Stop record ★

Take photo 📷

1

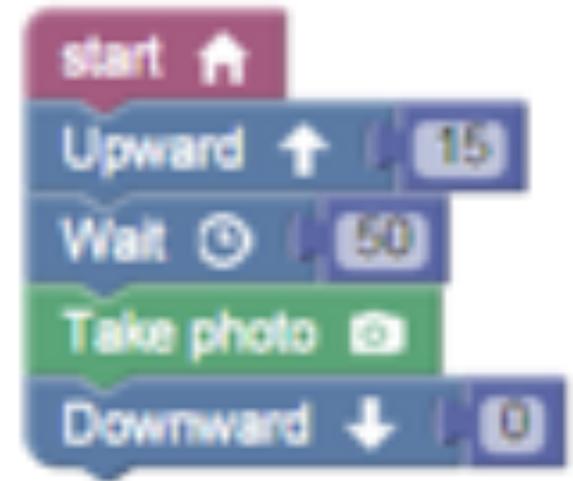
2

3

90°

Remote Control Panel - Example

1. Start
2. Upward by 15 cm
3. Wait for 50 second
4. Take a photo
5. Downward by 0 cm
(trigger Auto Landing Function)



Remote Control Panel

Show JavaScript Send Route



Commands are sent to
Android Copter



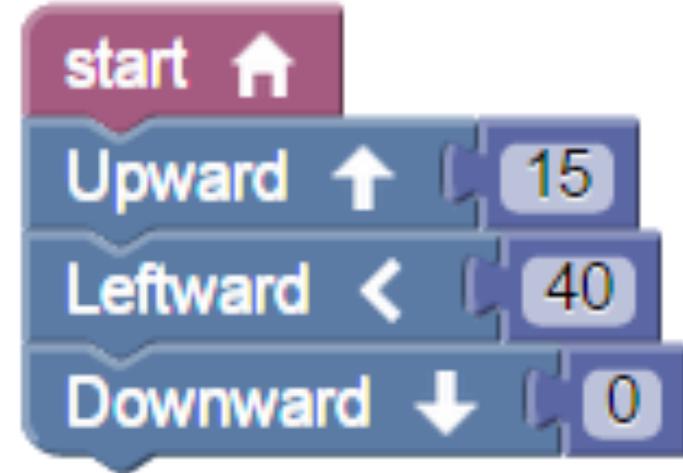
```
Code:  
start();  
upward(15);  
wait(50);  
take_photo();  
downward(0);
```

Demo

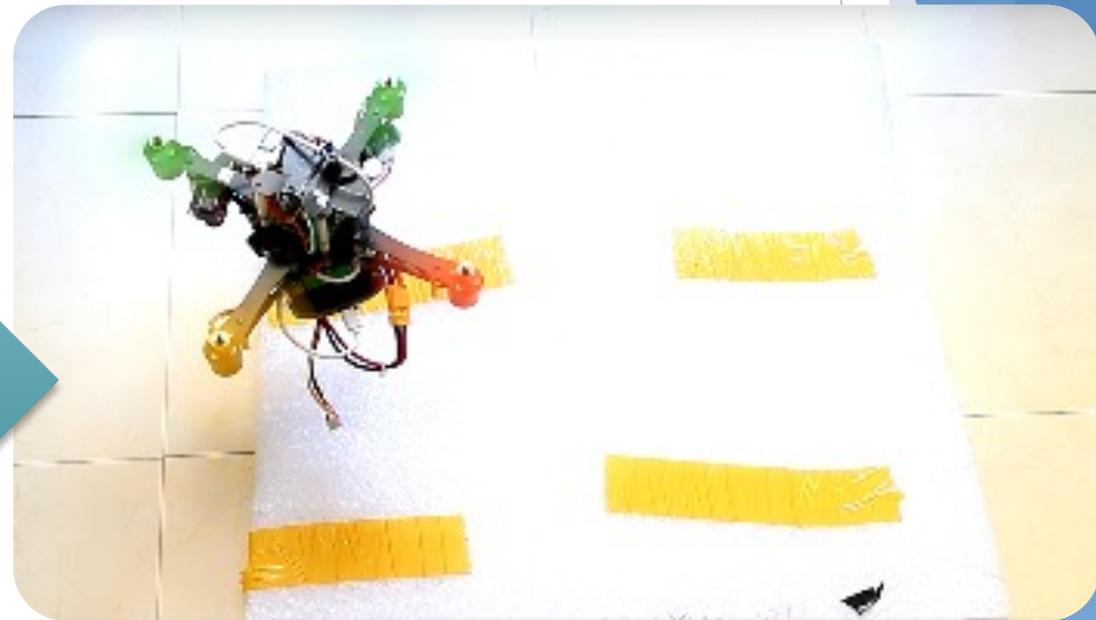
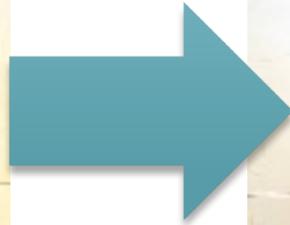
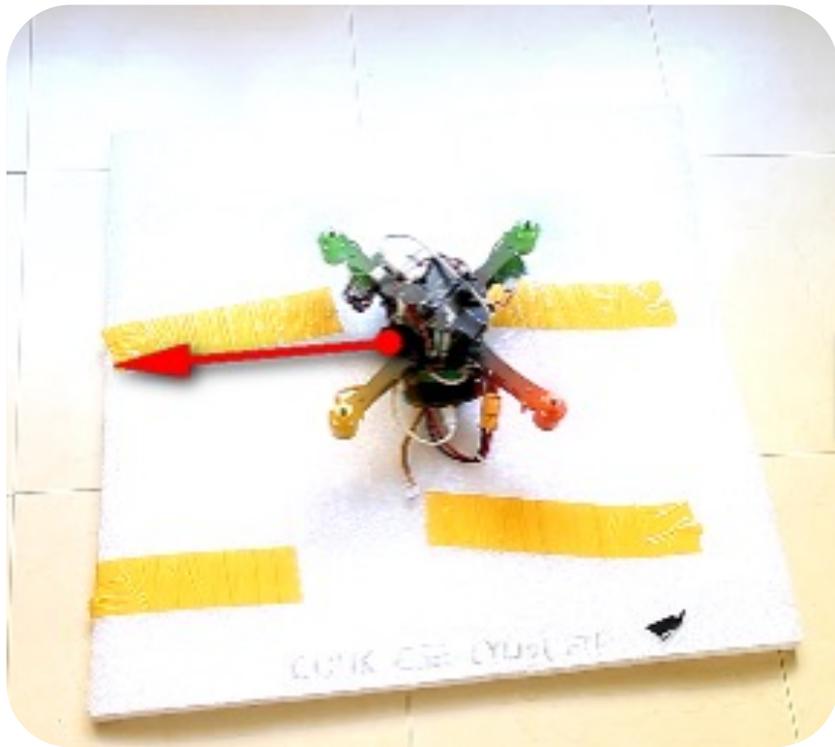
▶ <http://leah.ddns.net/controller/>

Demonstration - Go Left

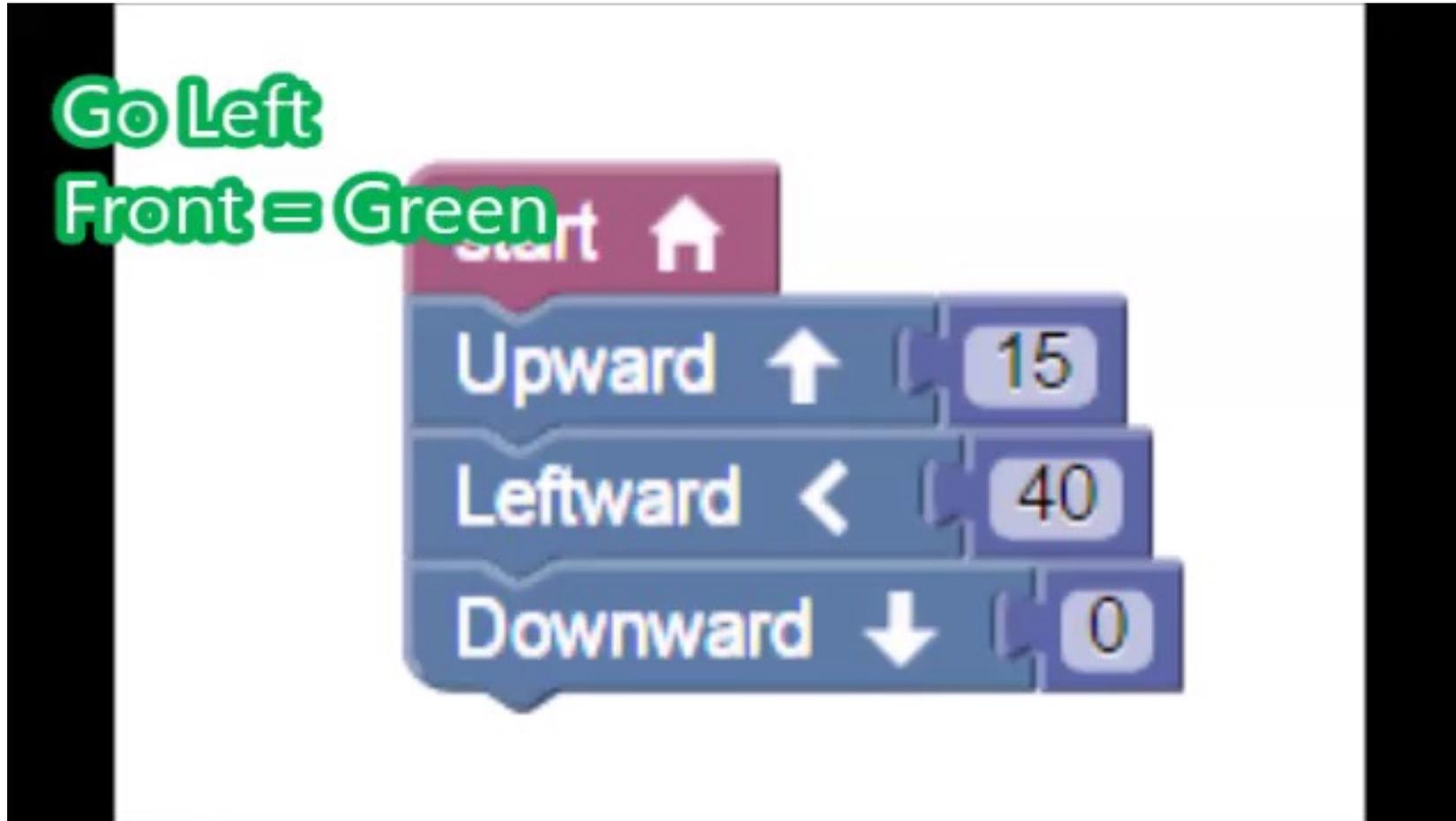
1. Start
2. Upward by 15 cm
3. Leftward by 40 cm
4. Downward by 0 cm
(Trigger Auto Landing Function)



