Technical Plan of Operations

VCU Capstone 2025

Aerial Precision

Adil, Beck, Drake, Germain

GEODECISIONS®

A Division of Gannett Fleming

Introduction

The purpose of this document is to outline the process and functionality for creating three dimensional models by using photogrammetry or LiDAR data. This document was commissioned by GeoDecisions through Virginia Commonwealth University's capstone program. The following methods were produced through the research and design of the assigned capstone team.

Section I outlines the 3D modeling process using the ESRI software suite, and specifically SiteScan, through photogrammetry, which stitches georeferenced photographs together in order to create models that can be manipulated. Section II outlines the 3D modeling process using LiDAR data collected from a LiDAR sensor, which uses lasers to transmit data and construct a point cloud based on the receipt of the transmissions after they have been reflected back to the sensor. Section III describes the improvements that could be made to further enhance the models produced or the ease of collecting data.

Screenshots are included to demonstrate how to follow the processes where applicable. This guide aims to display how to go from raw data collection to displaying the data in a viewer. The following sections will first outline the necessary equipment and capabilities and then will proceed step by step through the process until a finished model is completed. Results will vary slightly as custom data will be used each time this process is repeated, but by following the processes and guidelines set forth in this document a working model should be created in each instance.

Section I – Photogrammetry

This section focuses on the hardware and software used to capture and process aerial drone photos into a photogrammetry product compatible with the ArcGIS suite and related modules. Since most of this process involves utilizing existing tools and relies on the user's understanding of ArcGIS, screenshots will describe the bulk of instruction on how to produce photogrammetric meshes.

Hardware

Hardware selection is limited only by the price point of the drone and camera system, but very clean results were achieved with a consumer-grade drone flown by line of sight around the areas of interest. In using a readily available (at time of writing) drone, the **DJI Mavic Air 2** with its built-in **48MP still camera**, it was ensured that similar results would be reproducible with a comparable system by future users. Since the processing software is platform-agnostic, any high quality drone photo sets should provide accurate models, assuming a good GPS lock.

Software Suite

The software for photogrammetry outlined in this document is **ESRI's SiteScan**, a photo-to-model cloud processing tool that takes georeferenced photos as inputs and outputs a three dimensional mesh. This simplifies 3D model generation by reducing the need for powerful local processing, and avoiding projection mapping photo data onto separately generated models. It should be noted that the cost of a SiteScan license is not included in the price breakdown due to it being provided by GeoDecisions free of charge for use by the project team. This cost is likely to be the largest expense, at around $5,000 for a 12 month license, but alternatives exist (e.g. Pix4D) that can accomplish much of the same functionality at a lower price point. The geometry produced by this software is exportable as files or directly to **ArcGIS Pro**, where it can be further refined with plugins and placed on a basemap to provide context to the scan site. From there, scenes can be viewed within **ArcGIS Online**, keeping the process within the ESRI suite wherever possible.

Data Collection

Drone flights should be performed in accordance with all federal, state, and local laws. If in doubt, consult with local authorities and/or a legal representative.

Capturing good photogrammetry data relies on a few factors beyond just the drone and camera. Among these are the ability to receive accurate GPS data on the drone, wind and weather conducive to stable flight, and lighting conditions that produce limited shadows, which might interfere with the edge-finding algorithms. Below are a handful of useful tips to keep in mind when making data collection runs for photogrammetry:

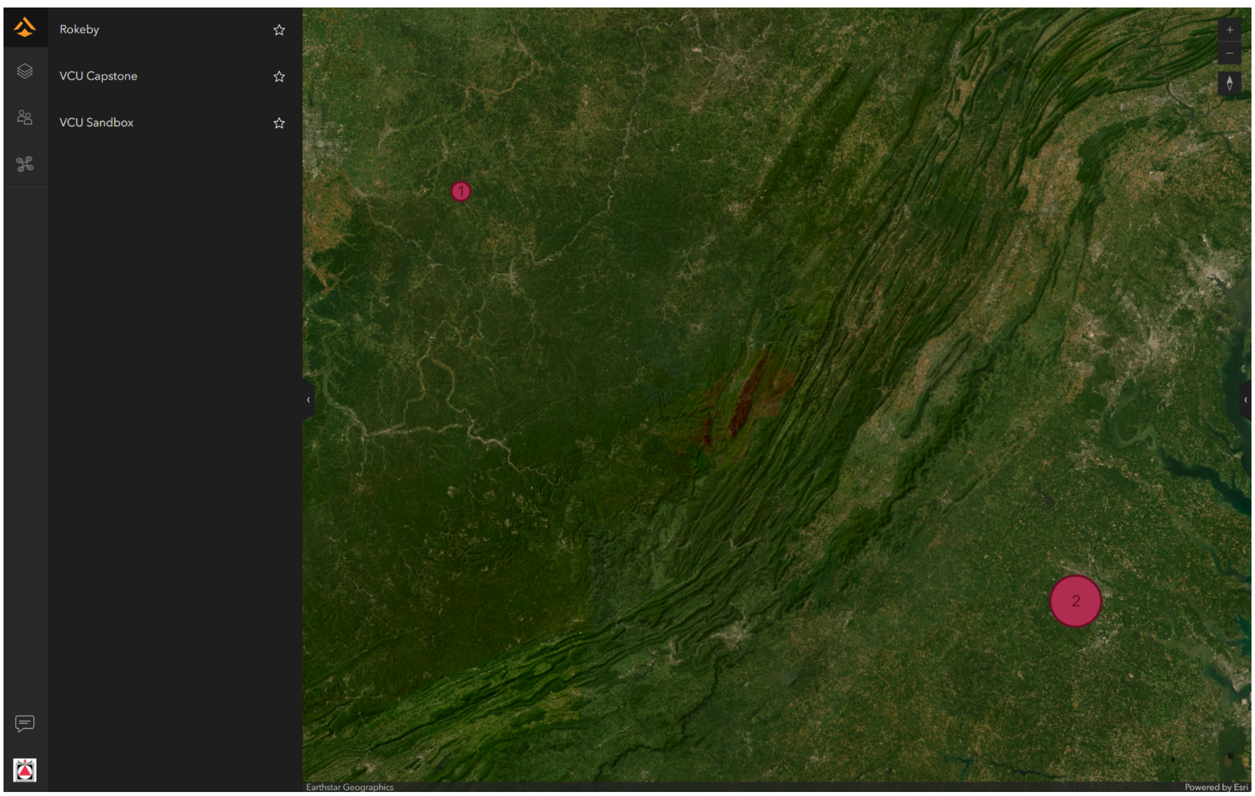
- Proper photo overlap is essential for stitching. For the best results, aim for about 80% overlap between photos.
- Extend the photos taken a good bit beyond the edge of where the model should end. Even if something is included in the pictures, there might not be enough data to include it in the model. To prevent this, try having the photographs go well past the area where the edge of the model should be, and continue that overlap.
- Take both nadir and oblique images. Most of the photos should be done in passes that are straight overhead, but to add depth and additional height information to the model, take images that are angled towards the structures.
- At least three rows of passes should be taken to provide the proper overlap. Aim to take the pictures in a grid pattern with the 80% overlap, moving the drone slightly in the direction of the row, and then when the row is finished, move the drone up or down on that grid and then proceed in the other direction, taking pictures as you go across.
- SiteScan offers a flight tool to plan a flight and fly the drone automatically. This functionality is only offered through the iPad version of their app, so an iPad would be required if desiring to have the flight perform autonomously.

