**TASBID Manual**

Author: Yu Li

Contents

[**1.** **Get started** 2](#_Toc73435193)

[**2.** **Scripts protected by IP** 2](#_Toc73435194)

[**3.** **Game Views** 4](#_Toc73435195)

[**4.** **Control Mode Selection** 4](#_Toc73435196)

[**a.** **Keyboard Control** 5](#_Toc73435197)

[**b.** **Hand controller** 5](#_Toc73435198)

[**5.** **Inspection Operation** 6](#_Toc73435199)

[**a.** **Calibration** 6](#_Toc73435200)

[**b.** **Taking off** 7](#_Toc73435201)

[**6.** **Simulation Parameter** 8](#_Toc73435202)

[**a.** **Drone** 8](#_Toc73435203)

[**b.** **Bridge** 9](#_Toc73435204)

[**c.** **Traffic** 9](#_Toc73435205)

[**d.** **Wind** 9](#_Toc73435206)

[**7.** **Tasks and Damages** 10](#_Toc73435207)

[**a.** **Tasks** 10](#_Toc73435208)

[**b.** **Damages** 10](#_Toc73435209)

[**8.** **Game Termination** 11](#_Toc73435210)

[**9.** **In-task Feedback** 11](#_Toc73435211)

[**a.** **Real-time Speed** 11](#_Toc73435212)

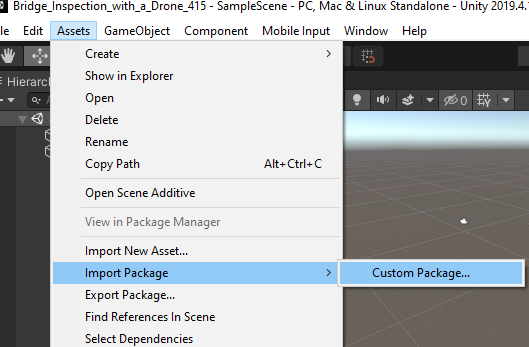
[**b.** **Crash prevention and notice** 12](#_Toc73435213)

[**c.** **On-path warning** 12](#_Toc73435214)

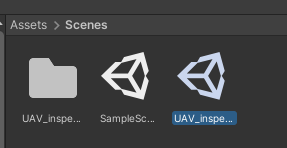
[**10.** **Data Collection** 13](#_Toc73435215)

1. **Get started**

Create a new project in Unity and import this package.

****

Find the inspection scene named “UAV\_inspection” in “Scenes” folder.

****

If the scene lighting looks wired in your Unity, try a lower Unity version. Both Unity version 2019.4.12f1 and 2019.4.15f1 works well for me.

1. **Scripts protected by IP**

You will find errors in this simulation when you import the package, because the drone controller asset adopted in this simulation is not a free asset, it can be bought from: <https://assetstore.unity.com/packages/tools/physics/drone-controller-full-pc-joystick-mobile-111163> . Therefore, some related original source codes are hided in this simulation. Make sure you purchase the drone controller asset to obtain the original code, or you can develop by yourself. All original code inside the class of the following scripts have been hided in this simulation listed as below.

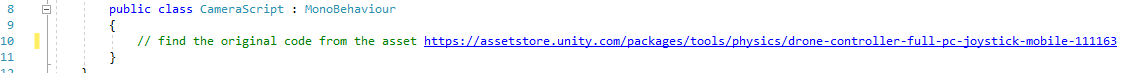
* Assets -> Drones -> DroneController -> Editor -> DroneMovementEditor



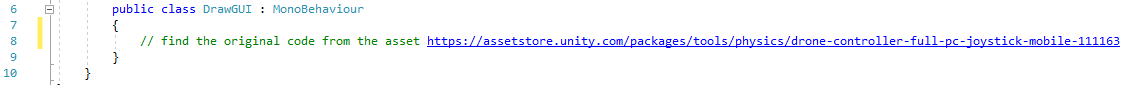
* Assets -> Drones -> DroneController -> Editor -> CameraEditor



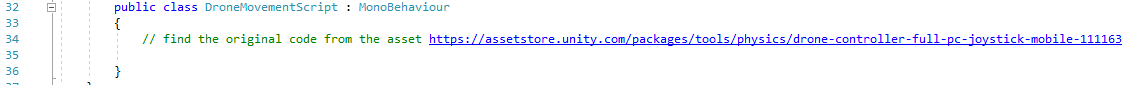
* Assets -> Drones -> DroneController -> Plugins -> CameraScript