Demonstration - Go Left

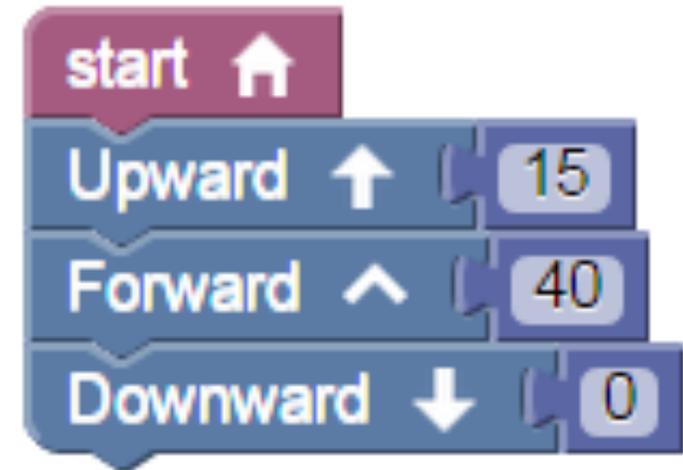


Demonstration - Go Left

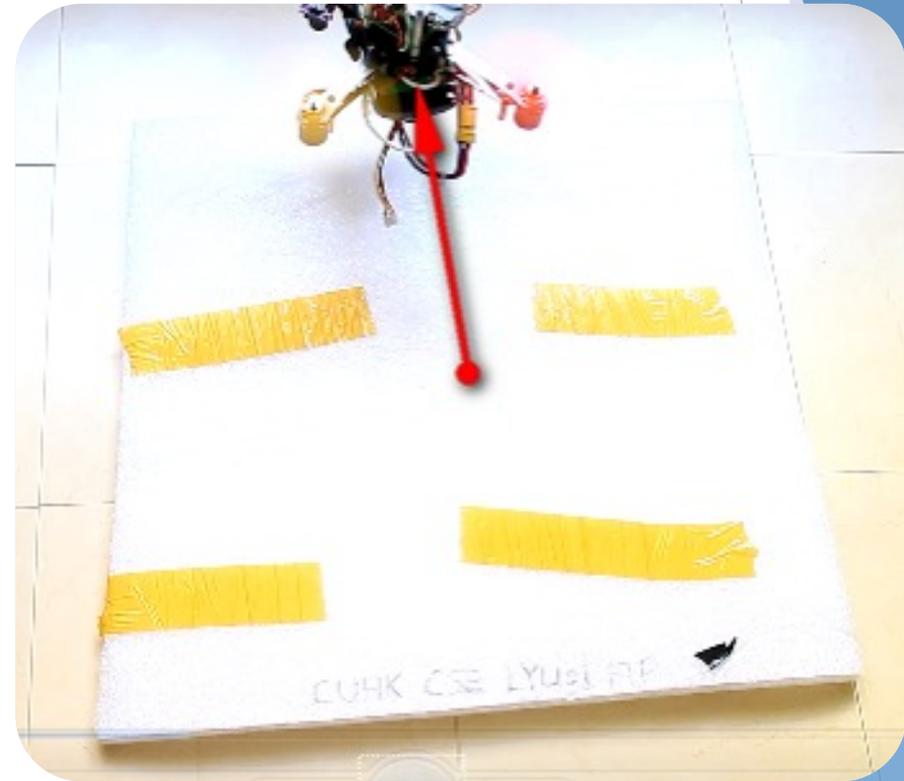
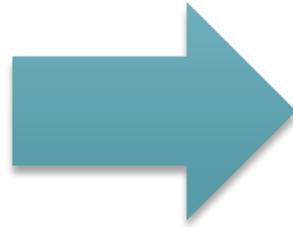
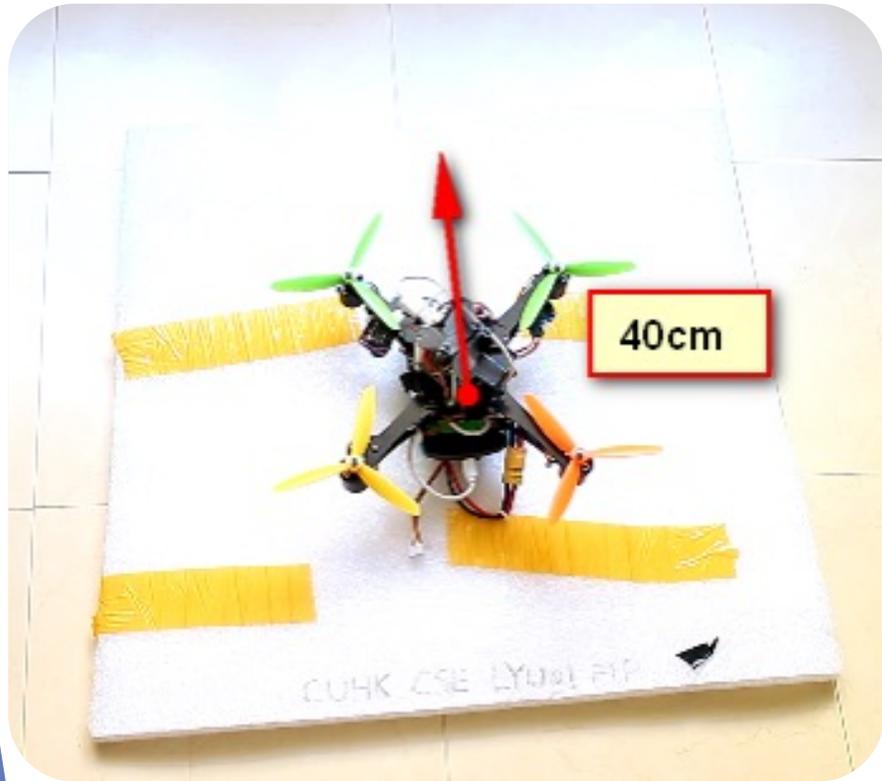


Demonstration - Forward

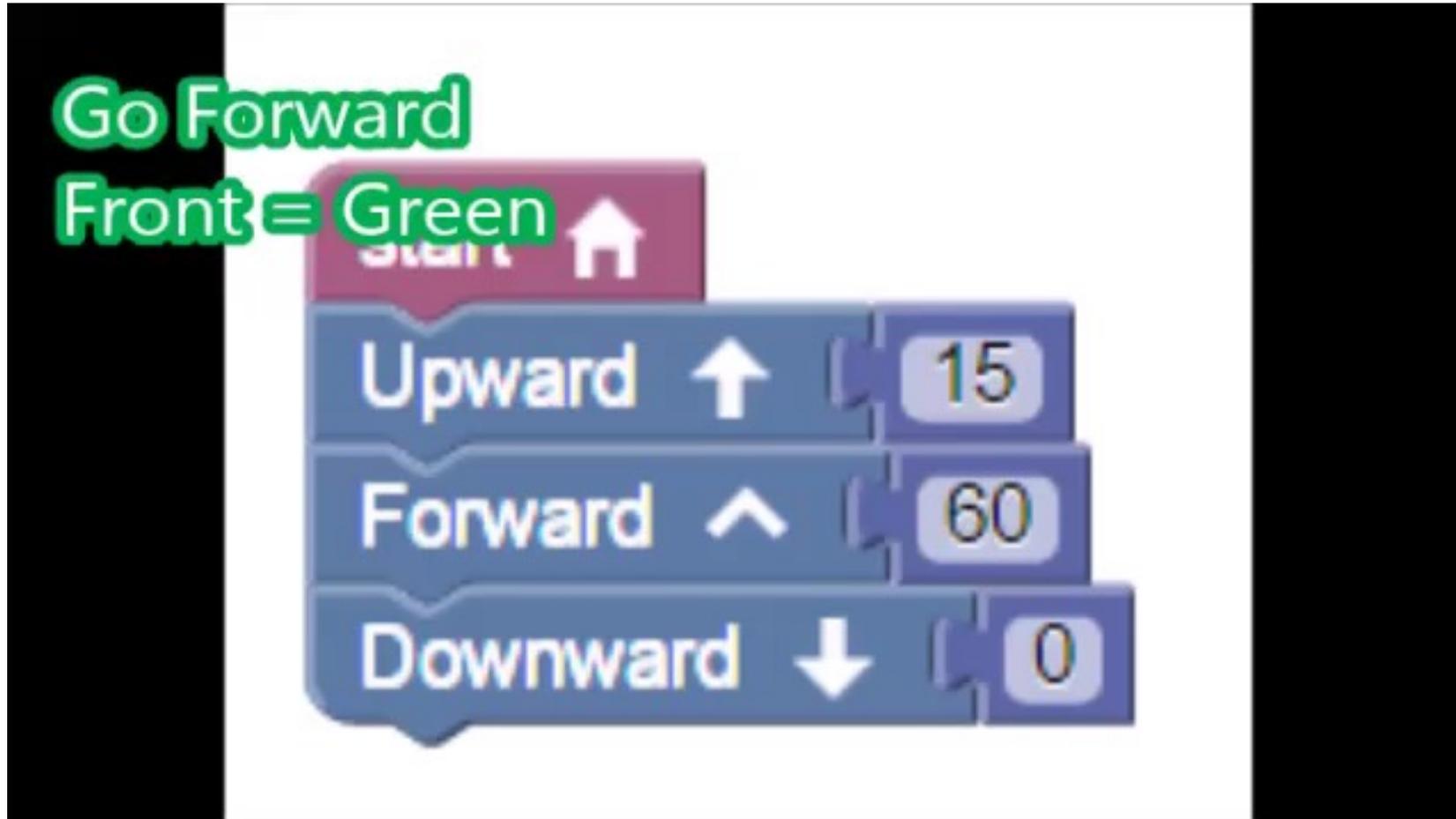
1. Start
2. Upward by 15 cm
3. Forward by 40 cm
4. Downward by 0 cm
(Trigger Auto Landing Function)