While not implemented here due to a lack of accurate GPS equipment, information about Ground Control Point formats can be found at this link: https://support.esri.com/en-us/knowledge-base/how-to-set-ground-control-points-for-use-in-site-scan-m-000023042

Data Processing

Processing the data involves uploading the photos into the ArcGIS SiteScan module, which processes the photoset into a variety of photogrammetric products that are easily placed on a basemap. This section assumes that photos have already been taken and all data needed is already collected. If this is not done, go back to the data collection section.

SiteScan credentials are required. To begin, log into SiteScan, which will bring you to the main screen.

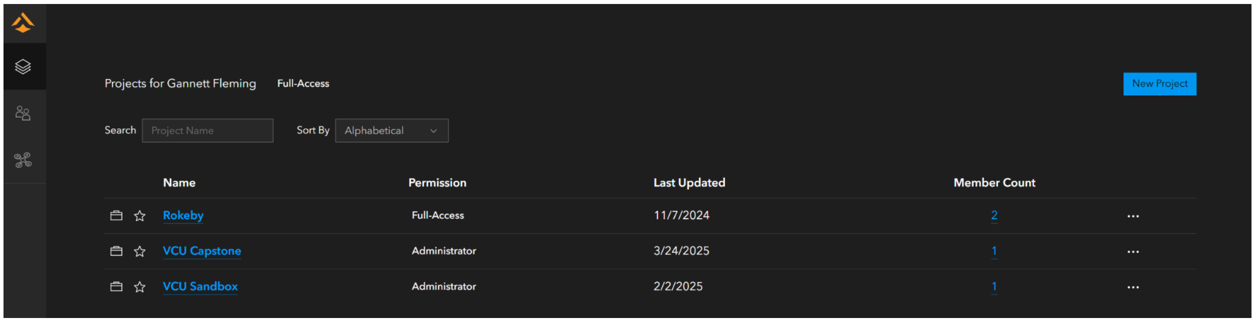


This page lists all of the projects. The icons on the left allow navigation to, in order, the project list, a list of people in the organization, and drone fleet management.

Getting Started: Creating or Selecting a Project

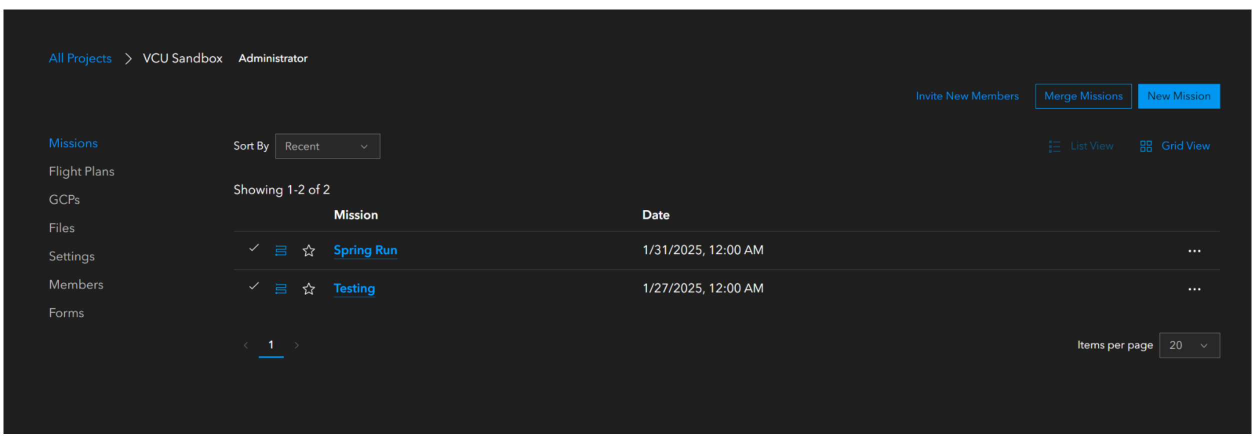
A project serves as a container that holds different missions and scans pertaining to a particular location or work product.

In order to create a new model, select a project to work in or create a new one from the projects page, shown below. If you create a new one, you will need to select a location, coordinate system, and datum according to your preferences, as well as who will have access to it.



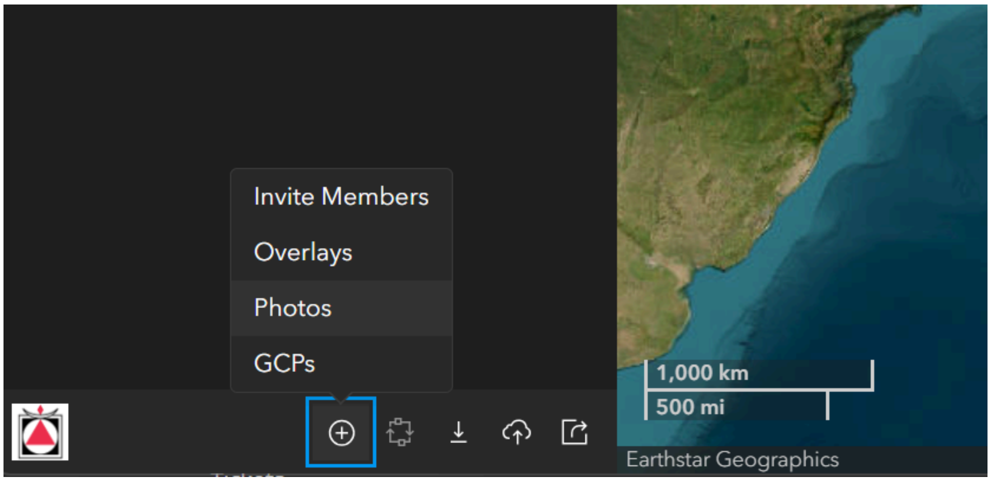
Select a Mission

A mission in SiteScan is the collection of a single set of images to create a model. At this stage, if a flight plan was used, a mission will have already been created and that is the one that should be selected. If the flight was done manually, create a new mission.

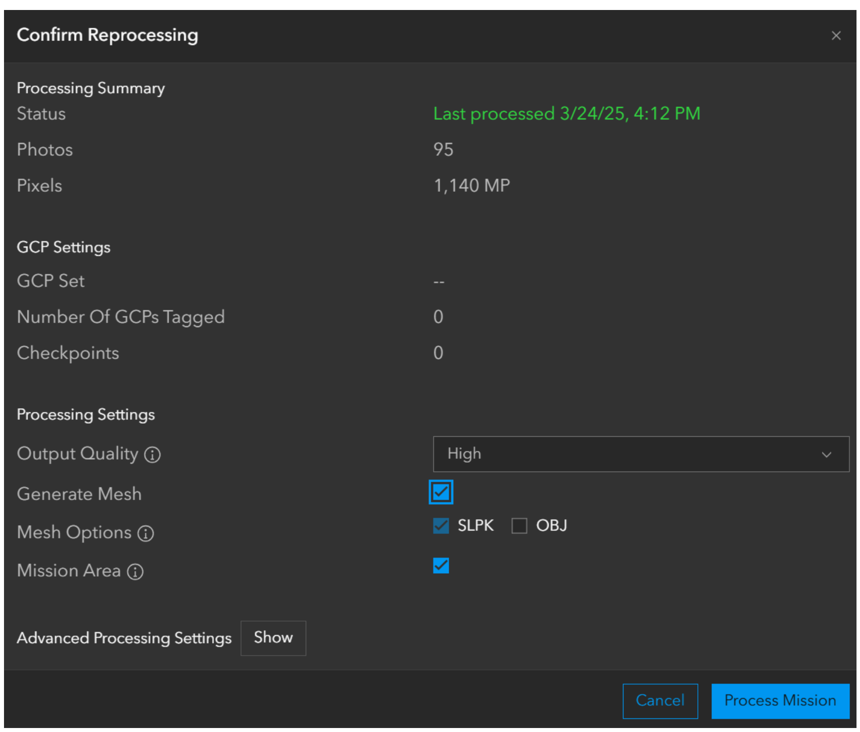


Process the Photos

Give the mission a name and date and upload the photos that were collected when prompted. Photos can also be uploaded using the plus button once the mission has already been created, found in the toolbar along the bottom:



Once photos have been uploaded, they will need to be processed. The processing button is located to the right of the plus button above. The processing screen will require a few inputs before it begins.

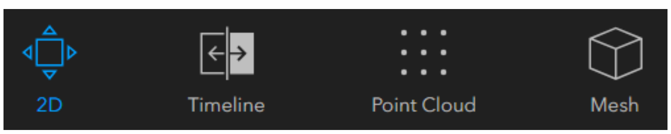


Be sure to click “Generate Mesh” and select which type of file output you would like to create under the mesh options. An SLPK will be sufficient for ArcGIS purposes, but for the model to be used in an external software, an OBJ file might be useful as well.

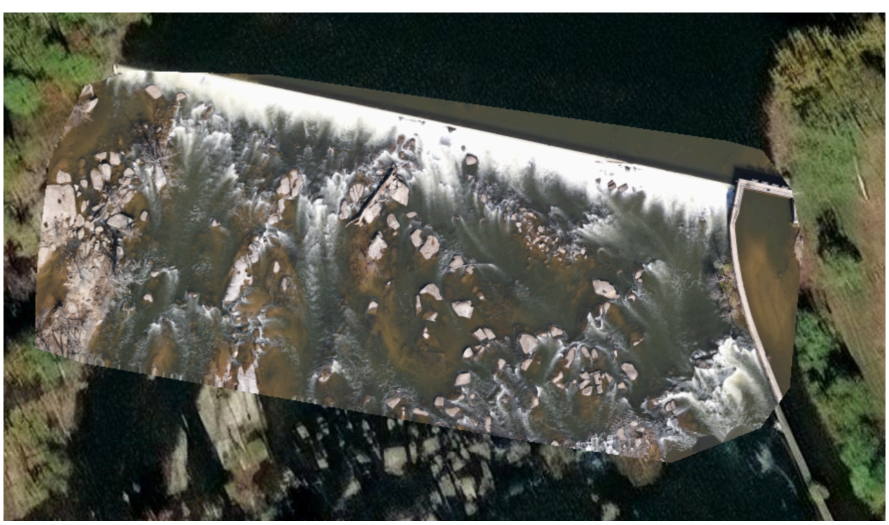
During some instances of processing the mission, if there is a mismatch between the coordinate system of the project and the photos, the processing screen will prompt the user to select an appropriate projection in order for the model to process correctly. In most instances, the SiteScan algorithm is able to auto-detect what that should be.

Processing time should take typically between an hour and two hours, and will fail if the projection system does not match up.

Viewing the Output



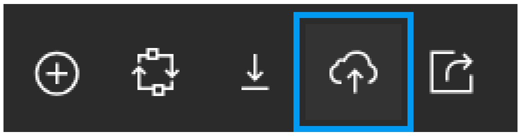
The bar in the top left displays which options may be used to view the data. The 2D option shows an aerial view of the stitched together photographs.



Clicking the timeline option allows for comparison of different models over time, and uses all the missions in the project to create an overlay.

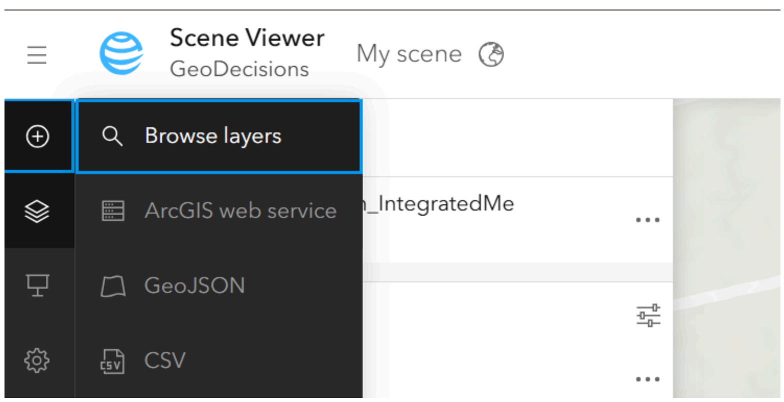
The point cloud option shows a point cloud that has been generated by ESRI's modeling system based on the stitching that was created using the photogrammetry. This view does not fill in the gaps and generally provides insight into the parts of the model which it felt was most accurate. For example, with the pictures taken over water, there is a lot less variability in the surface in the point cloud than in the stitched mesh.

The last option, mesh, shows the fully created 3D model that has been produced with the images. It also allows it to be overlaid upon a map. Adjusting the offset for this model to align with the ground is a critical step to ensure the proper placement on the z-axis. This will also need to be done in ArcGIS if exported there.



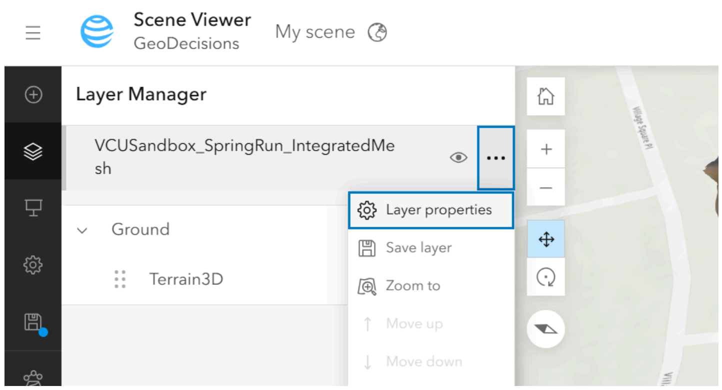
The project may be exported from SiteScan either by downloading the desired file format with the download button in the bottom bar (left of highlighted), or uploaded to ArcGIS Online using the button highlighted above. With the button on the right, a shareable link can be created that allows for a link to be sent that contains the current view so that someone external may view the results of the model.

Once exported into ArcGIS Online, create a new scene and select the desired layers to view in the Scene Viewer by using the add button in the top left.



Add the desired layer. For this example, we will use the Integrated Mesh produced by our SiteScan. Other models, including the point cloud, are available as well.

Once added, if the model sits above or below the surface as in the model above, go into layer properties as shown below.



Choose to configure more under "Absolute Height," and adjust the offset to be flush with the ground level. This will produce a working model in Scene Viewer.