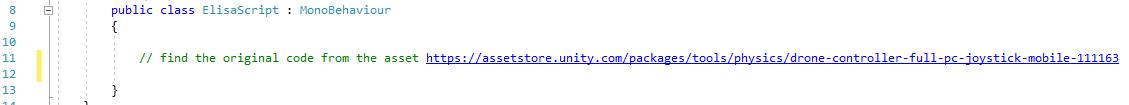
* Assets -> Drones -> DroneController -> Plugins -> DrawGUI



* Assets -> Drones -> DroneController -> Plugins -> DroneMovementScript



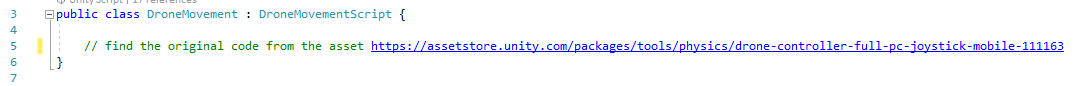
* Assets -> Drones -> DroneController -> Plugins -> ElisaScript



* Assets -> Drones -> DroneController -> Scripts-> DroneCollision

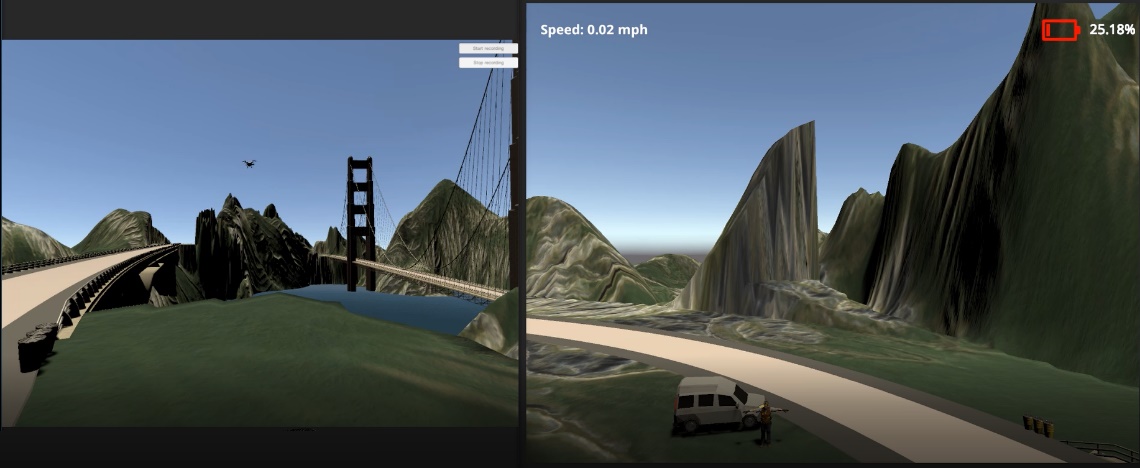


* Assets -> Drones -> DroneController -> Scripts-> DroneMovement



1. **Game Views**

There are two views set in this simulation: operator's view in display 1 (the left part in the following figure) and camera's view in display 2 (the right part in the following figure).

****

The operator’s view can be found by the object of “Operator\_view”. The camera’s view can be found by the object of “uavCAMERAVIEW” under the object of “Drone\_white”. There is another hided view of bird view in the object of “THIRDVIEW” under the object of “Drone\_white”. You can select it and change the display screen if you need it.

1. **Control Mode Selection**

There are two control modes: keyboard and hand controller. Find the mode choices in “Inspector” of the object “Drone\_white”.

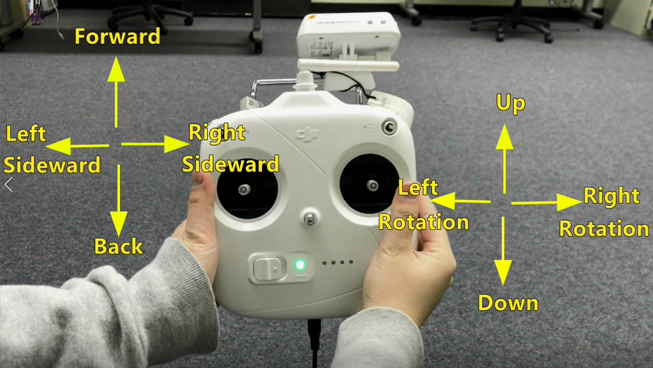
|  |  |
| --- | --- |
|  |  |

* 1. **Keyboard Control**

The control key can be revised here. There are 12 inputs for keyboard controller.