Demonstration - Forward

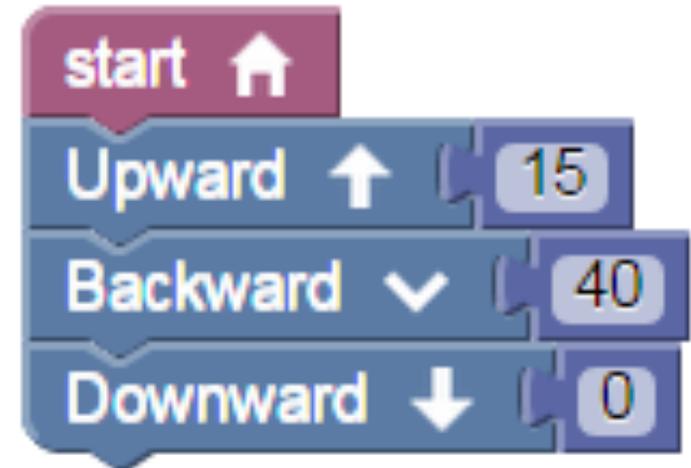


Demonstration - Forward

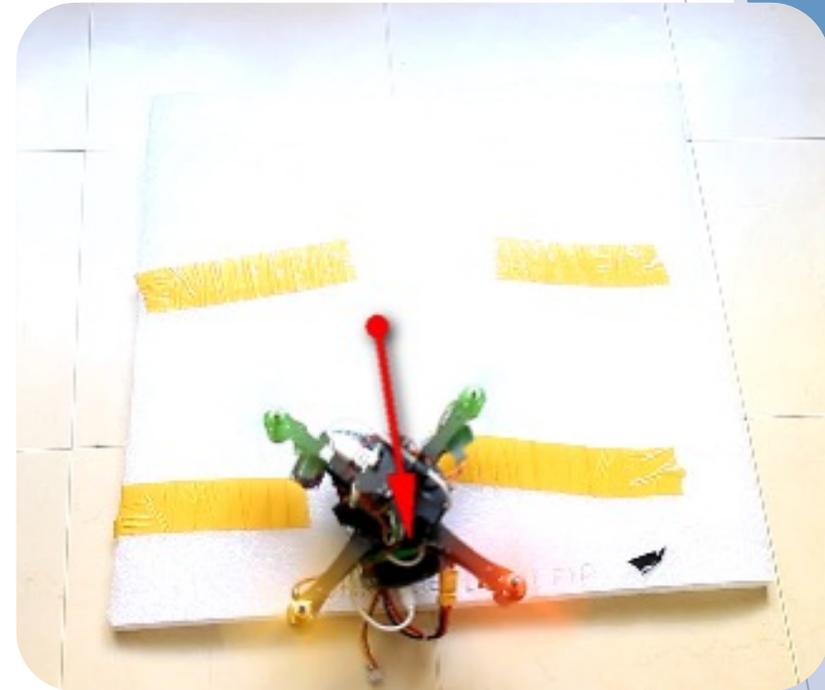
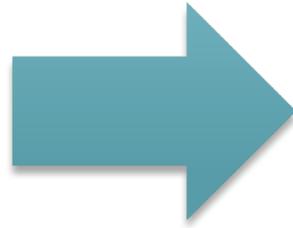
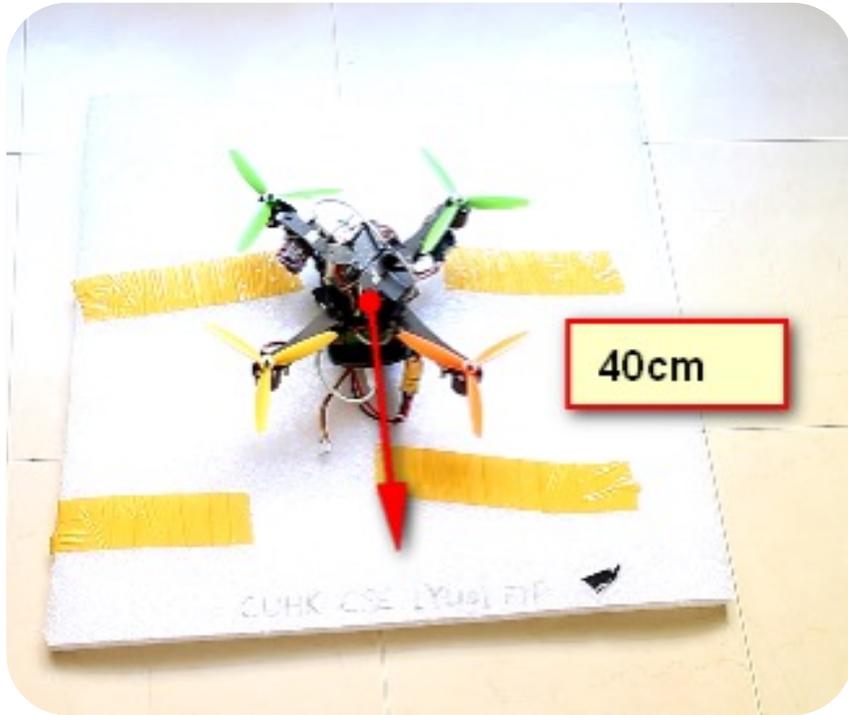


Demonstration - Backward

1. Start
2. Upward by 15 cm
3. Backward by 40 cm
4. Downward by 0 cm
(Trigger Auto Landing Function)

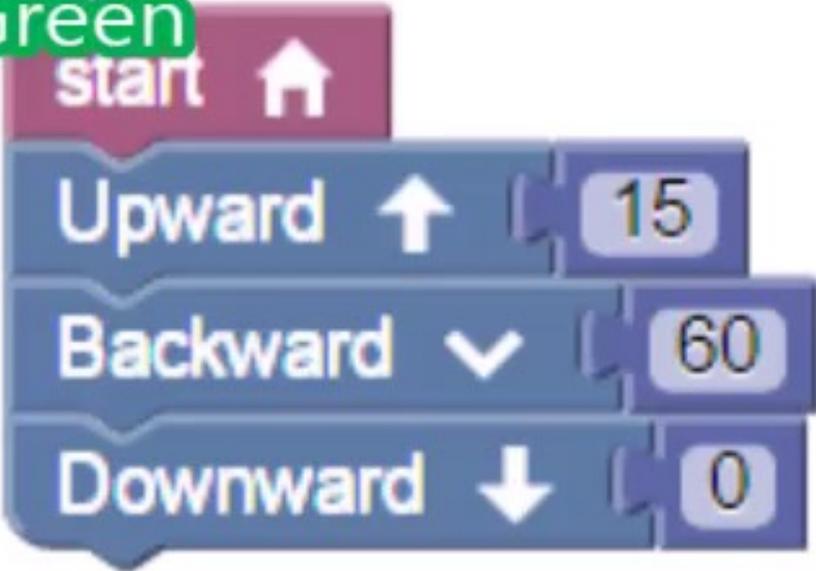


Demonstration - Backward



Demonstration - Backward

Go Backward
Front = Green



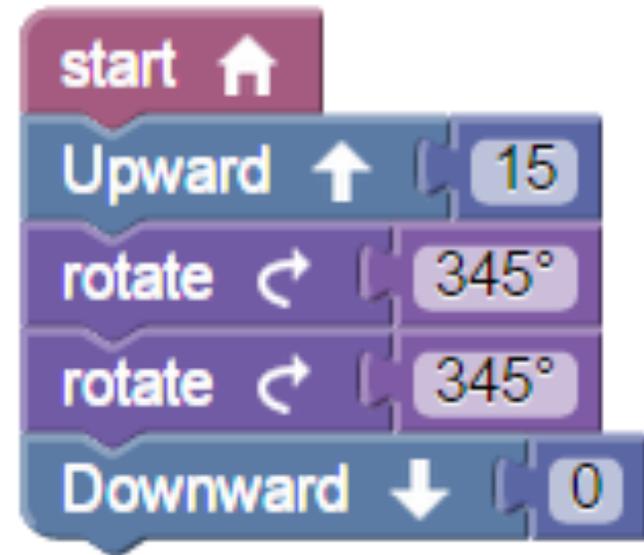
The image shows a Scratch script with the following blocks:

- start** (pink block with a house icon)
- Upward** (blue block with an upward arrow icon and a value of 15)
- Backward** (blue block with a downward arrow icon and a value of 60)
- Downward** (blue block with a downward arrow icon and a value of 0)

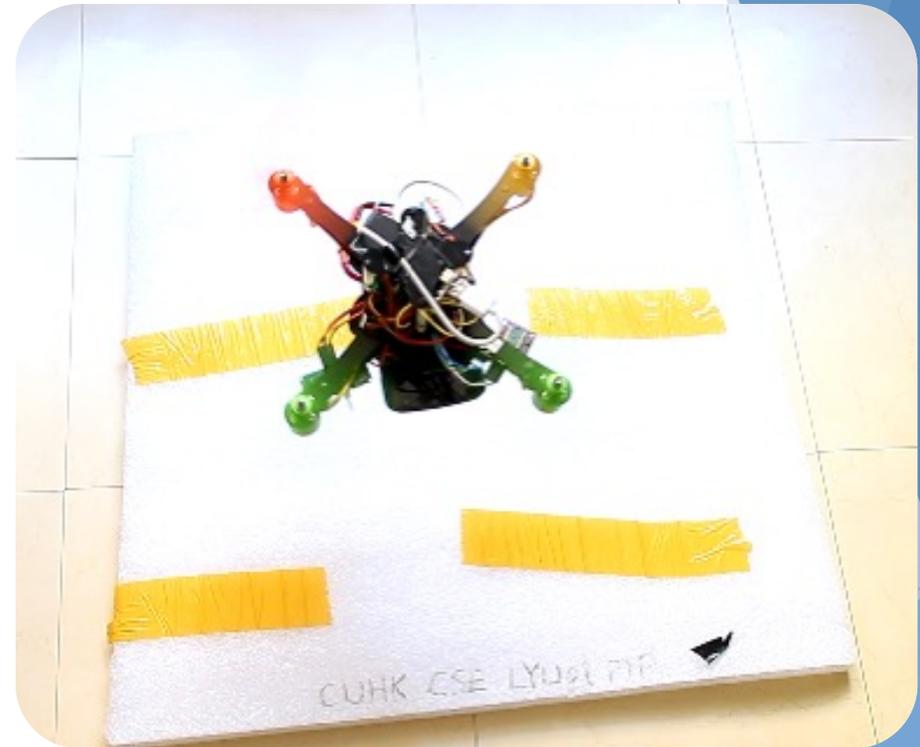
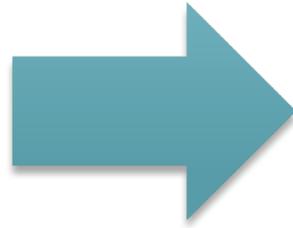
The script is set against a black background. The text 'Go Backward' and 'Front = Green' is written in green above the script.

