Field 3D PTV v1.0

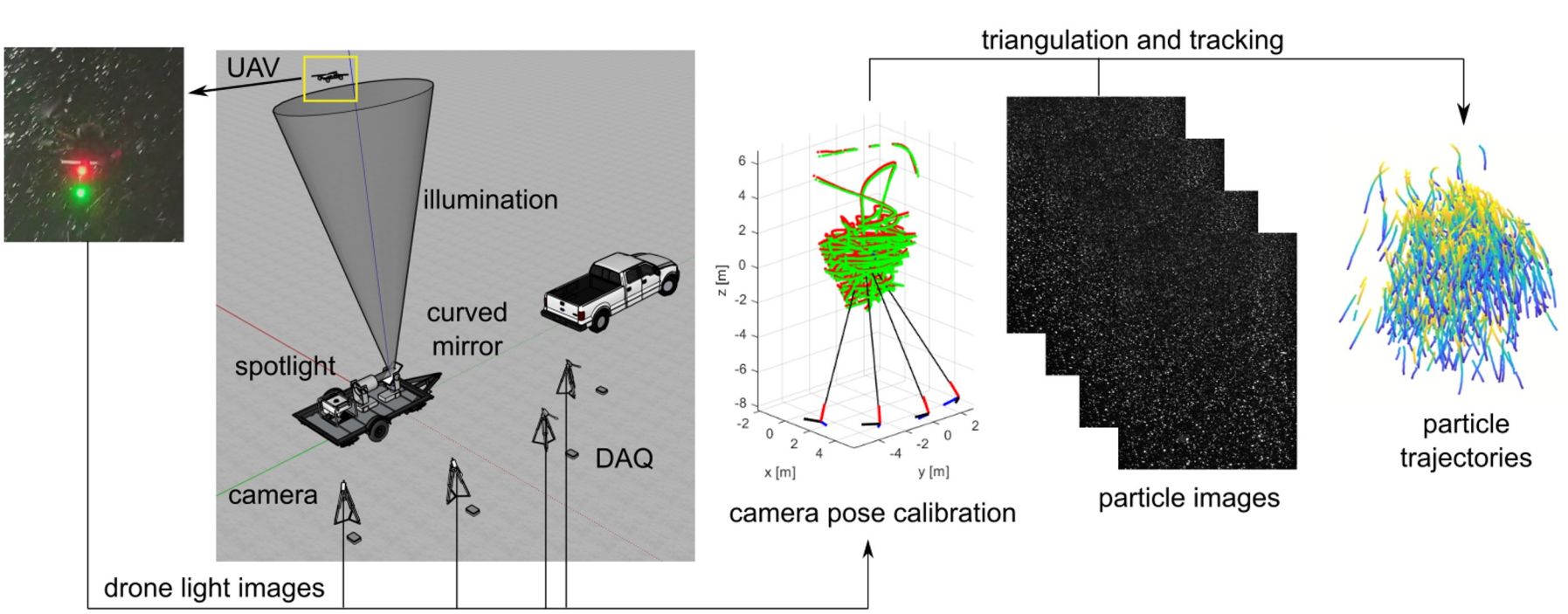
Flow Field Imaging Laboratory, University of Minnesota

Outlines

- Product description
  - Hardware
    - Hardware overview
    - Detailed description of key components
  - Software
    - Software overview
    - Detailed description of software used
- Image Acquisition procedure
- Data processing procedure
- Demonstration cases
  - Case I: Confetti Settling
  - Case II: Snow Settling
  - Case III: Pollen Dispersion

Product Description

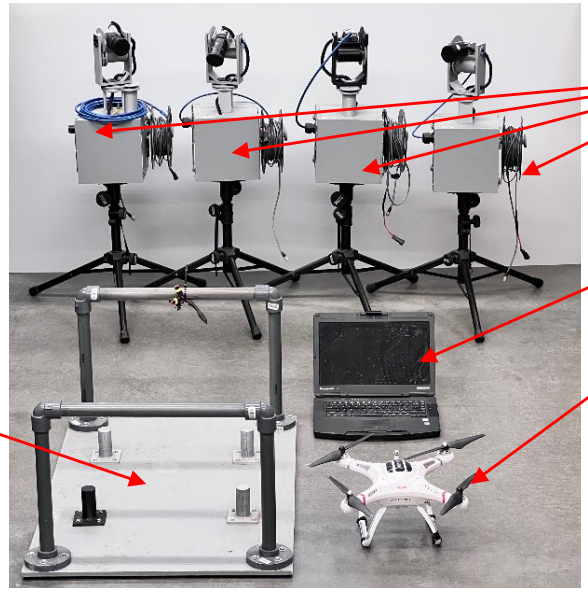
Field 3D PTV System v1.0 Overview



- Hardware: a WiFi-controlled four-camera multi-view imaging system for data recording and a drone with two-color LED spaced in a fixed distance for calibration
- Software: Remote camera control for field data acquisition, EasyWand code for calibration, and OpenLPT for 3D particle reconstruction

