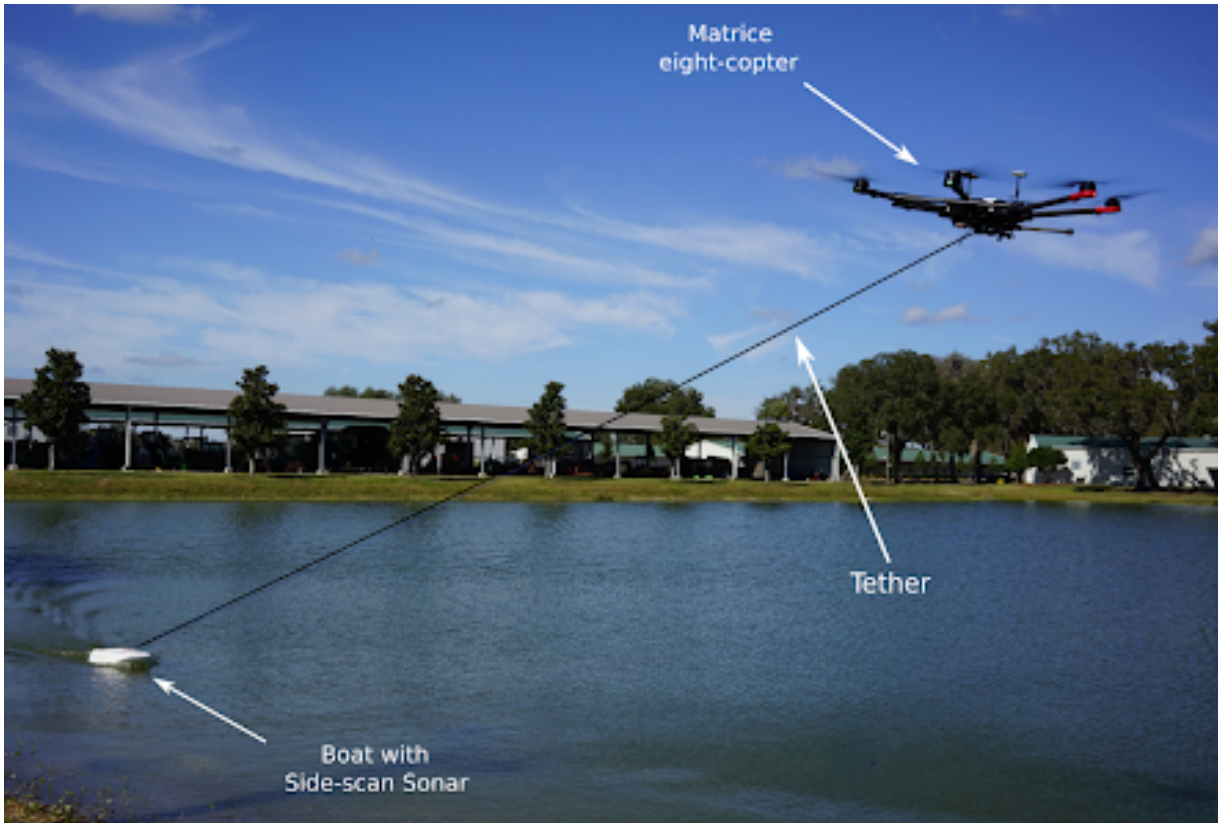
Bathy-drone Path Planning GUI User Guide



Introduction.....2
- Bathydrone system.....2
- Workflow (high level).....2
- What the GUI systems do (explicitly denote the inputs and outputs of the system).....2
- Report issues / contact information.....3
- Contributors and Acknowledgement.....3
- Citations to related publications.....3

Sample Data.....3
Step-by-step description.....3
Demo.....4
Troubleshoot.....9
- Troubleshooting. Possible cases in which GUI could fail.....9
- Point to the directory of the code for each case of failure.....10