Demonstration - Rotation

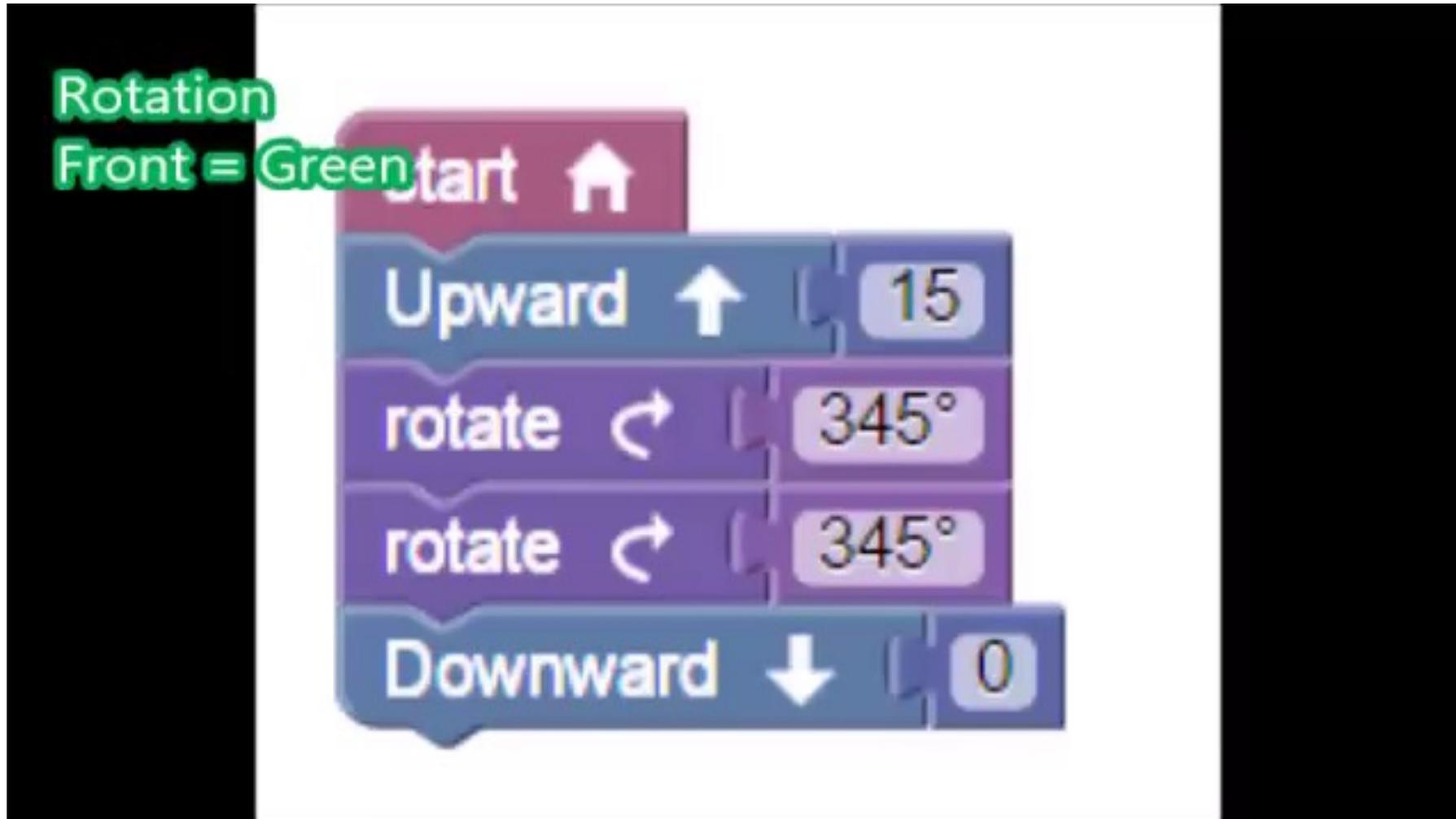
1. Start
2. Upward by 15 cm
3. Rotate by 345 degree
4. Rotate by 345 degree
5. Downward by 0 cm
(Trigger Auto Landing Function)



Demonstration - Rotation



Demonstration - Rotation



Demonstration - Hovering

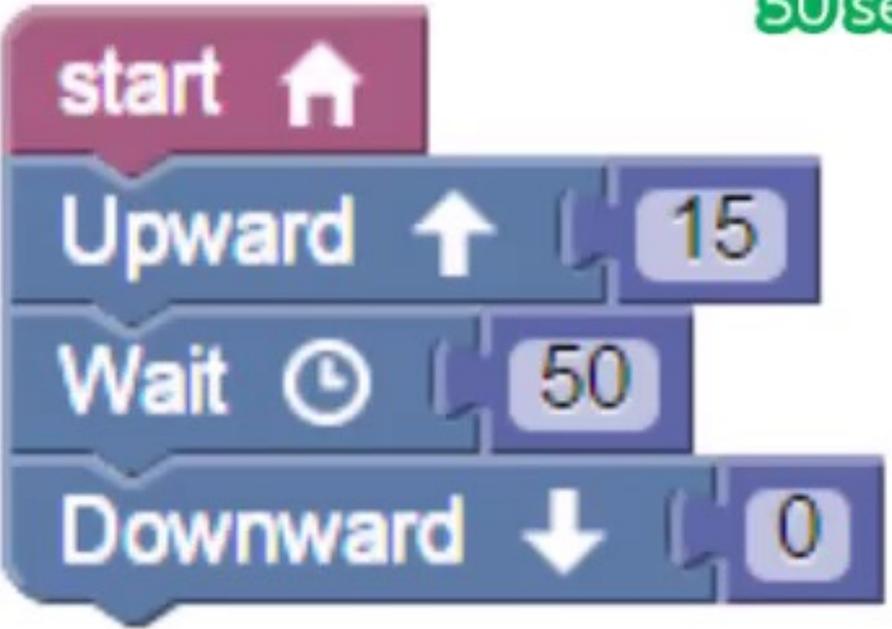
1. Start
2. Upward by 15 cm
3. Wait for 50 second
4. Downward by 0 cm
(Trigger Auto Landing Function)



Demonstration - Hovering



Demonstration - Hovering



Hovering in the air
50 sec (2x playback)

start ↑

Upward ↑ 15

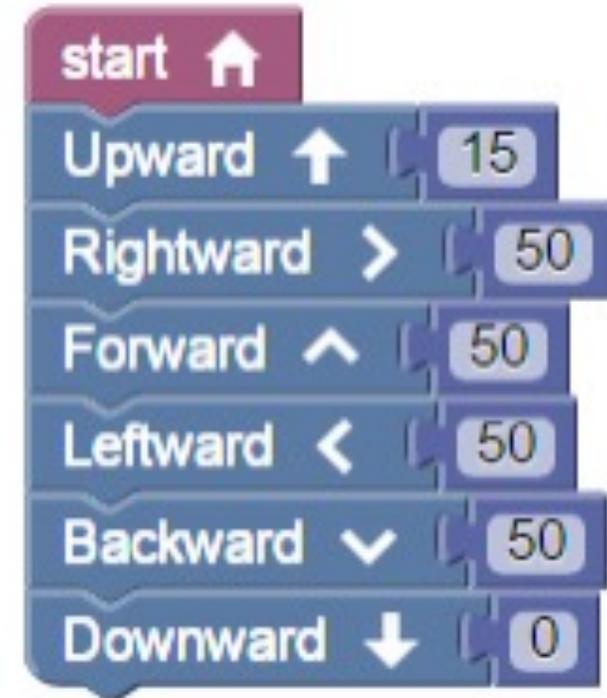
Wait ⌚ 50

Downward ↓ 0

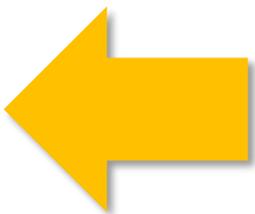
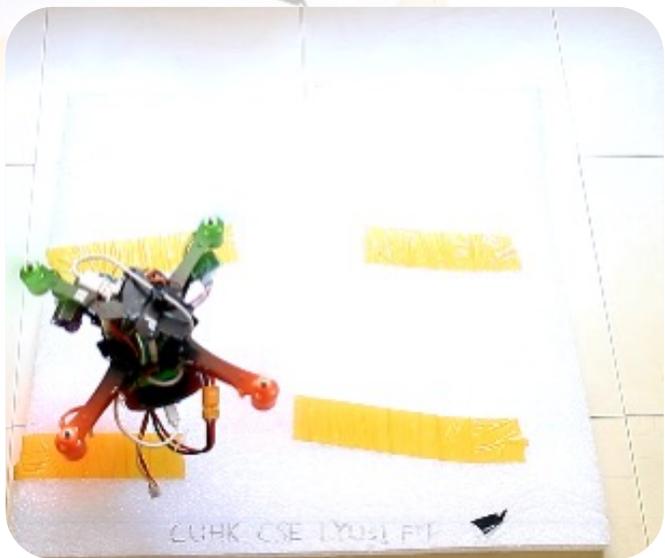
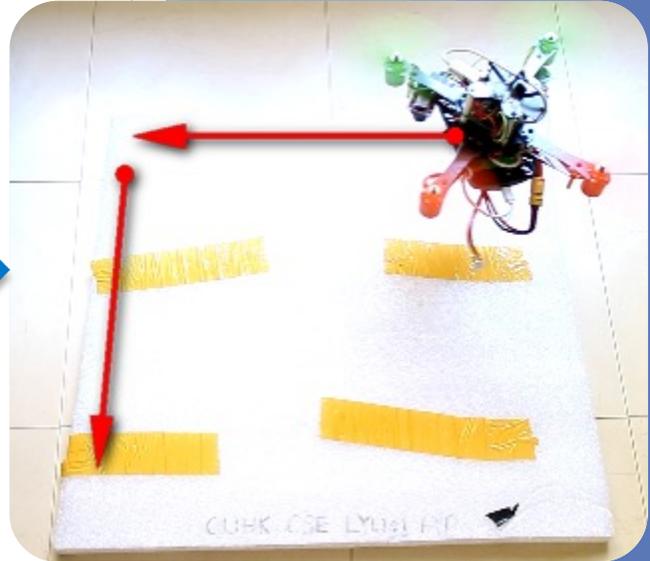
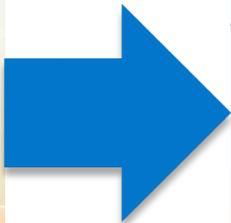
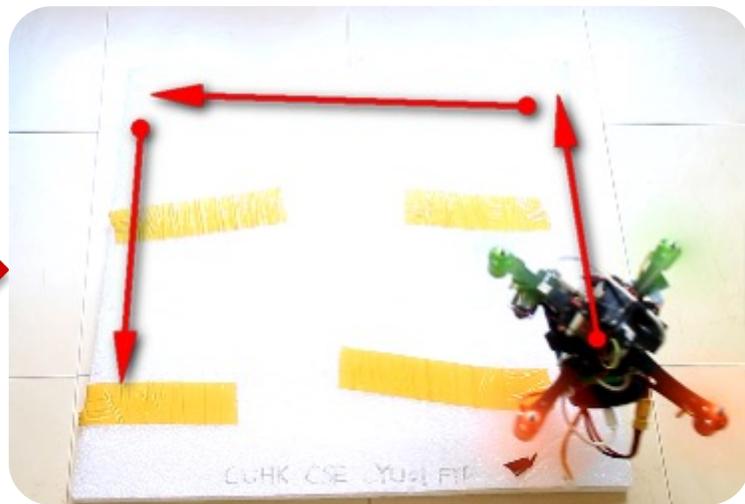
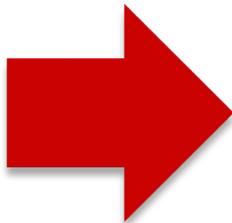
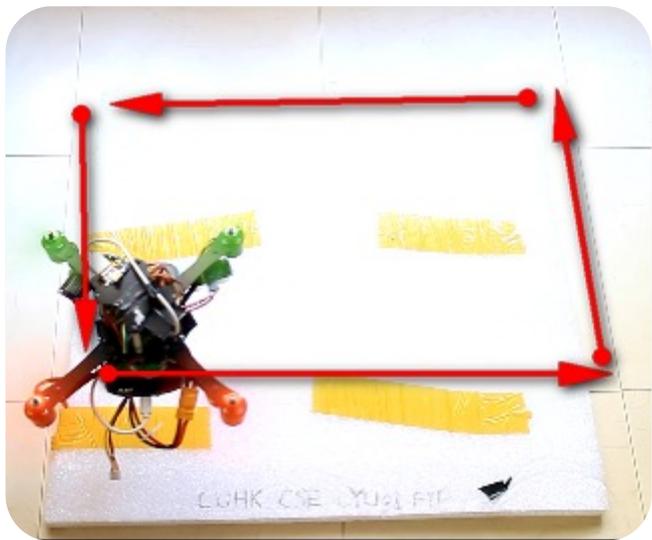
The image shows a Scratch script for a hovering demonstration. It consists of four blocks: a 'start' block with an upward arrow icon, an 'Upward' block with an upward arrow icon and a value of 15, a 'Wait' block with a clock icon and a value of 50, and a 'Downward' block with a downward arrow icon and a value of 0. The text 'Hovering in the air 50 sec (2x playback)' is written in green above the script.