Hardware

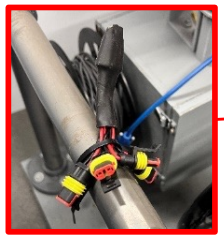
Hardware Overview



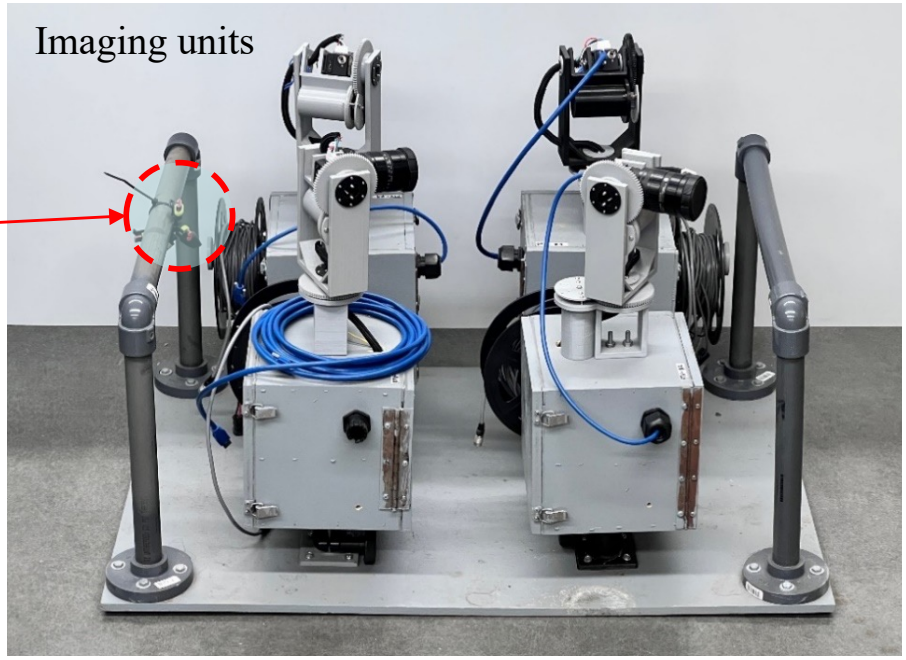
- Main hardware components
  - Imaging module for acquiring and storing PTV data
  - Central computer for controlling camera alignment and data acquisition
  - Calibration module for camera alignment and 3D image reconstruction
  - Lighting module for enhancing the signal in the desired sample volume (optional)
- Entire imaging hardware component can be transferred through a customized mounting platform

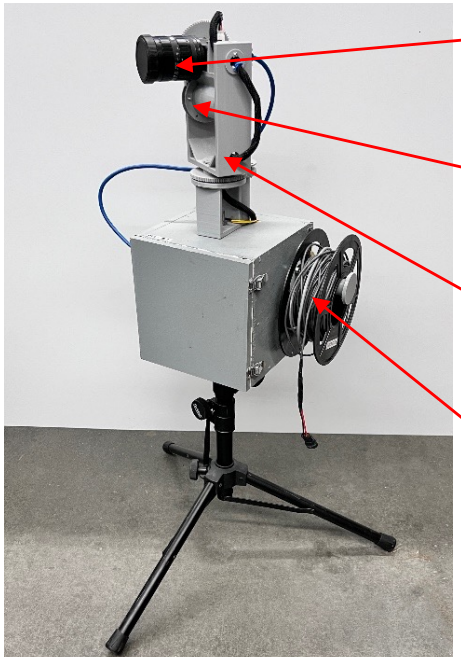
Notes:
• Hardware components: ~\Hardware\Open3DPTV_v1p0_component list.xlsx
• Imaging module (the four boxed units): camera, motorized stage, data processing, data storage, tripods
• Lighting module: optional depending on the background; some cases (e.g., pollen dispersion) use sun light.
• Calibration module: red and green LED attached to a carbon tube, with 34.5 cm gap