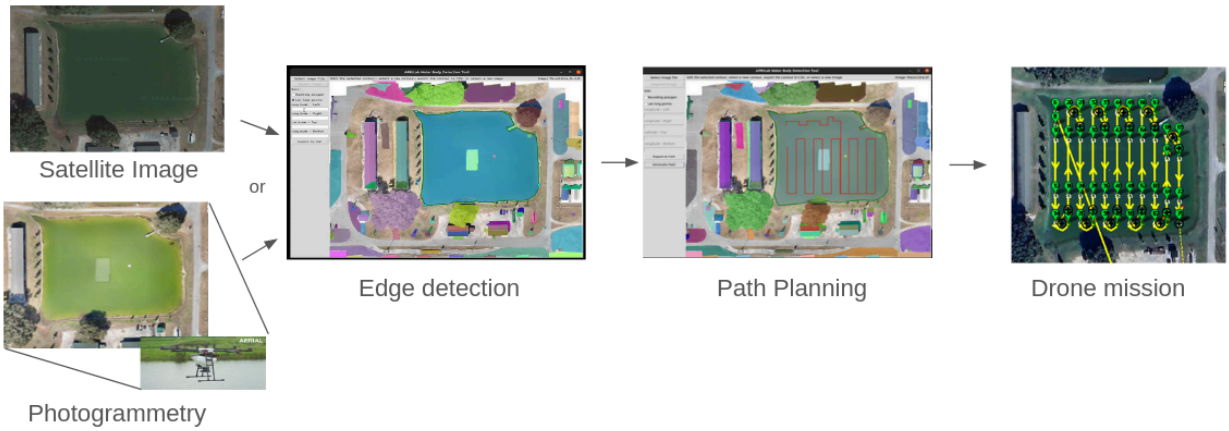
Introduction

● Bathy-drone system

The Bathy-drone is an autonomous platform consisting of a drone towing a boat equipped only with a rudder and a sonar sensor. Unlike traditional systems, the Bathy-drone does not require a boat ramp for deployment. It can fly directly to the survey location to perform a range of missions, including bathymetry mapping, underwater structure inspection, and marine life surveying.

This user manual provides guidance on utilizing the Bathy-drone for autonomous bathymetry survey missions using its dedicated Graphical User Interface (GUI). The GUI integrates advanced AI techniques capable of extracting the boundaries of water bodies from satellite or photogrammetry images. Users can then generate mission command files for the drone through the GUI, leveraging its built-in path planning algorithms to create efficient and precise survey paths.

● Workflow



● What the GUI systems do

Mainly, the GUI offers the following functions:

- Automatically extract the water boundary from a given image (FastSAM algorithm[1])

[1] FastSAM is a relatively lightweight Convolutional Neural Network trained to segment images. It has been applied here to our objective of water body detection, where we have developed a custom Tk-based GUI in Python to integrate the entire workflow. This allows us to manually select the water body segment, enter latitude and longitude information, and even modify the bounding polygons FastSAM generates in case of any detection errors.

- Allow users to edit the boundary that is automatically extracted
- Compute the desired survey path (path planning algorithm developed by UF)
- Generate a mission file for the drone

Report issues / contact information

To report an issue with the tool please email Andres Pulido at andrespulido@ufl.edu.

Contributors and Acknowledgement

We would like to thank Blake Sanders, Nicholas Sardinia, and Sayeed Ali for the development of the GUI, segmentation and path planning algorithms.

Citations to related publications

Journal

A. L. Diaz, A. Ortega, H. Tingle, A. Pulido, O. Cordero, M. Nelson, N. Cocoves, J. Shin, R. Carthy, B. Wilkinson, and P. Ifju, "The Bathy-Drone: An Autonomous Uncrewed Drone-Tethered Sonar System," Drones, vol. 6, no. 10, p. 294, Oct. 2022, doi: 10.3390/drones6100294. [Paper]

Conference

Andres Pulido, Blake Sanders, Nicholas Sardinia, Antonio Diaz, Henry Tingle, Peter Ifju and Jane Shin, "Coverage path planning for a robotics platform equipped with a side-scan sonar sensor for underwater bathymetry mapping" AIAA SciTech 2024 Forum [Conference Paper]

Andres Pulido, Ruoyao Qin, Antonio Diaz, Andrew Ortega, Peter Ifju, Jane Shin, "Time and Cost-Efficient Bathymetric Mapping System using Sparse Point Cloud Generation and Automatic Object Detection ", OCEANS 2022 Conference [Preprint] [Conference Paper]

Andres Pulido, Antonio Diaz, Andrew Ortega, Peter Ifju, Jane Shin, "Modeling, Trajectory Planning, and Control of the Bathy-drone: A Drone Towing a Boat Equipped with Sonar for Bathymetry Mapping ", AIAA SciTech 2023 Forum [Conference paper]

Sample Data

- Sample data is provided as a demo purpose. Users can use this to test run the GUI.