Demonstration - Rectangular Path

1. Start
2. Upward by 15 cm
3. Rightward by 50 cm
4. Forward by 50 cm
5. Leftward by 50 cm
6. Backward by 50 cm
7. Downward by 0 cm
(Trigger Auto Landing Function)

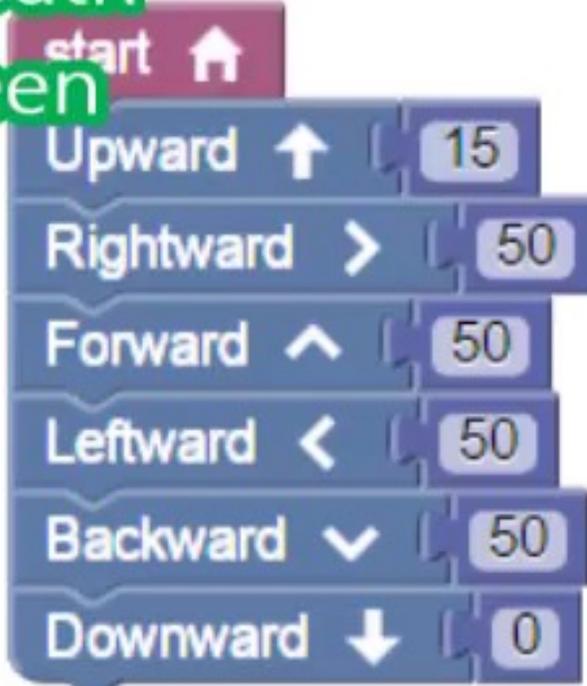


Demonstration - Rectangular Path



Demonstration - Rectangular Path

Rectangle path
Front = Green

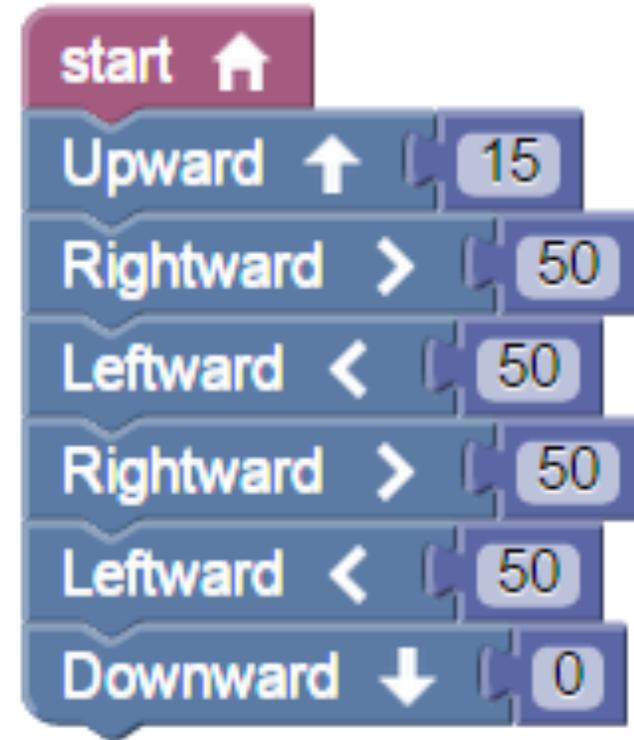


The image shows a sequence of Scratch code blocks for a rectangular path. The blocks are stacked vertically and connected by their interlocking tabs and sockets. The sequence starts with a pink 'start' block with a house icon. This is followed by a blue 'Upward' block with an upward arrow and a value of 15. Next is a blue 'Rightward' block with a rightward arrow and a value of 50. This is followed by a blue 'Forward' block with an upward arrow and a value of 50. Next is a blue 'Leftward' block with a leftward arrow and a value of 50. This is followed by a blue 'Backward' block with a downward arrow and a value of 50. Finally, the sequence ends with a blue 'Downward' block with a downward arrow and a value of 0.

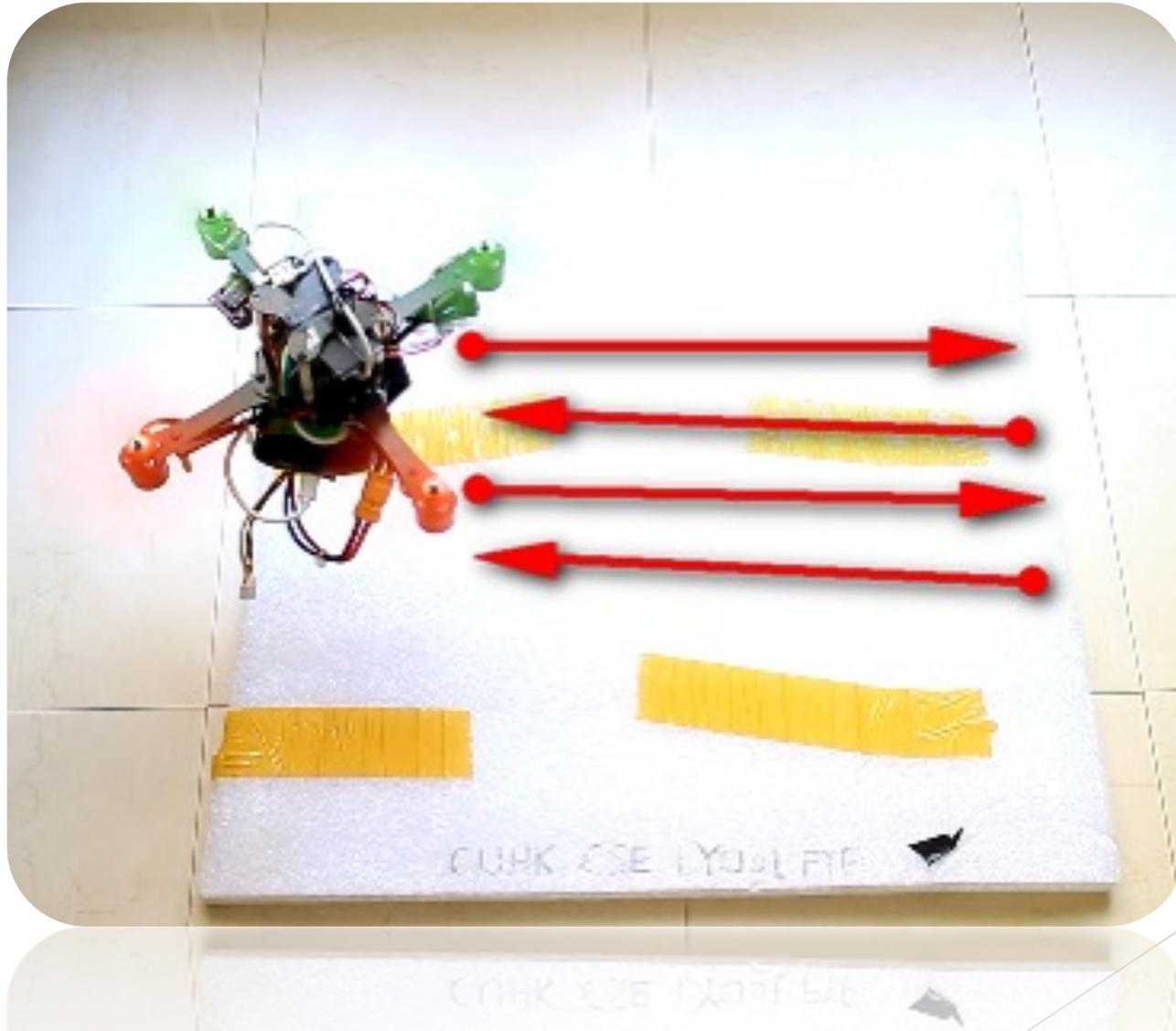
Block	Value
start	-
Upward	15
Rightward	50
Forward	50
Leftward	50
Backward	50
Downward	0

Demonstration - Right, left, right, left

1. Start
2. Upward by 15 cm
3. Rightward by 50 cm
4. Leftward by 50 cm
5. Rightward by 50 cm
6. Leftward by 50 cm
7. Downward by 0 cm
(Trigger Auto Landing Function)