Imaging Module



Spools for sync cables

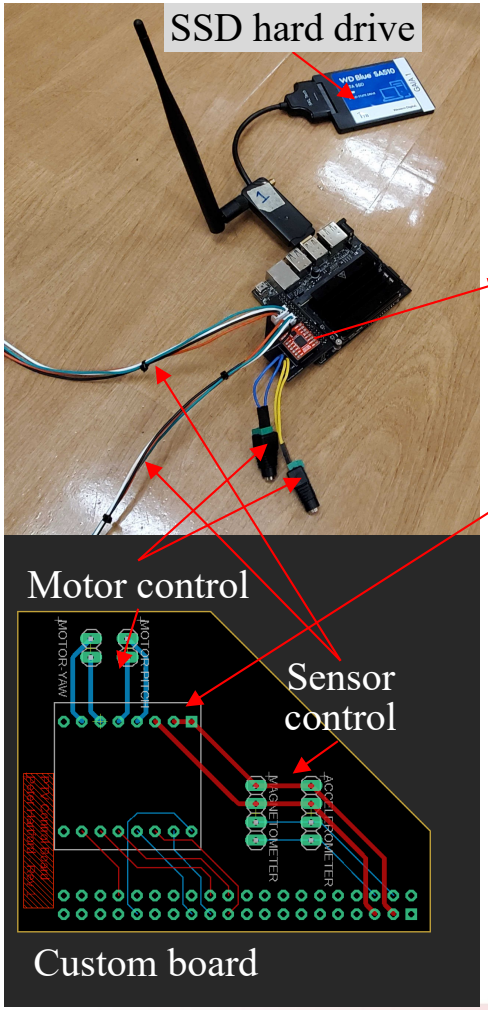




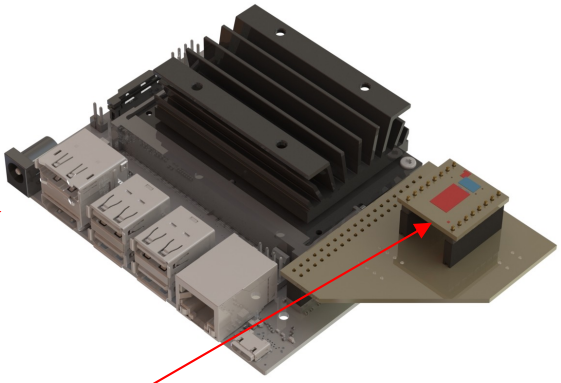
- Include four imaging units, protective wooden boxes for easy setup, each mounted on a tripod
- Cameras are synchronized via sync cables, with one master imaging unit controlling three slave units
- Each imaging unit includes a camera with a lens attached, a two-axis motorized stage for camera mounting and alignment, and GPS sensor. Data acquisition board are inside the wooden box enclosure

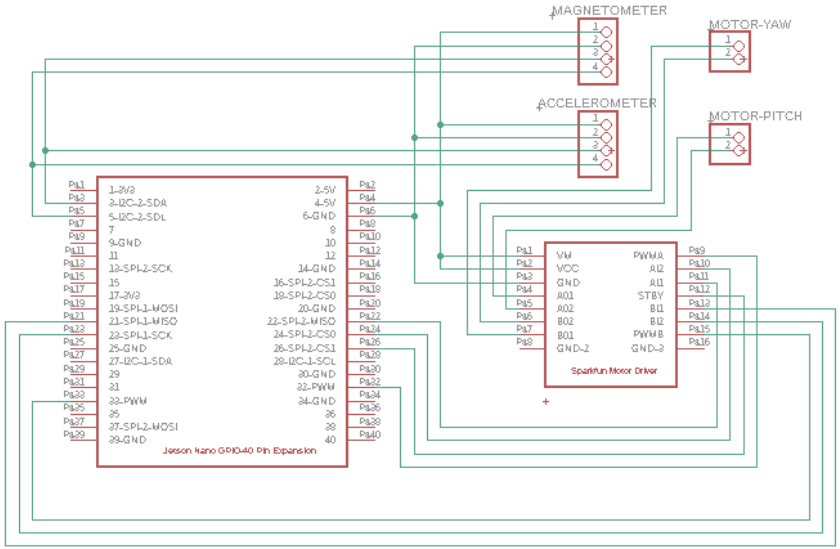
Notes:
- Camera: 2.8 Megapixel at 95 FPS, Lens: 16 mm f/1.4
- Two-axis motorized stage: one axis for pitch control with acceleration sensor attached and one axis for yaw control with magnetometer attached
- Data acquisition board: Nvidia Jetson Nano
- Box enclosure: 20 cm x 20 cm x 20 cm in dimension (design files in ~/Hardware/2nd Gen PTV Box)
- Platform dimension: 92 x 64 cm
- design files in ~/Hardware/Gimbal system/Printed Gimbal

Imaging Module: Camera and Motorized Stage Control



NVIDIA Jetson Nano



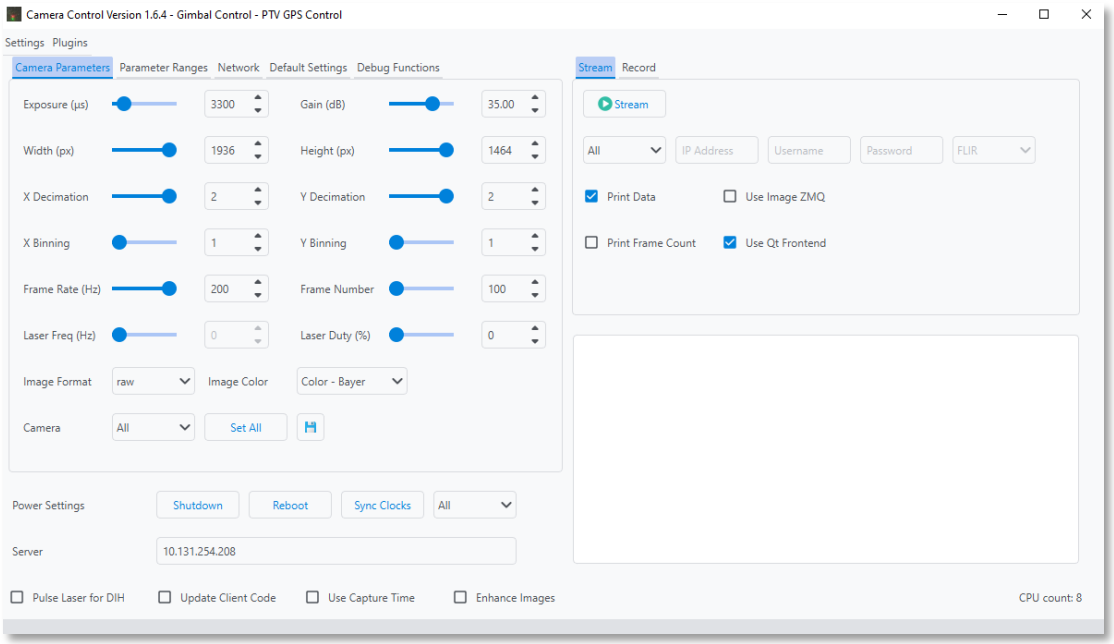


- Camera, WIFI adapter, and SSD hard drive connected to Jetson Nano directly through USB ports
- Sensors (magnetometers and accelerometers) and motors (pitch and yaw) connected to a custom PCB board attached to the Jetson Nano, with header pins and wired connections designed for easy replacement
- SSD hard drive held within the boxes and stack with the Jetson Nano and board