Sample data can be found in the following directory of the repository. The data are `tif` files

- Provide a guideline on what the users would need to get data that's needed to run this GUI.

Step-by-step description

1. Open the program
2. Load the photogrammetry images for survey area
3. Auto-segmentation and user editing
4. Generate a coverage path
5. Export the drone's waypoint file

Demo

- This section includes a very detailed how-to guide to run this GUI using the sample data.

To install the GUI follow the instructions in [the README.md of the code](#).

Once the GUI program is open, select the `'Select image file'` widget located on the top left of the GUI. This will prompt the user to select an image. Select the `'MavicCitra'` image. Once selected, the GUI should appear in a similar figure to that as Fig 1.

Fig 1). Screenshot of FASTSAM GUI appearance after the Lake Citra satellite image is selected.

After the satellite image is imported into the GUI, select the `'Segment Image'` widget located directly below the 'Select image file' to segment objects within the image into different colors.

Once the image segmentation has taken place, click on the lake located within the image to create a contour of the lake. After the contour is created, select the `'Generate Path'` widget located on the bottom of the left most column of the GUI. After both steps have been completed, the GUI should appear like Figure 2.

Fig 2). Screenshot of FASTSAM GUI after the 'Segment Image' and 'Generate Path' widgets have been clicked.

Next, check the `'Lat-long points'` box located on the left most column. This will allow for the entries of each latitude and longitude box to be "un-greyed" allowing for the user to enter the corresponding latitude and longitude values associated with the top, bottom, left, and rightmost points of the lake.

The following values shown in Figure 3 were inputted into the GUI for the bilinear interpolation calculation.

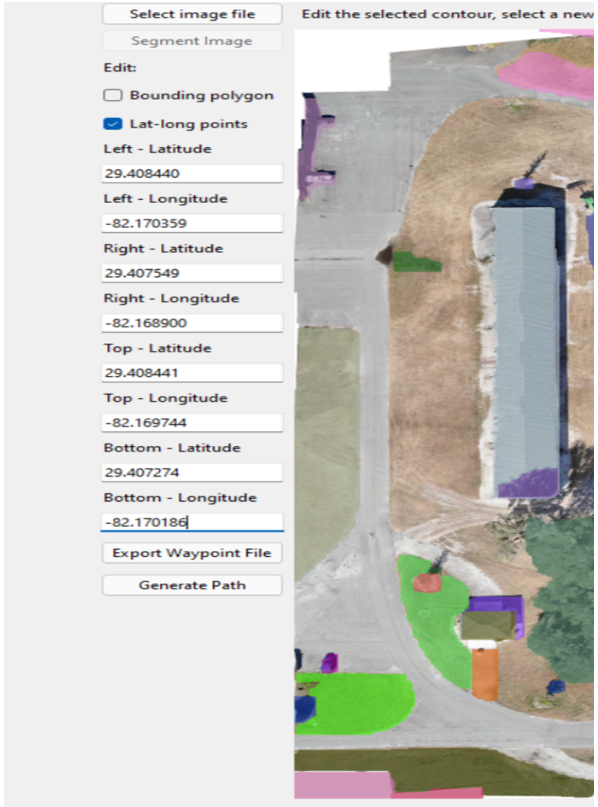


Fig 3). Screenshot of latitude and longitude values inserted into GUI.

After the appropriate latitude and longitude values are inputted into the GUI, the user is to select the `'Export Waypoint File'` widget. This will generate a text file that ends with `'.waypoint'`. The waypoint file can be located within the same directory that the FAST SAM GUI python program is located in.

After the .waypoint file is generated, the user is to open Mission Planner.

The interface for Mission Planner should be identical to that as shown in Figure 4.

Fig 4). Screenshot of Mission Planner user interface when it is initially opened.

Once Mission Planner is open, the user is to select ‘Simulation’ (located on the topmost row) and then select `'Multirotor'`. A pop up will appear asking the user if they would prefer the “Dev” or “Stable” version to be run. The user is to select the “`Stable`” version. After, the Mission Planner interface should appear identical to that of Figure 5.

Fig 5). Screenshot of Mission Planner interface after selecting the stimulation for a multirotor.