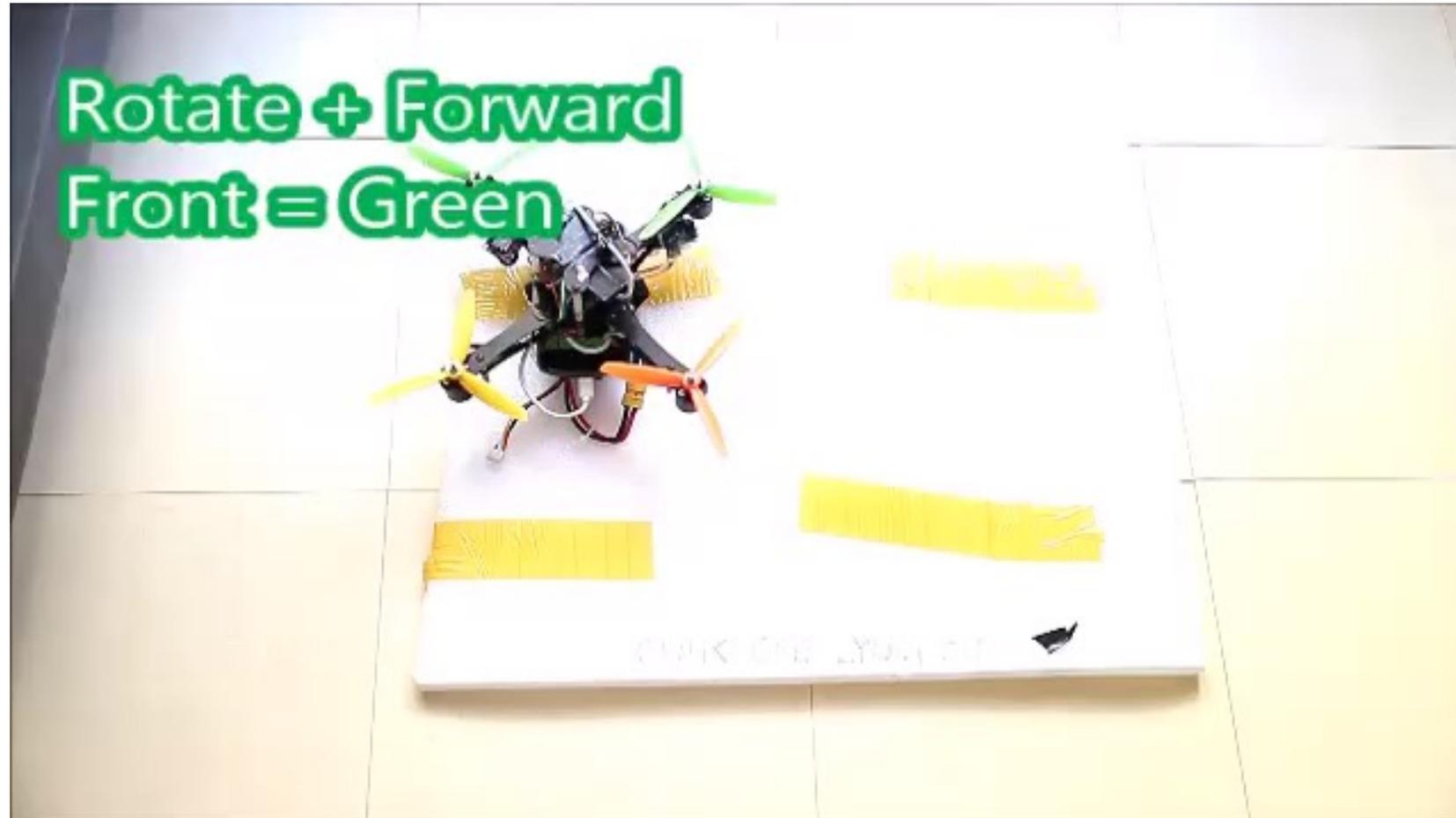
Demonstration - Right, left, right, left



Demonstration - Right, left, right, left



Demonstration - Rotate & Forward

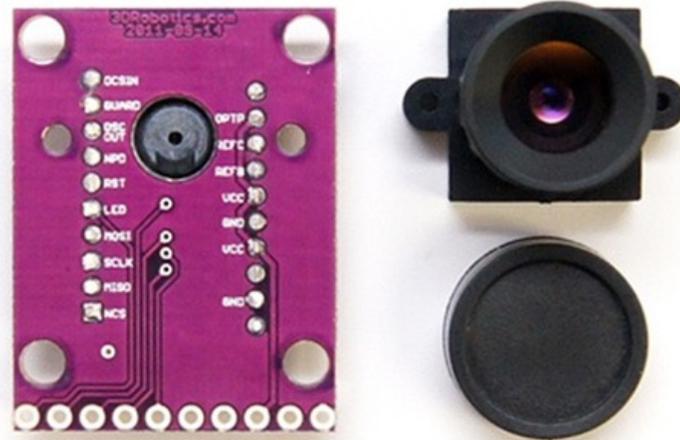


Demonstration - Camera Capture



What is Optical Flow Sensor?

- ▶ A mouse sensor
- ▶ A camera to make film capture



How the Optical Flow Sensor work?

High
Resolution

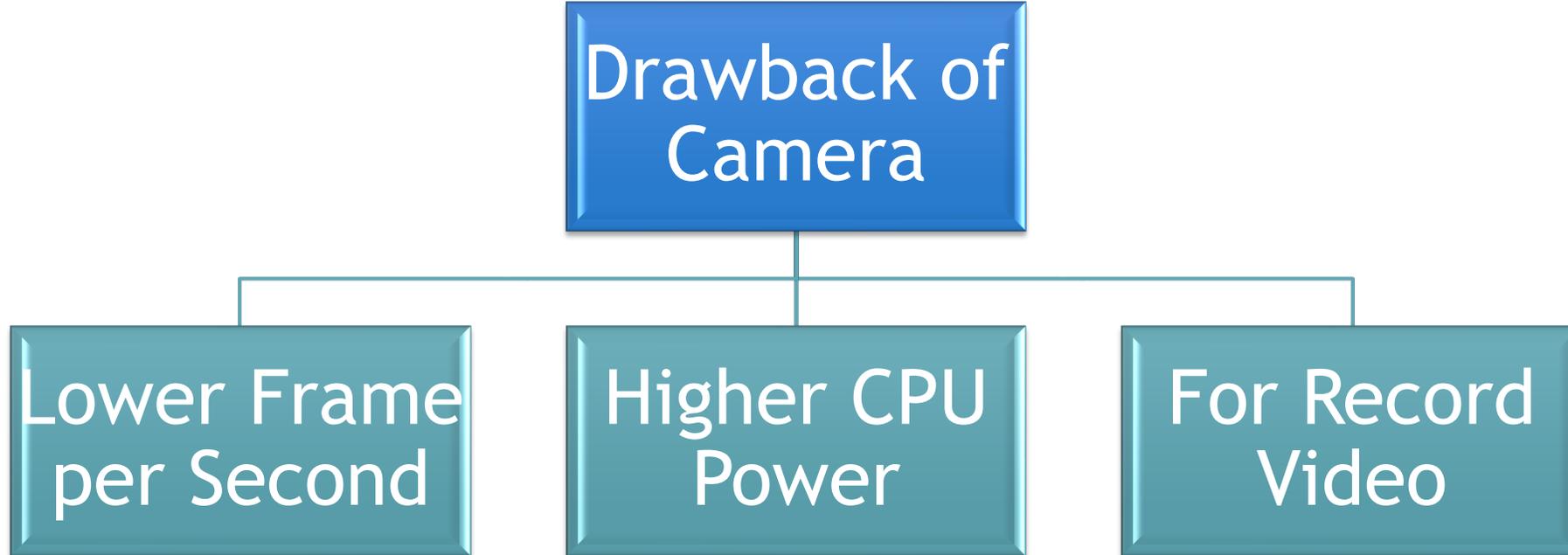
High
Speed

SPI
Interface

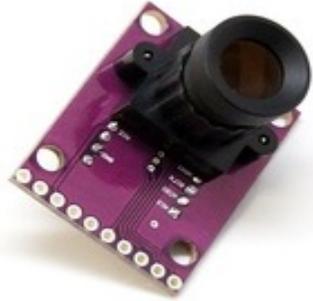
Why Optical Flow Sensor is needed?

- ▶ Newton's first law: a moving object will continue moving
- ▶ Optical Flow Sensor detects the movement to prevent the quadcopter move away.
- ▶ Accuracy: roughly 10cm
- ▶ GPS Accuracy: more than 100cm (outdoor only)

Why not use Camera instead of Optical Flow Sensor?



Usage in this Project

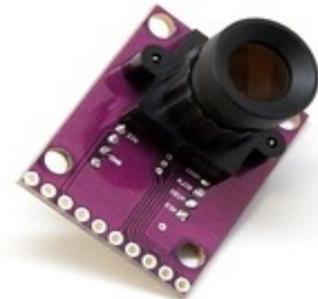
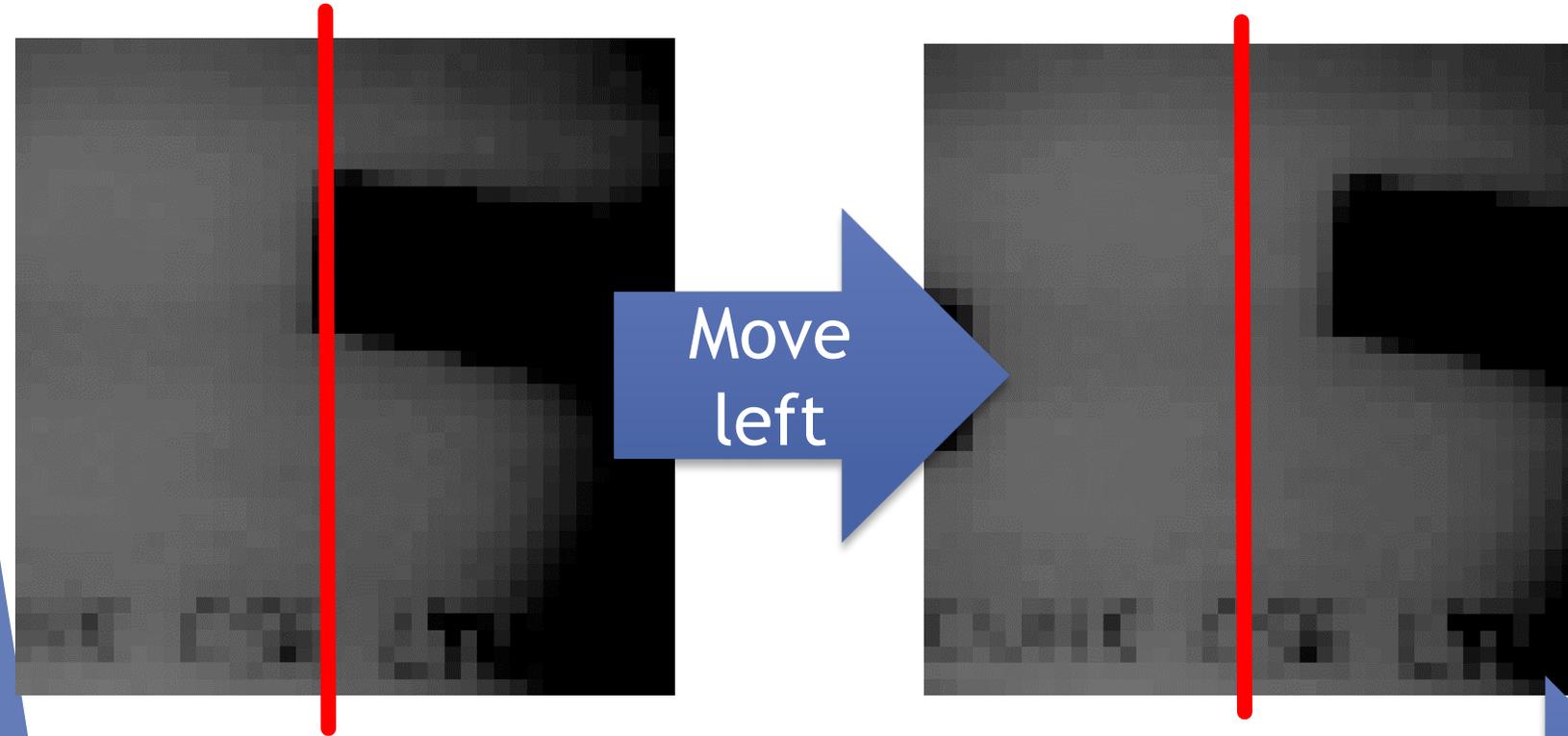