Notes:
- PCB design file location: ~/Hardware/Gimbal system/PCB Files
- Motorized stage control incudes sensors (magnetometer, accelerometer) and motors (pitch and yaw)
- Motorized stage connected to Jetson Nano through custom PCB board
- Camera (blue colored cable), SSD, WIFI through USB port

Central Computer for Data Acquisition Control

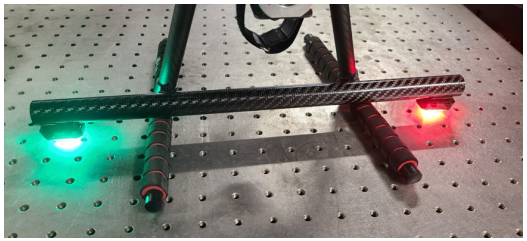
- Filed laptop for deployment
- Required Software: *Remote Camera Control* (latest version: v1.66)
  - Calculates camera spacing based on measurement volume and height
  - Integrates with the GPS control plugin to provide feedback control for motorized stage positioning
  - Manages imaging settings and camera recording functions

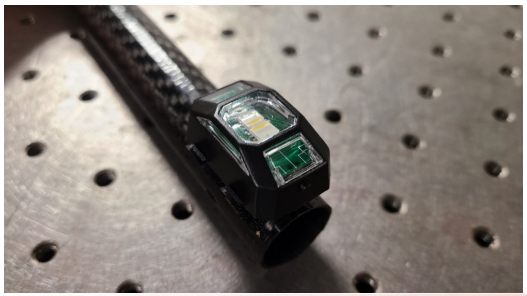


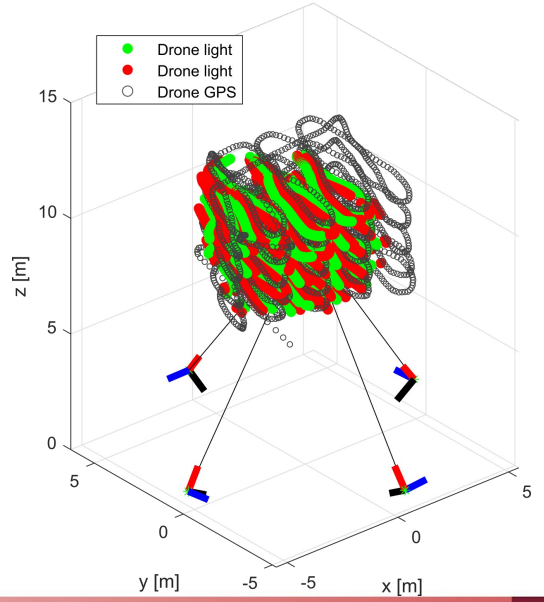
Notes:
• Remote Camera Control package included in ~/Software, downloadable from GitHub
• GitHub link: https://github.umn.edu/HongFlowFieldImagingLab/remote-camera-control

Calibration module

- Calibration Overview
  - Calibration establishes the geometric relationship between multiple cameras
  - Uses a calibration bar equipped with fixed-distance red and green lights
- User Tips
  - A drone is used to carry the bar; depending on the application, manual handling may be an option
  - Move the bar throughout the full sampling volume to ensure accurate calibration
  - Keep the bar horizontal and ensure the lights face downward during calibration