Section II – LiDAR

Hardware

Outlined below are the components and materials used to produce a LiDAR sensor payload that is easily mounted to a drone for aerial data collection. The system was developed to make use of as many off-the-shelf components as possible in order to reduce cost for the proof of concept design. In pursuit of this, a heavy lifting drone that allowed for a sizable payload to be mounted underneath was desired. Selection of the LiDAR sensor was based primarily on weight, as well as footprint, since those criteria determine the flight worthiness of the system as a whole. The choice of microcontroller was driven by data throughput of the LiDAR sensor, ability to operate remotely from user input, and power consumption requirements. Supporting these primary components are a collection of power supplies, voltage regulators, and mounting hardware to secure the payload to the drone.

- **RadioLink M435**
  - Heavy lift drone with 6.6 lbs (3kg) capacity, includes payload plate
  - Available with FPV capabilities built-in and ready to fly
  - Customizable, modular flight controller for added functionality

- **UniTree L1 RM LiDAR**
  - 360° x 90° scan area allows for hemispheric LiDAR capture
  - 8oz (235g) overall weight, 3" (75mm) square footprint
  - High rate capture (~20k points/sec) and ~100ft (30m) range
  - Onboard IMU allows for LiDAR Inertial Odometry (LIO)
  - Includes 'support board' for serial-USB conversion and power supply

- **Raspberry Pi 4 Model B**
  - Open source development tools, support for remote (headless) use
  - Size/weight and power consumption (5V 3A)
  - System familiarity (Linux based, not GPU-driven e.g. Jetson)

- **5V 3.5A 10,000mAh Power Supplies**
  - RPi can be run directly from a power supply with no conversion
  - Simple boost converter (5V 3A > 12V 1A) can be used for LiDAR
  - Size/weight comprise bulk of payload, may limit installation options

- **Mounting Hardware**
  - 6” square 3/16” thick HDPE plate for payload base
  - 3D printed housing for RPi and LiDAR power supply
  - M2.5, M3 screws and standoffs for mounting RPi, LiDAR resp.
  - 3M brand “Industrial Strength” Velcro, for items without thru-holes
  - 8” 50 lbs tensile strength zip ties for backup attachment (fail-proofing)

- **Other Electronics**
  - RadioLink PRM-03 telemetry module for live battery/flight info
  - Wireless keyboard/mouse for remote access to start new scan/reboot
  - HDMI dummy plug provides easy headless operation

Parts List with Links

RadioLink M435 (Mfg website)
    RadioLink PRM-03 telemetry module (Amazon)
    6S LiPo balance plug extension and JST-SH 1.0mm 4-pin leads (Amazon)
Raspberry Pi 4 Model B (Mfg website)
    256GB microSD card (Amazon)
    RPi 4B Heatsink with Fan (Amazon)
    MicroHDMI to HDMI adapter and HDMI dummy plug (Amazon)
    Rii K06 wireless keyboard and trackpad (Amazon)
UniTree L1 RM (Mfg website)
    USB 5V to 12V Boost Converter (Amazon)
    3.5mm x 1.35mm male barrel plug pigtail or 90° plug (Amazon)
12V 4.5A 10,000mAh Power Supply [2 ea.] (Amazon)
12” x 12” 3/16" HDPE plate (Amazon)
M2.5 screws and standoffs, M3 screws and standoffs (Amazon)
8” zip ties, 3M 15 lbs Velcro, Assorted heat shrink tubing (Amazon)
1ft USB to USB-C fast charging/data cable [2 ea.] (Amazon)

Software Installation and Setup

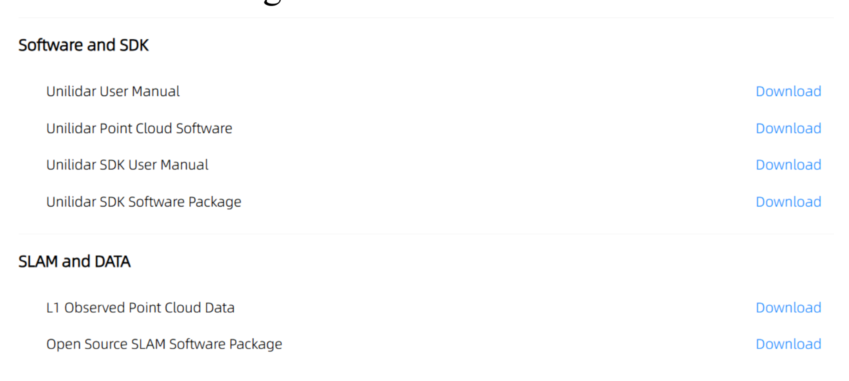
The software kit used for this project was the Unilidar L1 SDK and their implementation of the LIO algorithm. This uses a ROS implementation that runs within a Linux distribution.

For this project specifically, we used Ubuntu Server 20.04 and installed a GUI on that distribution. 20.04 was selected because one of the packages specified in Unilidar's SDK documentation is only compatible with this version of Ubuntu, and at the time of implementation, Server is the only version of this distribution still available within Canonical's downloads.

Once Ubuntu Server was installed, connected to the internet, and a GUI library installed, the two packages provided by Unilidar were built on the command line. Both packages were installed in the exact way specified in the README files within each directory, with the LIO SLAM library being the one that details exactly what needs to be running at which time in order for the program to run correctly.

https://www.unitree.com/download/LiDAR

The packages needed are the "Unilidar SDK Software Package" and the "Open Source SLAM Software Package".



There were two main issues after having installed the packages when trying to build the project. First, the catkin_make command would only run after source devel/setup.bash had been run in each directory, so be sure to run that script before trying to use catkin_make. The other major hiccup happened when using catkin_make to build the project. The Raspberry Pi's memory had a difficult time finishing the build process because of its computing resources. The way to fix this is to use a "-j4" option at the end of the make command that fails, and to create a swapfile within the Ubuntu OS that it can use for additional memory. The software should then build, but it may still take some time.

Once the software has been appropriately set up using the README instructions, the ROS system will need to be run, followed by the SLAM program in different terminal windows. The ROS system can run for the entire duration of the Pi being on, but running and stopping the LIO SLAM program starts and stops the data collection respectively. A viewer is provided to watch from the screen as the LiDAR collects data. Once the LIO program is done, it saves the scans.pcd file in a specified directory. Be sure to copy and rename this file to another location before running another LIO SLAM run, or it will be overwritten and the data lost.

In order to run the software while the drone was in the air, a bash script was developed for this project to start and stop the processes described above, as well as to copy the scans.pcd file to another directory and rename it, and then finally to convert the PCD file to an LAS file for easy input into ArcGIS. Two separate bash scripts were run in different terminals to prevent program interference between the LIO algorithm and the ROS program that powered it. The LIO bash script was started by the ROS bash script, and it also included the logic for the manipulation of the scans.pcd file at the end. Appropriate sleep times for the script were added to ensure proper program end and clean times, and sleep was also used to specify how long the LIO would run.

The bash scripts were set to run at startup every time the GUI started up. Some issues arose with a headless implementation, so the Pi needs to be configured where the display output is active, either through specifications in the hardware or an adapter for the HDMI that tricks it into thinking it is outputting to the HDMI. The renamed scans.pcd's converted LAS file could then be extracted for use within ArcGIS once the algorithm had finished running and the specified time had elapsed.

Assembly of the Sensor Payload

The two main components of the sensor payload that will determine overall layout are the LiDAR and the 3D printed enclosure for the RPi. Cabling and the boost converter are critical to keep clear of the rotors, but otherwise can be routed and secured where needed.