Improve the
Stability

Hovering in a fixed Point

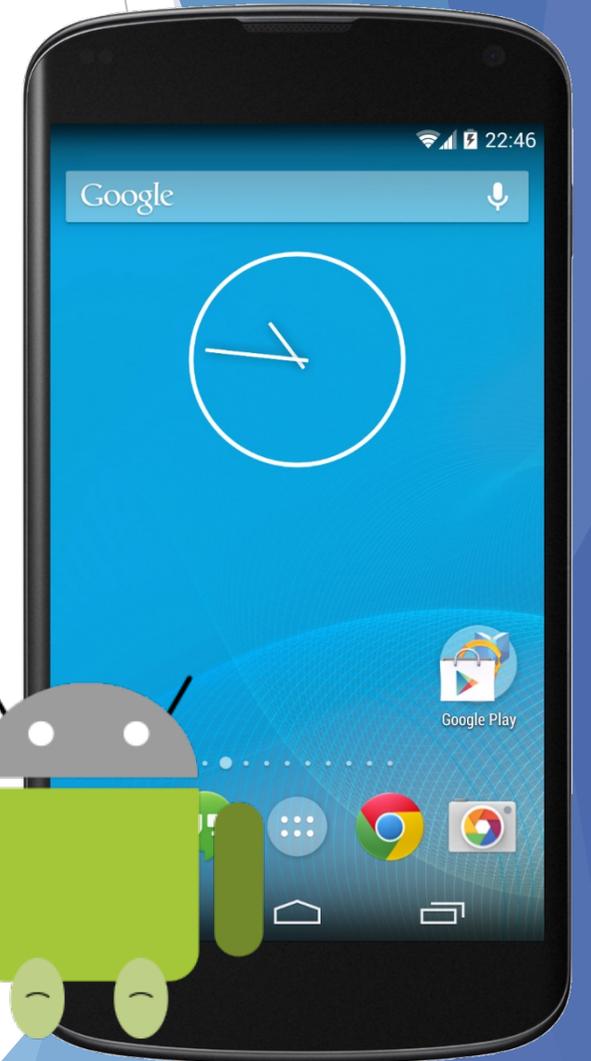
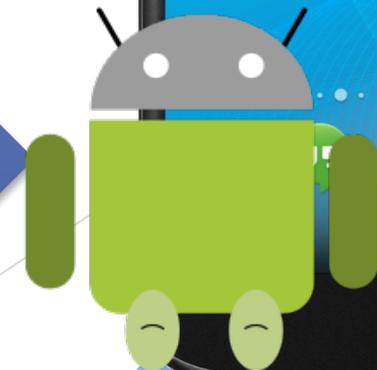
Path Tracking

Optical Flow Sensor - Example

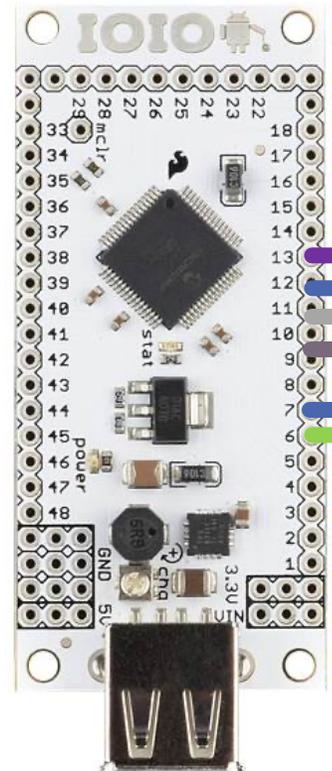


Movement of
Objects

{delta x, y }

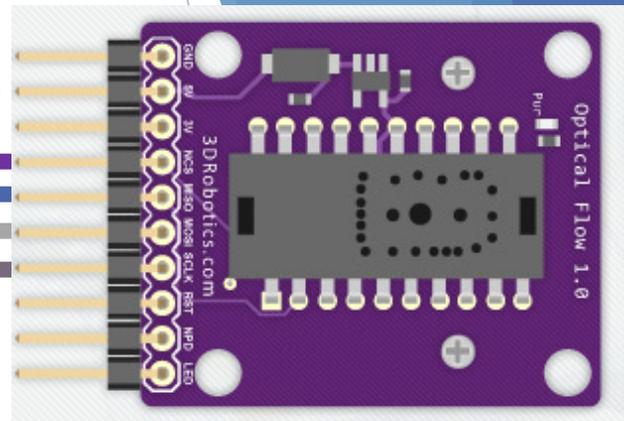


Sensors



SPI

5V
GND



GND 5V



Assumption

Assumption

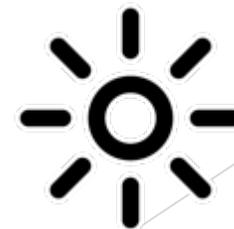
Stable Flying
Height



Best Ground
Surface

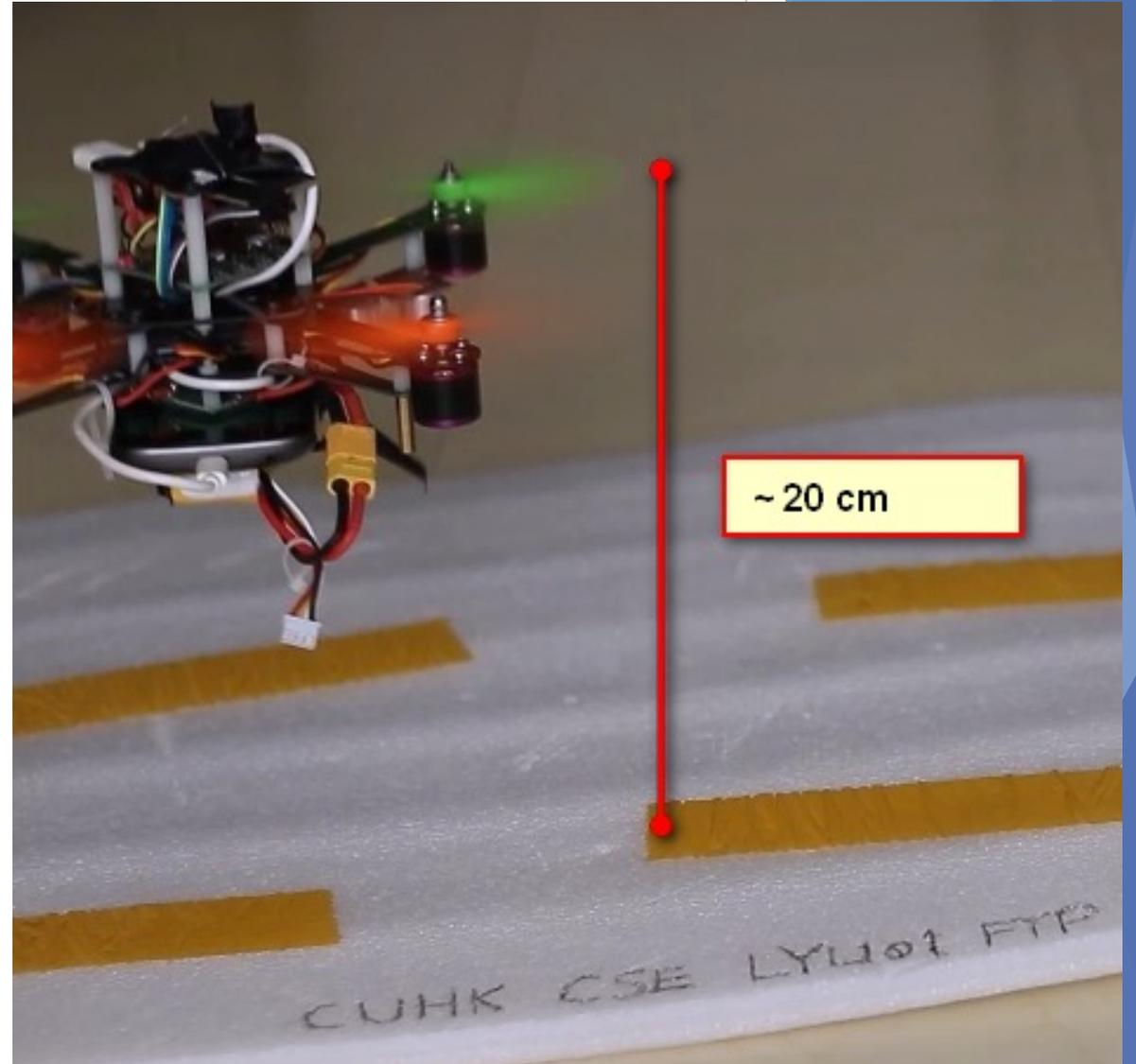


Brightness



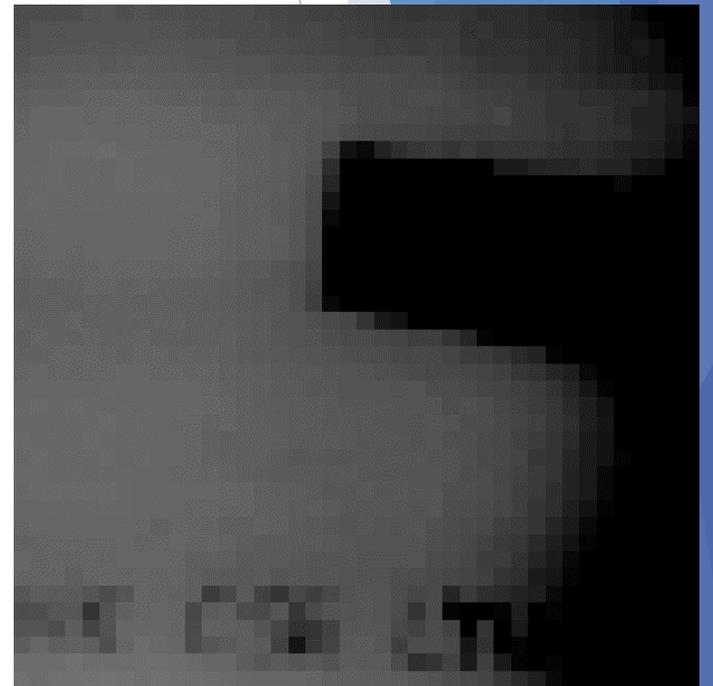
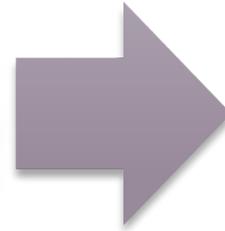
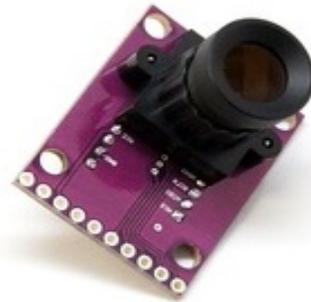
Assumption - Stable Flying Height