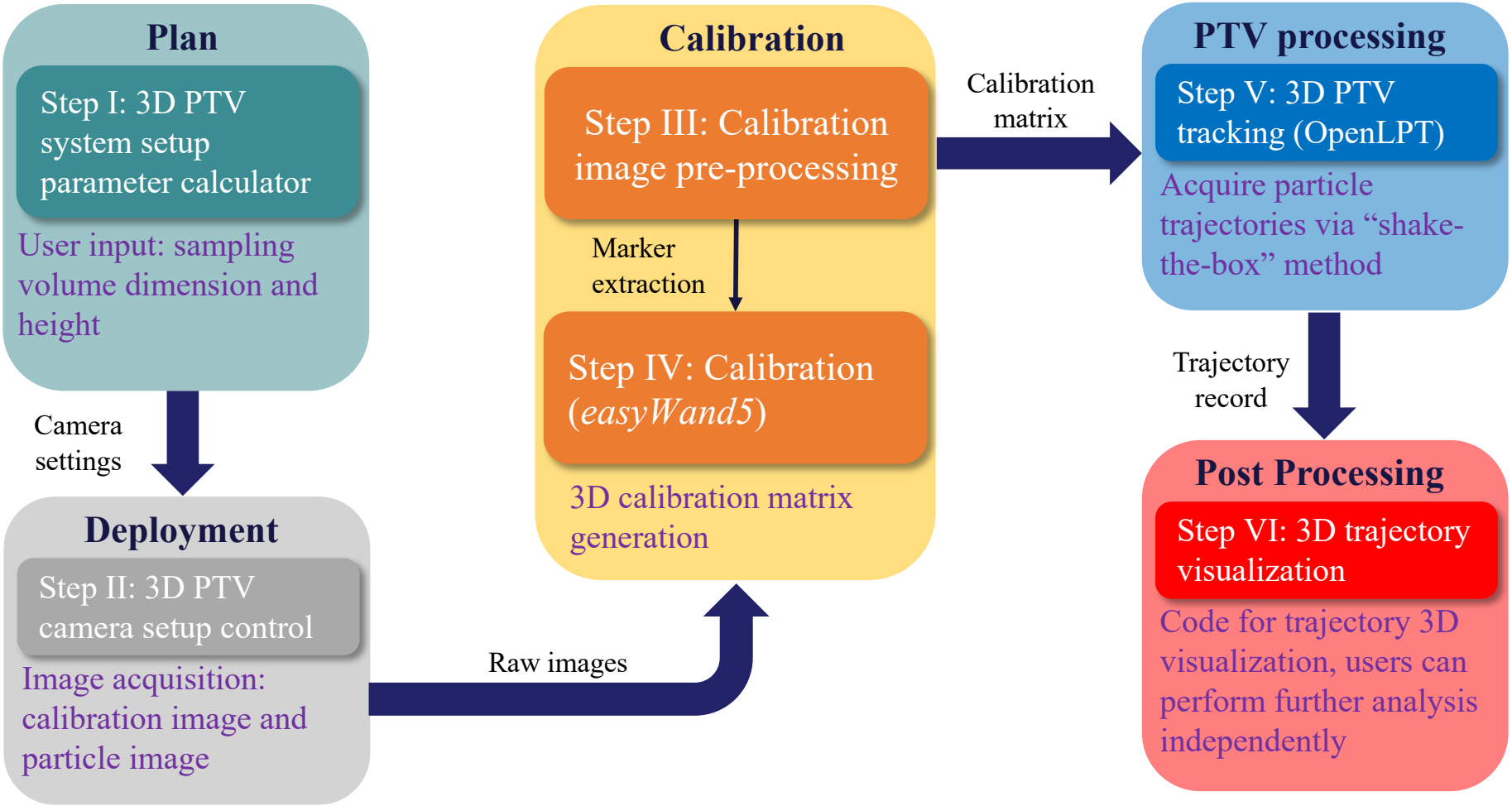


Notes:
• Calibration bar: the calibration scale depends on the drone used. The calibration bar used in our experiment is 34.5 cm long.

Software

Notes:
• Location: ~/Software

Software Overview

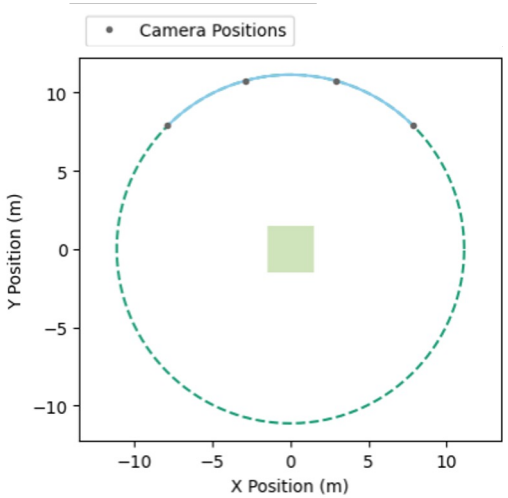


Notes:
- Step I (3DPTV system setup parameter calculator): ~/Software/3DPTV_Image_Acqusition/system setup parameter calculator
- Step II (3DPTV camera setup control): ~/Software/3DPTV_Image_Acqusition/camera setup control
- Step III (calibration image preprocessing): ~/Software/3DPTV_Processing_Pipeline/calibration image pre-processing
- Step IV (3D calibration matrix generation using easywand): ~/Software/3DPTV_Processing_Pipeline/easyWand5
- Step V (3D PTV tracking using OpenLPT): ~/Software/3DPTV_Processing_Pipeline/OpenLPT - ShakeTheBox
- Step VI (3D trajectory visualization): ~/Software/3DPTV_Processing_Pipeline/3D trajectory visualization

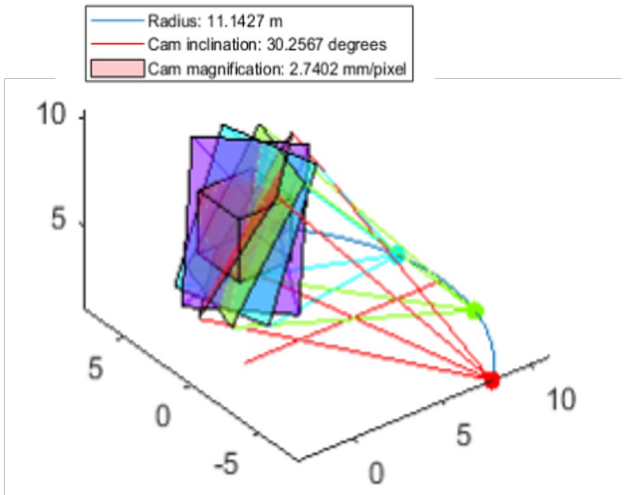
Step I: 3D PTV system setup parameter calculator