Developing the enclosure was done in Blender, using publicly downloadable files for the RPi 4B and L1 LiDAR to ensure accurate placement of mounting holes. Ventilation holes were added around the perimeter of the box to allow the heatsink fan to force warm air out into the airstream, but placed to allow voltage regulators to be Velcroed onto clean interior walls of the enclosure. Access was included for the RPi's USB-C power port, USB ports (LiDAR data, keyboard and mouse), and microHDMI port (microHDMI adapter to HDMI dummy plug), with room to spare since dimensions vary across various plugs/manufacturers. Another hole is present to allow the power, data, and USB-C cables to reach the LiDAR support board, and a final hole near the top edge of the enclosure allows for a fool-proofing zip tie to be added once installed on the HDPE plate. The corners are reinforced and a tight fit (3mm) hole is provided for attaching the enclosure to the bottom of the HDPE plate. The model produced measures 110mm x 85mm x 50mm and is provided as an addendum.

1. **Layout components on HDPE plate** where they make sense, starting with the LiDAR to make sure it has the best field of view (at least forward and down unobstructed). The L1 LiDAR ignores all objects within 2" (50mm) of the center of the sensor, but should be kept as clear of the ground as possible. Next place the enclosure where it re-balances the plate back toward the center (assuming the LiDAR isn't centered on the plate), generally behind the LiDAR. Be sure that all cables can be routed where needed, plug them in to verify flight setup will fit and not strain cables/ports. Cut the plate to size.
    a. **Build "blinders" to block the LiDAR**'s view of the drone. If portions of the drone are visible to the sensor and greater than 2" from the sensor, they will be recorded as part of the point cloud, which can complicate post-processing. Ideally, the blinders would be secured with screws and glue to keep them as solid as possible.

2. **Mark component mounting hole locations and drill** for close fit with M3 hardware (3.2mm). This can be undersized to 3mm since HDPE will self-thread with steel screws and a tight fit helps prevent vibration. Other holes can be drilled for zip ties: the enclosure zip tie, the four payload mounting zip ties at each corner of the plate, and any additional ones that are needed to secure the plate to the drone. Drill bits for plastic help prevent melting and tearing holes, and a sharp blade will clean up edges and holes very neatly with some practice. Included in the 3D files is a “lid” that can also act as a template for component mounting hole placement, since it was originally designed to support the LiDAR on top of the enclosure.

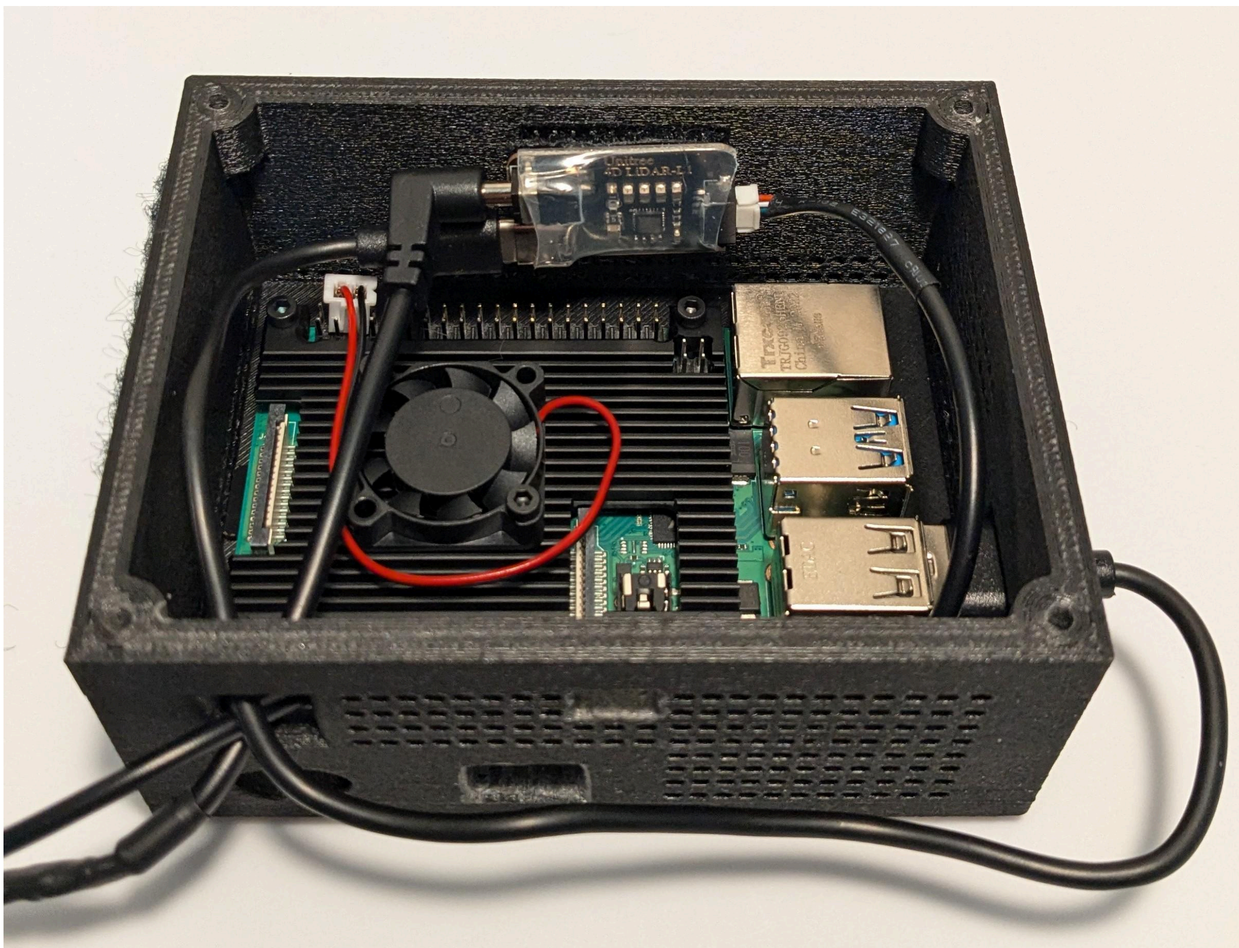
3. **Mark payload mounting hole locations and drill** to provide a way to bolt the HDPE plate to the drone’s carbon fiber load plate, as well as standoffs to prevent excessive flexing of the HDPE. The holes for mounting the payload need to be 1⅛” from the centerline running from front to back in order to line up with the most accessible slots on the load plate. This is important to the order of operations in payload installation. The standoff holes must lie directly below each of the 4 arms with motors on them, and in a position such that they prevent the battery packs from sliding forward or aft out of their straps. For the provided hardware list, this equates to about 6” apart, 1½” in from either edge along the diagonals (draw lines connecting opposite corners of the HDPE plate), assuming that the plate is mounted centered beneath the drone. **It is worth test fitting and judging the best location during payload installation.**

4. **Install RPi into the enclosure** using washers or standoffs to ensure height alignment with USB, power, and display ports. Use M2.5 screws through the heatsink, RPi, washers, and out the bottom of the enclosure. Secure them with nuts to provide good contact with the heatsink.

5. **Prep 12V boost converter** by cutting off the barrel connector included and soldering on the correct 3.5mm x 1.35mm barrel connector for the LiDAR support board. Similarly, shorten the USB side of the converter to prevent loose wires catching the rotors. Insulate/reinforce with heat shrink.