- ▶ 15 - 20 cm is preferred



Assumption - Best Ground Surface

- ▶ Detailed surfaces with big objects



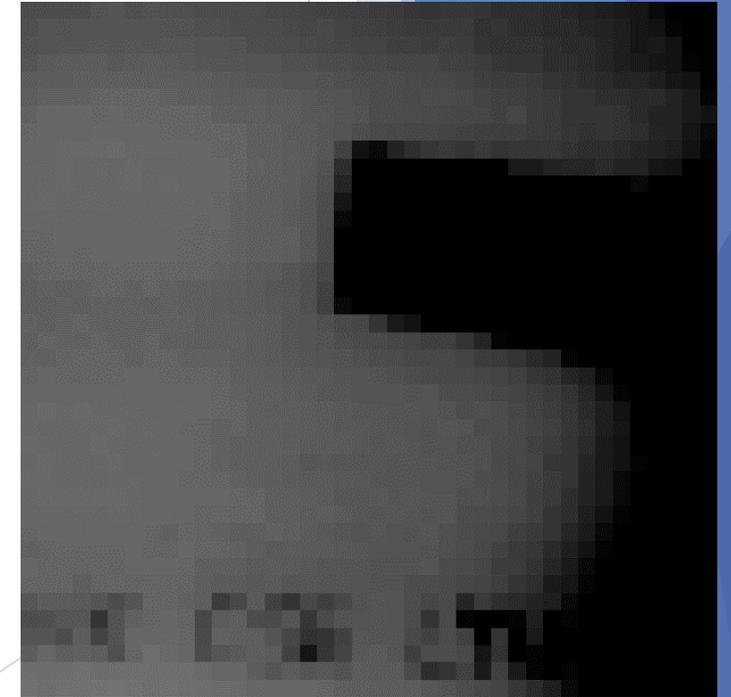
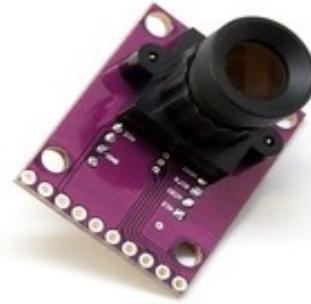
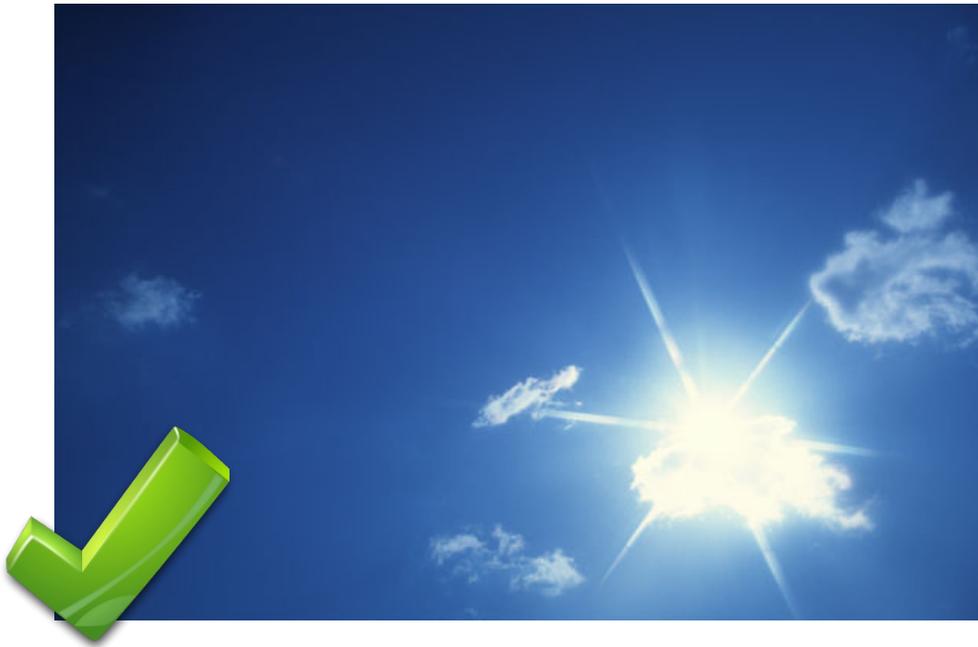
Assumption - Best Ground Surface

- ▶ Normal indoor ground
- ▶ Normal outdoor ground



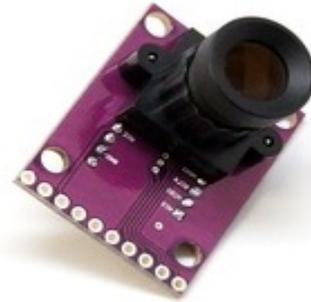
Assumption - Brightness

▶ Daylight



Assumption - Brightness

► Indoor light



Google Play

742 users downloaded

83 / 742

 AndroidCopter (WIP/Preview) 1.11	免費	83 / 742	★ 4.00 / 7	4	2014/11/19	已發佈
---	----	----------	------------	---	------------	-----

★ 4.00 / 7

4-star rating

Future Development

► Application



Goods Delivery of Online Shopping Website



Photo Taking of Emergency Incidents



Security Guardians of Large Area

Future Development

- ▶ Use expensive hardware to improve the performance

Higher Resolution of camera

- Take Clearer Photo and Better Video

Higher Resolution of Optical Flow Sensor

- Achieve Better Stabilization

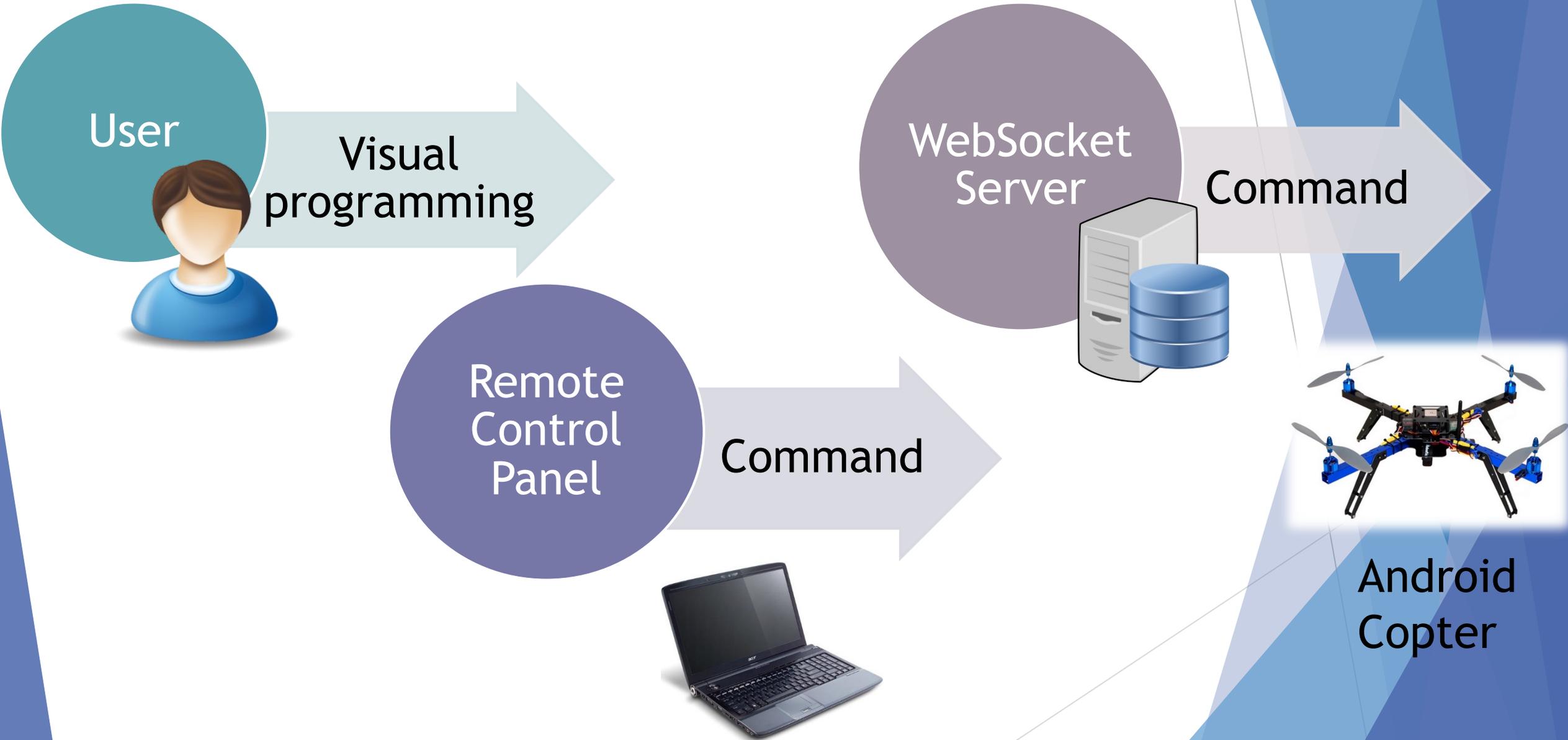
Conclusion

- ▶ Workable in current stage
- ▶ Depends on environments
(Brightness, Ground Surface, Flying Height)

The End

▶ Thank you very much!

Data Flow Diagram



Command Mechanism