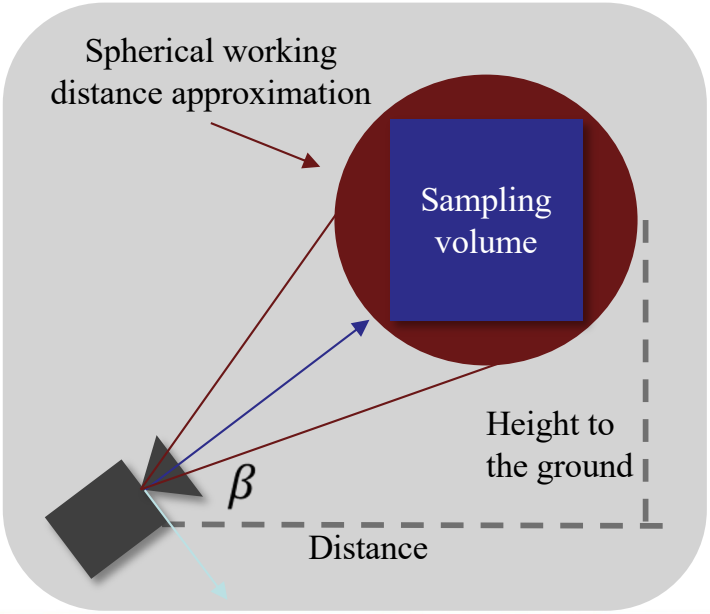
- MATLAB code for determining camera location and orientation
  - User inputs
    - Sampling volume dimensions and location (center height to the ground)
    - Radius of cameras
    - Camera parameters
  - Spherical working distance derived from sampling volume dimension
  - Camera position and orientation (yaw and pitch) calculated based on above inputs

Planed camera locations



Planed camera yaw and pitch

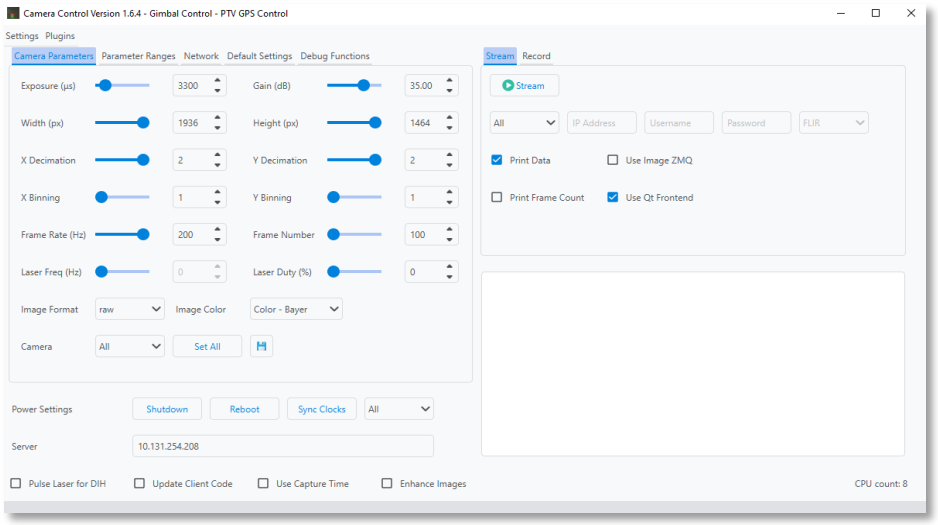


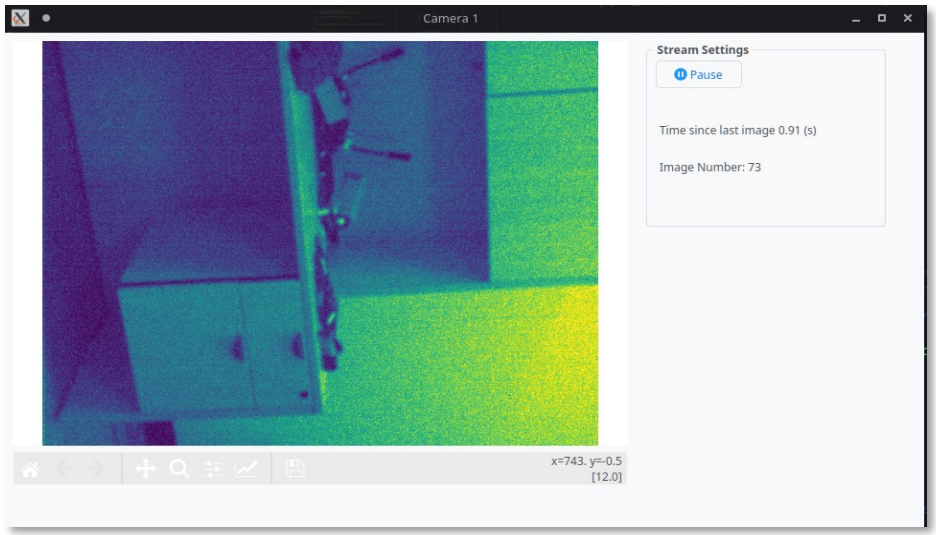


Notes:
• Input parameter examples: 6 m x 6 m x 6 m with center of sample volume at 12 m above the ground.
• Camera parameters can be found in the hardware section
• Software location: ~\Software\Multicamera System Calculator