* 1. **Hand controller**

The default joysticks setting is based on a DJ Phantom2 hand controller, which only have 8 inputs (i.e. 4 axis) and shown as below.

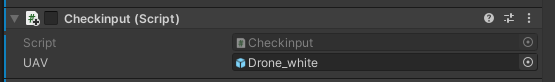


Make sure you modified your “Input Manager” in Unity to keep the same name of Joysticks and the correct axis. “Input manager” can be found from “Edit” -> “Project Settings” -> “Input Manager”

|  |  |
| --- | --- |
|  |  |
|  |  |

The inputs of Joysticks can be revised if you use a different hand controller, which might have more or less inputs.

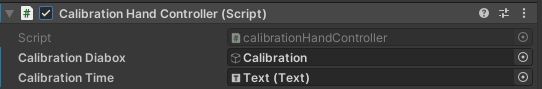
A component named “Checkinput” can selected if you wanna to check the real joystick value. This component is unselected in default.



The connection of DJ Phantom2 hand controller to the Unity can be found here: <https://timebutt.github.io/static/using-a-phantom-dji-controller-in-airsim/>.

1. **Inspection Operation**
   1. **Calibration**

If you select hand controller to operate the UAV, make sure you select the component “Calibration Hand Controller” in “Inspector” of the object “Drone\_white”.



The default calibration time is 10 sec. It can be changed by the variable of “calibrationTime” in the script of “calibrationHandController”.

During the hand controller calibration, push sticks furthest to each direction. Calibration is important for your later controller operation.

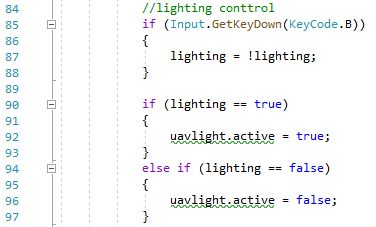
Note: if you use keyboard to control the UAV, make sure you unselect the component “Calibration Hand Controller”.

* 1. **Taking off**

Push “up” sticks (it should be the default key “i” if you use keyboard) for taking off and start the inspection. This can be revised from line 38-58 of the script of “StartFly”:

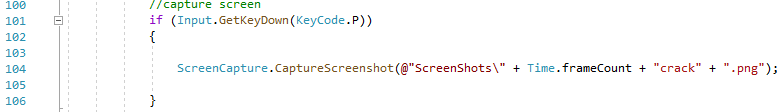
* 1. **Turn on/off the light**

Press keyboard “b” to turn on or turn off the light. The key can be modified from line 84-97in the script of “StartFly”.



* 1. **Screenshot of the crack**

Press keyboard “p” to capture the screenshot. Make sure the Game view is the correct one for screenshot if you have multiple game views. The key can be modified from line 100-106 in the script of “StartFly”.



* 1. **Landing**

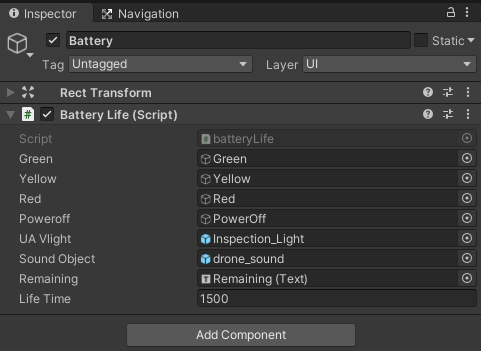
Only when the UAV is landing ahead the operator but not too far away, push “down” stick to land and turn off the UAV. If you land but the UAV is still working, that means you land too far away. The landing location and distance can be modified from line 61-83 in the script of “StartFly”.

1. **Simulation Parameter**
   1. **Drone**

Parameters for modeling the drone include the drone model, mass, maximum forward speed, maximum side-ward speed, movement force, rotation speed, slow-down time, movement sound, the propellers' rotation, battery capacity, and movement types. All these drone parameters (except battery capacity) are based on the original drone controller asset setting.

The object of “Battery” can be found under the object of “Canvas”. The default capacity os 1500sec (i.e. 25min).