6. **Velcro LiDAR support board** inside enclosure onto interior wall opposite the power and HDMI ports. This way cables can be routed with maximum bend radius and extra length can be tucked inside once mounted to the plate.

7. **Route and plug in all cables to support board** (USB-C, 3.5mm x 1.35mm power, LiDAR 4-pin), verify position of 12V boost converter and velcro in place. This can be reinforced with an additional zip tie if ¼" holes are added to either side of the mounting location.



Note hole placement, L-R from top left: LiDAR cable access, zip tie mounting point, oval-shaped power port access, microHDMI access. Rough edges indicate iterative development cycle.

8. **Mount LiDAR to HDPE** with M3 screws and plastic washers, making sure to put relatively even torque on each screw and not overtightening (hand tight with the long end of an allen wrench).

9. **Mount enclosure to HDPE** with M3 screws and plastic washers, threading slowly in to make sure that the screw threads bite into the print lines of the enclosure. While this is a repeatable way to secure the enclosure, the addition of threaded inserts in the 3D print is likely to be superior. Use a zip tie through the appropriate hole in the enclosure and the hole provided in step 2 so that there is a backup should the print fail and the mounting screws come out. This zip tie can also be used to secure loose cables.

The sensor payload is now complete, and can be powered on and used as a standalone unit for testing, or for interior/small scale scans.

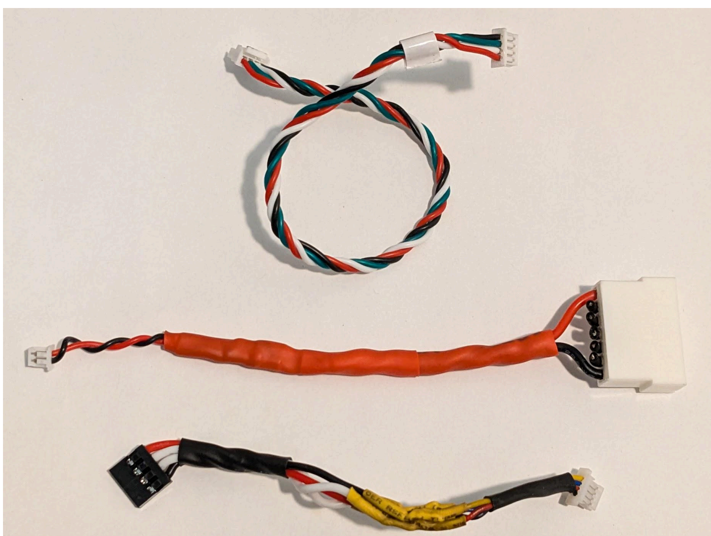
Drone and Transmitter Configuration

Successful flight can be accomplished out of the box with the M435 drone, but it is recommended to take some additional steps to ensure a smooth flight experience and support data collection operations. RadioLink provides a variety of drivers and a flight controller config tool (Mission Planner) based on the specific hardware being used, which provides a very detailed interface for modifying the drone's settings. Similarly, the transmitter will need to be set up to match the settings selected on the flight controller with Mission Planner.

1. **Reset transmitter to clear current settings** and prevent overlapping or overwriting controls when making the changes below.
2. **Assign Channel 9 to Variable Slider D**, which puts the FPV camera tilt access at the right index finger slider (on the bottom of the transmitter).
3. **Set timers for 10 and 15 minutes, assigning them to Switch G** (3 pos.) so that one direction starts both timers, and the other direction resets them, with the center position unassigned to let the timers run. This will be the collection timer and the total flight timer, respectively.
4. **In SYSTEM settings, ensure that the data receive mode is set to SBUS** and that the flight controller's output mode matches. The transmitter emits a colored light based on the transmit mode to help make this clear.
5. **With no battery connected, plug the flight controller into a PC** with a USB-C cable, and open Mission Planner. Connect to the proper COM port after the drivers install (you may need to manually install them from the RadioLink website). At this point, **run through the initial configuration tool**, which calibrates the drone's gyros/accelerometers and sets the output range of the transmitter.
6. **In Mission Planner, set the flight attitudes** (there are 6 of them for the CrossRace flight controller) in a way that makes sense for the data collection runs planned. Certain modes require GPS connectivity and will cause the drone not to arm or to return to landing (RTL) if the drone has no GPS lock. For this reason, it is recommended that stabilization (STABL) be used as the startup/default flight mode. Read linked manuals for details about flight attitudes.

7. **With Mission Planner still open and showing flight attitudes, set the same attitudes to match on the transmitter**. Switches C (3 pos.) and D (2 pos.) control these flight attitudes, with the SwD acting to toggle SwC between two modes, allowing six total attitudes. Ensure that the PWM output of the transmitter matches each mode in Mission Planner; as switch positions are changed on the transmitter, the highlighted attitude should change (and match!) in Mission planner. Modify PWM values if any mismatches occur.
8. **In Mission Planner, set RTL parameters to safe values**. If the scan location is known, a “home base” location can be set as Waypoint 0 in the ‘Plan’ tab so that a safe location can be established as the landing point. If this information is not set, there is a risk that the drone will immediately land upon loss of connection with the transmitter, or if RTL is inadvertently activated (i.e. loss of GPS lock, nudging switches, etc).
9. **Finally, reverse Channel 7 so that the Arm switch (SwA) is armed when pushed up**. This should complete the basic setup of the flight controller and transmitter, but additional customization is also possible. Refer to linked manuals for more information.

To enable telemetry a PRM-03 module must be installed on the drone and connected to the flight controller, battery balance plug, and receiver. This will involve soldering/heat shrinking a number of cables to make the connections.



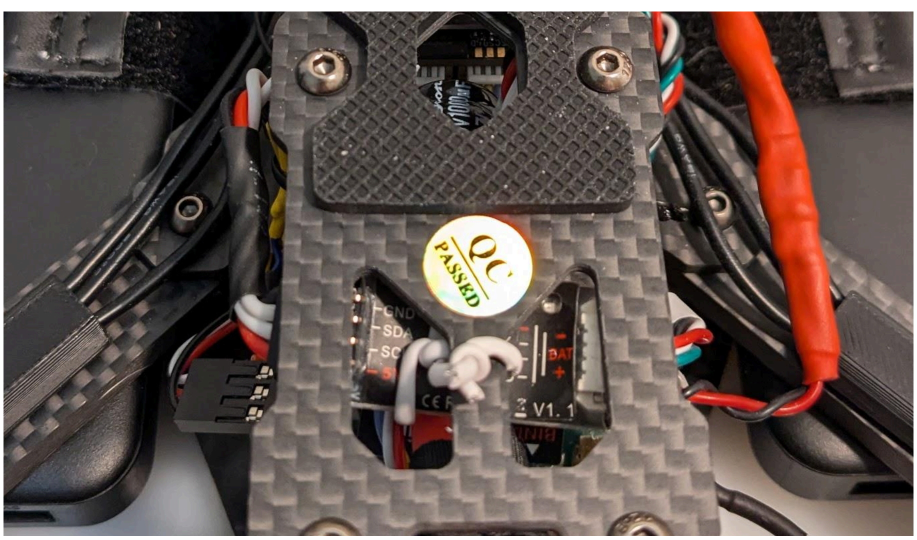
Cables needed to connect PRM-03 telemetry module to CrossRace flight controller

1. **Connect the flight controller's TELEM1 port to the PRM**'s 4-pin RX/TX port with the provided cable. Secure extra length with a twist tie or cut and resolder to an appropriate length. *Top cable in photo above.*

2. **To connect the battery voltage port**, solder the provided 2-pin pigtail to the first and last leads on the 6S battery lead extension, respecting polarity. This will provide the telemetry module with the total battery voltage, as opposed to individual cell voltages. Reinforce this cable with extra heat shrink as the pigtail's wires are quite thin. *Middle cable.*

3. **To connect the PRM-03 to the receiver**, use the 4-pin female header cable, and solder on a 4-pin JST-SH 1.0mm male plug to the other end, double checking which data wire goes where (SDA, SCA) before making connections and heat shrinking them. *Bottom cable.*

4. **Cables should be as short as needed**, and kept twisted to the extent possible in order to reduce interference on small data lines running next to high current motors. The telemetry module can be tucked into the drone's chassis and secured with a patch of Velcro and a twist tie as backup.