Step II: 3D PTV camera setup control

- Remote camera control GUI via SSH
- Panel for camera settings
  - Pitch and yaw alignment adjusted with feedback control GUI
  - Imaging settings: frame rate, exposure, etc.
  - Multi-camera preview streaming
- Panel for video recording
  - Simple trigger with camera synchronization
  - Recording progress display



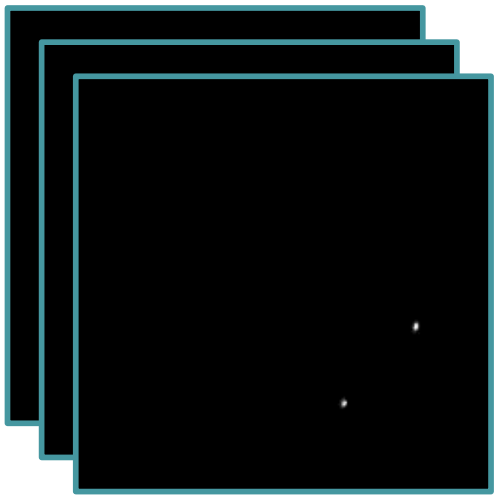


Notes:
- Launch the full remote camera control software and select from plugins
- Record using .raw file for high fps
- RGB mode for calibration imaging, BW for data acquisition
- Software location: ~\Software\camera setup control

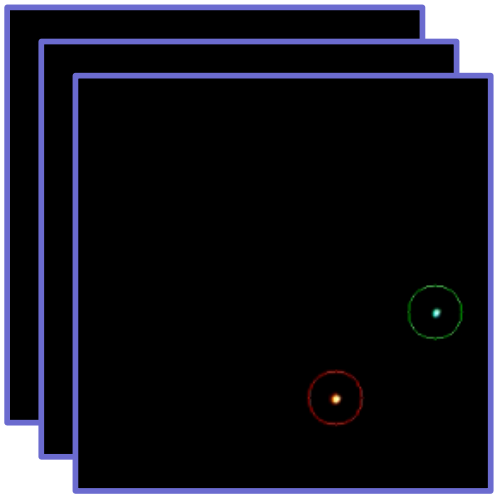
Step III: Calibration image pre-processing

- Python script for pre-processing calibration images
  - Load raw images
  - Extract marker coordinates from calibration images
  - Automatically detect red and green dots
  - Save marker coordinates as .csv files
  - Export an .avi file with markers overlaid for result review

Raw calibration images



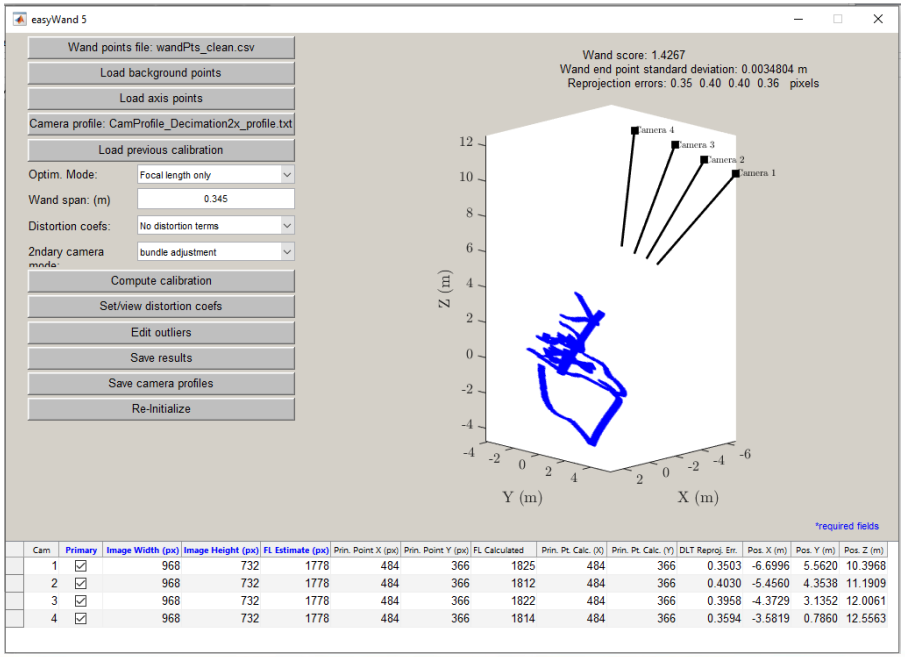
Markers identified



Notes:
• .raw format is used when recording to reach higher fps, needs decoding
• Color setting: Calibration images are RGB for identifying red and green dots, tracking images are monochrome
• Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/calibration image pre-processing

Step IV: Calibration (easyWand5)

- Wand-based multi-camera calibration tool
  - GUI for calibrating 2 or more cameras using a calibration wand
  - Based on the MATLAB Computer Vision toolbox
- Converts calibration results for use in OpenLPT
  - Recalculates calibration using the Tsai method employed in OpenLPT
  - Generates necessary configuration files for running OpenLPT