The remaining battery level as a percentage of the battery capacity is updated in real-time and displayed at the upper right corner of the camera view. The battery icon is in green color when the remaining power is 70% or higher, yellow if between 30% and 70%, and otherwise in red. Battery capacity can be revised in the script of “batteryLife”.



* 1. **Bridge**

The simulation uses a free Road Architect asset (which can be found from here: <https://github.com/MicroGSD/RoadArchitect> ) to create the bridge models, wherein multiple types of bridges are available for choice. The simulation includes an arch bridge and a suspension bridge to provide trainees different experiences in training and assessment. Learn more from the Road Architect asset if you need to change the bridge type.

* 1. **Traffic**

The traffic is simulated by a free asset (which can be downloaded from: <https://github.com/mchrbn/unity-traffic-simulation> ). Learn more from this traffic simulation if you need to modify the volume and the moving path of the vehicles.

* 1. **Wind**

Only wind under level five of the Beaufort Wind Scale is considered as a possible weather condition for drone inspection in this simulation. Accordingly, three levels of wind are considered in this simulation: light, gentle, and medium. They correspond to the wind speed around 2mph, 11mph and 22mph, and cause the force of 0.12N, 3N and 12N, respectively. The three-dimensional vector can set up the direction of the wind. All parameters can be modified in the script of “windObject”.



1. **Tasks and Damages**
   1. **Tasks**

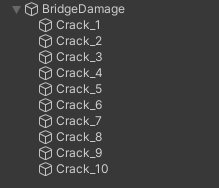
There are totally four inspection tasks in this simulation Details of these task can be found from: <https://sites.google.com/view/yuli1102/projects/bridge-inspection/flight-simulation?authuser=1#h.hr9wan70ijrs>.

The ground truth inspection path is set by Line Renderer in this simulation. They can be found under the object of “Path”. The materials of these lines are set by “Default-Skybox”. Change the material of these lines if you want to show them during the game.

There are four scripts (named “task1”, “task2”…) are set for these four tasks, respectively. These scripts are used for calculating the minimum distance between the UAV to the task ground truth and determine whether the UAV is flying around the ground truth path or not. If you want to modify parameter, make sure you modified the same parameter in all these scripts to keep the constant.

* 1. **Damages**

There are a total of 10 bridge cracks attached on the bridge. They can be found in the object of “BridgeDamage”.



1. **Game Termination**

There are four terminated conditions:

1. Finish all tasks and return to the ground team location.
2. Crash vehicle
3. Crash people
4. Unable return before the power runs out.

The termination condition of (2) and (3) can be modified in the script of “crashWarning”. The condition of (4) can be modified in the script of “batteryLife”.

**Graphical user interface, website

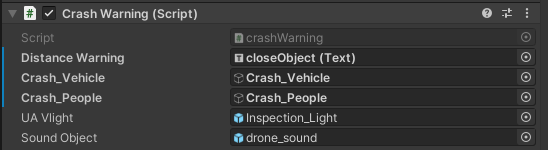
Description automatically generated**

1. **In-task Feedback**
   1. **Real-time Speed**

Besides the real time battery remaining, the drone's speed is always displayed at the upper left corner of the camera view. The object of the real time speed is the object of “RealtimeSpeed” under the object of “Canvas”. The system uses 10 mph as the speed limit for inspection. A warning message will be shown up if the UAV is over speeding. The warning object is “SpeedWarning”, can be found under the object of “Canvas”. Related speed warning can be modified in the script of “speedWarning”.

* 1. **Crash prevention and notice**

To avoid crashes, a warning message appears if the drone senses any object within 2.5m to the center coordinate of the drone. The warning message also appears when the drone crashes into the objects. The related setting can be found in the component of the drone. The crash warning setting can be modified in the script of “crashWarning”.



There is another component named “Crash Performance” for the drone. This is used to record the object that the UAV crashed into.



* 1. **On-path warning**

To encourage the inspector to keep the drone on the reference path, a message will display at the bottom left, reminding the subject to keep the recommended distance if the drone are 8 meters far away from the reference path. The default of this warning can be modified in the script of “pathWarning”, which can be found in the component “Path Warning” of the drone.



1. **Data Collection**

If you want to record the data, press the “start recording” in the right upper conner when you start the unity and before you operate the UAV. The data collected script is revised based on the original drone controller asset script “FlightRecorder”. Some new variables that needed to be collected are added. Therefore, if you need to playback, make sure you delete these new variables or you can revise the code.