Subsequently, the user presses `'Plan'` (located on the topmost row of the user interface) and then the ‘Load File’ button to load the .waypoint file into Mission Planner. Once the .waypoint file has been imported, the user presses `'Write'`. The Mission Planner interface should appear identical to Figure 6.

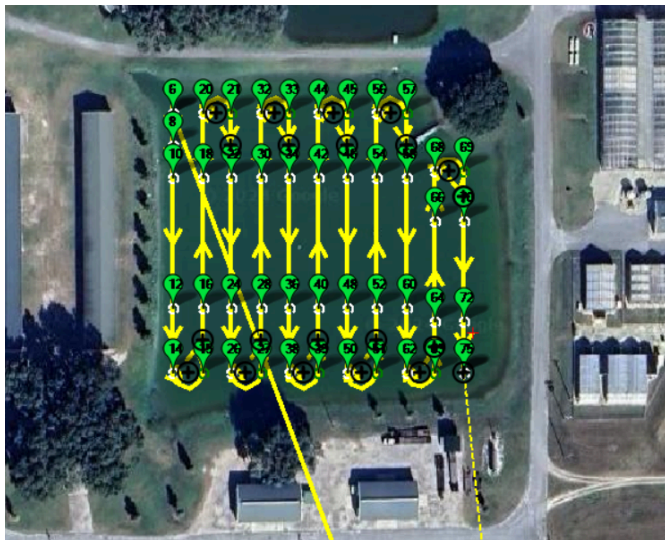


Fig 6). Screen shot of Mission Planner interface after .waypoint file has been loaded and written into Mission Planner.

Next, the user presses `'Data'` (located on the top left corner of the interface) and then the `'Actions'` tab located on the left of the interface (visual provided on Figure 7).

Subsequently, the user clicks the drop-down menu right next to the `'Do Action'` button. The user selects the `'Mission_Start'` option and then the `'Arm/Disarm'` button.

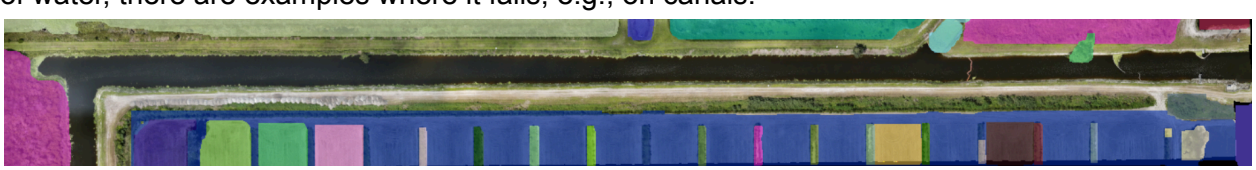
Lastly, the user clicks the ‘Do Action’ button to initiate the simulation.

Fig 8). Screenshot of Mission Planner displaying the 'Actions' tab after 'Data' is pressed.

Troubleshoot

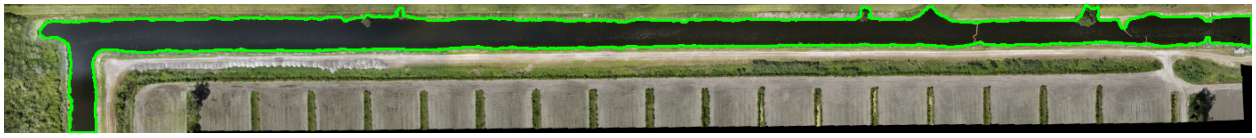
- Troubleshooting: Possible cases that GUI could fail.

In general, the FastSAM approach is preferable as it minimizes human input and integrates the majority of the workflow. However, as FastSAM is not designed specifically for detecting bodies of water, there are examples where it fails, e.g., on canals.



We utilize the OpenCV manual color filter approach in these scenarios. The pipeline begins with a static, full-color image as input. The input in this example is a photogrammetry of the canal. The first step is to apply a box blur to soften edges of the water body. Then, apply a Hue,

Saturation, Value, HSV, color filter to mark water colors as white, all others marked black. Then we find bounding polygons for each grouping of adjacent white-marked pixels and we select the contour with the largest area, resulting in the remaining canal contour.



- Point to the directory of the code for each case of failure

The actual User Interface is located here. Here is where the fastsam code is located. Here is where the path planner is located.