PRM-03 installed at the rear of the airframe

At this point, the drone and transmitter should be properly set up and ready to fly with live telemetry (battery voltage, roll/pitch/yaw, heading, and GPS data) provided to the transmitter. There is a great deal of customization possible with additional parameters (i.e. throttle curves for precision flying) or extra modules, should that be desired. Due to the intricacies of various flight controllers, Mission Planner parameters, and somewhat limited documentation available, those choices will be left to future users.

It is recommended to flight test the drone prior to payload installation in order to be sure that settings work as intended, and that flight is controllable when in proximity to objects being scanned.

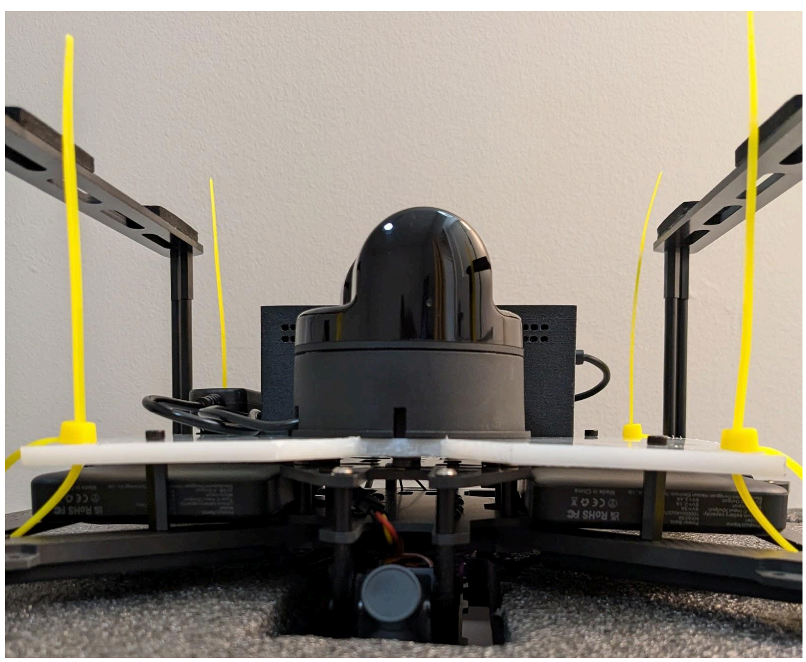
Payload Installation

The guiding principle for mounting important objects to fast moving things is redundancy ad nauseum. In the case of this proof of concept, the preferred order of operations is screws, zip ties, and Velcro to maximize solid points of contact, the ability to pre-tension and absorb stress, and relative speed of deployment. This is the area most ripe for improvements, and one where the addition of simple 3D printed brackets or bracing elements would see great benefits in reducing vibration and providing precise, repeatable positioning.

Before installation, consider the overall height of the payload, and whether additional clearance is needed. With the M435 drone and L1 LiDAR, 20mm M3 standoffs are added to increase the length of the landing legs such that the LiDAR has more than an inch of clearance on flat ground. Heat shrink is added to the front legs in order to aid orientation when in flight.

1. **Mount the 5V power packs** securely to the horizontal arms of the drone's frame using a pair of straps from the drone kit. It is important that the straps be thread through the drone's load plate and around the arms such that the metal of the strap is on the top side of the power pack. This way the payload can mount as flush as possible below them.

2. **Loosely attach payload with zip ties** through each of the four corner holes. Tighten opposite corners alternately to ensure the plate is as centered as possible, adjusting to also align the payload mounting holes in the slots as pictured above. Based on flexing, install appropriate standoffs and washers.



3. **Install payload mounting screws with washers** to secure HDPE plate to drone's payload plate. Expedient washers were produced from scrap HDPE which provided multiple benefits: availability, reduced wear on power pack straps compared to metal washers, and post-tensioning of same straps to further lock power packs into place.

4. **Tighten zip ties to reasonable tension** and trim loose ends flush to avoid snags. Route cables and install zip tie on enclosure (optionally using the zip tie to further secure cable routing) to provide secondary means of attachment. The LiDAR is considered secure with screws since the mounting holes are threaded directly into the unit, and HPDE is unlikely to fail or allow the screws to pull through during normal operation.

Data Collection

Drone flights should be performed in accordance with all federal, state, and local laws. If in doubt, consult with local authorities and/or a legal representative.

As configured, the M435 has a loaded flight time of about 15 minutes, but that was not verified with any endurance testing. Flight time will vary depending on the size of the site, but a casual walking speed provides enough point density for larger objects and surfaces to give models workable detail.

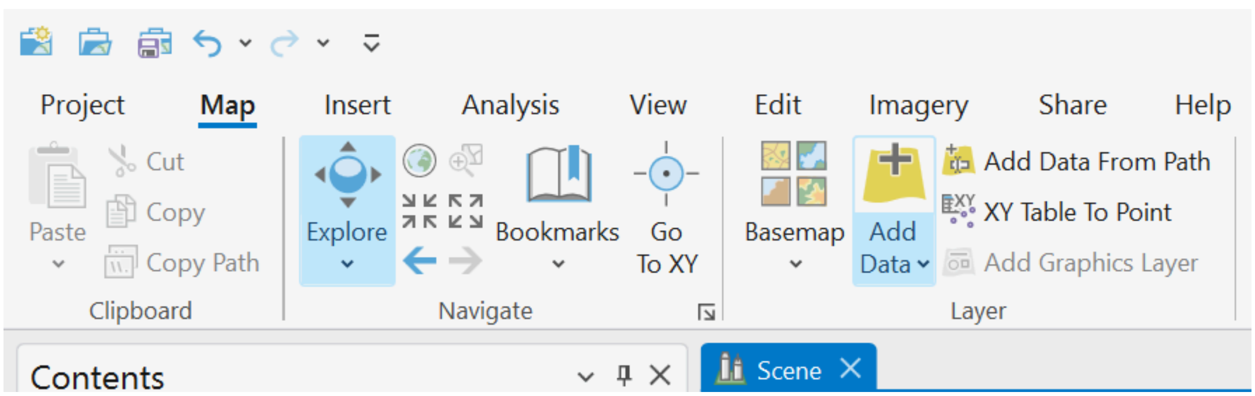
A pre-flight checklist is provided below to assist in data collection startup:

1. **Install propellers**, respecting the rotation direction of the motors.
2. **Install and secure payload** ensuring to route wires clear of the propellers.
3. **Clear debris from the launch area** to prevent anything striking the LiDAR or propellers on takeoff/landing.
4. **Mount LiPo battery** on top of drone with straps, but ***do not plug it in yet***
5. **Turn on transmitter** and confirm switches are in 'SAFE' position, SwA is down (motors disarmed), Flight Attitude set to Stabilize (or other non-GPS).
6. **Power LiDAR on** and confirm spinning with LEDs to indicate scanning.
7. **Power Raspberry Pi on** and perform following steps during startup (1 min).
8. **Verify propeller clearance** and cable routing, ensure launch area is clear.
9. **Plug in LiPo** to power the drone on. Once startup is complete, flip SwG to start the 10 and 15 minute timers to help track flight time and battery status.
10. **Arm motors with SwA** and begin data collection, keeping the drone level and pointing toward the object/area of interest, moving at a walking pace to achieve best results. Telemetry can be used to track battery levels and timers can be adjusted to reflect programmed scan time.

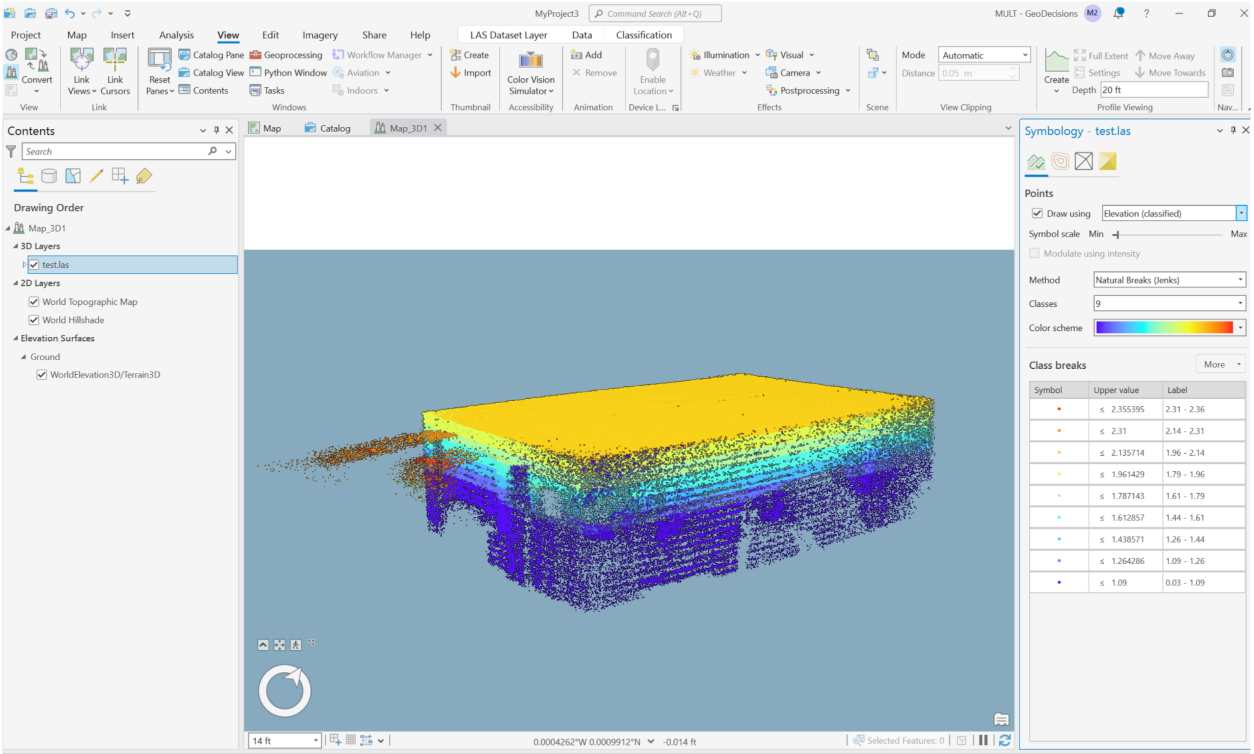
Data Processing

Offloading data from the Raspberry Pi is as simple as plugging in a flash drive and copying the .las (and/or .pcd, point cloud) files to a computer for upload to ArcGIS. While the data conversion from .pcd to .las is handled easily by the RPi, and theoretically the datasets could be uploaded using a Wi-Fi enabled RPi, it's best to use a PC with a graphics card to view and interact with large point clouds. ArcGIS natively accepts the .las files, and uploading them is a simple drag and drop operation.

The best way we found to display the point clouds within ArcGIS was to use a 3D Local Scene. After opening ArcGIS Pro, create a local scene. Use the "Add Data" tool to select the LAS file from your computer and add it to the scene.



The LiDAR model should be interpreted automatically as a LiDAR point cloud and will appear in the scene in 3D. Make sure it is a local scene and not a standard map or global scene. In order to position it properly, the model will need to be scaled and translated to be properly placed within its geographic placement.



Once uploaded to ArcGIS and turned into a model, positioning can be done using an existing photogrammetry model and any defining features of the point cloud as a reference. Unfortunately the M435 doesn't provide accurate enough GPS coordinates when it records flight data, and there's no real way to link the point cloud to GPS data in-flight. This is an area for improvement, and one where a combined photogrammetry/LiDAR payload would be of great benefit.

Section III – Improvements

While the processes described above will yield very functional results, it is primarily a proof of concept for an off-the-shelf photogrammetry and LiDAR scanning system that outputs easily integrated 3D models. Given that a large enough budget can make the problems of this project trivial, the development cycle identified a few areas as ripe for improvement within the limits already established. Among these are streamlining LiDAR data collection, improvements to the drone/payload design, and providing base stations for georeferencing.

During our tests over water for the dams, we found that the model has a difficult time producing accurate surfaces over the water due to its inconsistent visual components. Structures that were still visible and consistent below the surface stitched well and produced accurate results, but the water itself did not do very well. This was an instance where the point cloud was more accurate than the mesh model, as the point cloud helped mitigate the coarse stitching by throwing out portions of the model that were irregular.

Ground control points were something that was targeted but were not included as part of the final design due to time and resource constraints. The project is going to be continued this summer by a VCU undergraduate research project that will add more of this functionality and understanding to this project, but was not something that was integrated into any of the models produced during the course of this project. GCPs would add very accurate geolocation data to ensure the models are placed in the exact location through corrected GPS coordinates.

On the LiDAR drone, some improvements that could be made would involve the interface for starting and stopping the LiDAR collection process, providing an interface for seeing or controlling the collection during flight, and automated flight processes. These attributes would have been helpful additions but fell outside of the scope of designing a proof of concept that worked and was feasible within our time constraints. This implementation would probably require buttons and networking or radio communication between the drone and a controlling device.

An additional improvement would be to provide a more vibration-dampening mount for the LiDAR, or use the manufacturer's recommended method for reducing severity of oscillation – mounting a US quarter to the side of the sensor. The drone, while capable of keeping steady position for the most part, was not easy to take off or land due to the constant wobbling induced by the LiDAR. While a way to remotely control the scanning of the LiDAR as discussed above would also be a means of correcting this issue, a mount that better addresses the oscillations or even allows the sensor to extend below the landing legs would be of great benefit to scan fidelity.