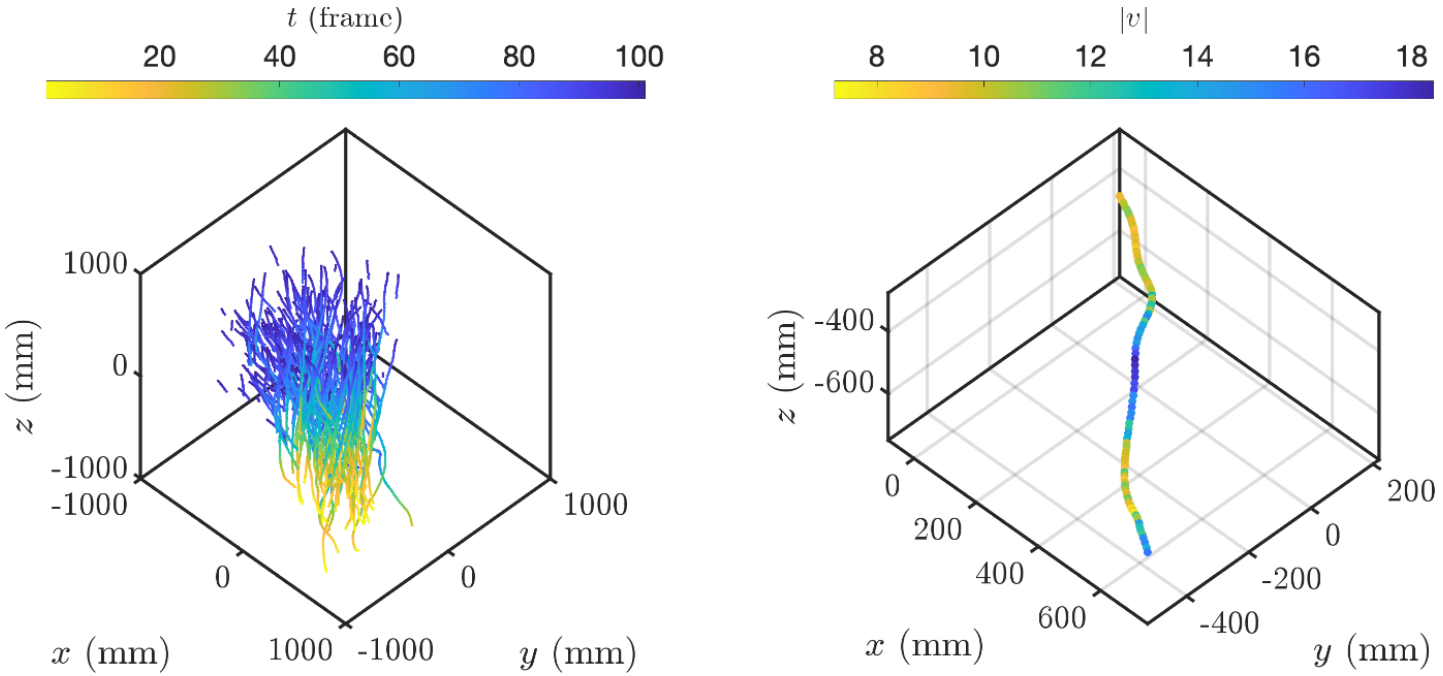


Cam	Primary	Image Width (px)	Image Height (px)	FL Estimate (px)	Prin. Point X (px)	Prin. Point Y (px)	FL Calculated	Prin. Pt. Calc. (X)	Prin. Pt. Calc. (Y)	DLT Reproj. Err.	Pos. X (m)	Pos. Y (m)	Pos. Z (m)
1	☑	968	732	1778	484	366	1825	484	366	0.3503	-6.6996	5.5620	10.3968
2	☑	968	732	1778	484	366	1812	484	366	0.4030	-5.4560	4.3538	11.1909
3	☑	968	732	1778	484	366	1822	484	366	0.3958	-4.3729	3.1352	12.0061
4	☑	968	732	1778	484	366	1814	484	366	0.3594	-3.5819	0.7860	12.5563

Notes:
- easyWand does not support Mac with Apple Silicon
- Downloadable from: https://biomech.web.unc.edu/wand-calibration-tools/
- easyWand5 location: ~/Software/3DPTV_Processing_Pipeline/easyWand5
- Converting results: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT

Step V and VI: 3D PTV tracking (OpenLPT) and visualization

- OpenLPT: Open-source Lagrangian particle tracking code
  - Based on the Shake-the-Box method
  - Optimized for removing ghost particles at a high particle image density
- Code for trajectory visualization is provided
  - Users can perform further analysis independently



Notes:
• OpenLPT package: ~/Software/3DPTV_Processing_Pipeline/OpenLPT - ShakeTheBox.zip
• OpenLPT post processing: ~/Software/3DPTV_Processing_Pipeline/3D trajectory visualization
• OpenLPT github page: https: //github.com/JHU-NI-LAB/OpenLPT_Shake-The-Box
• References: Tan, S., Salibindla, A., Masuk, A. U. M., & Ni, R. (2020). Experiments in Fluids, 61, 1-16.

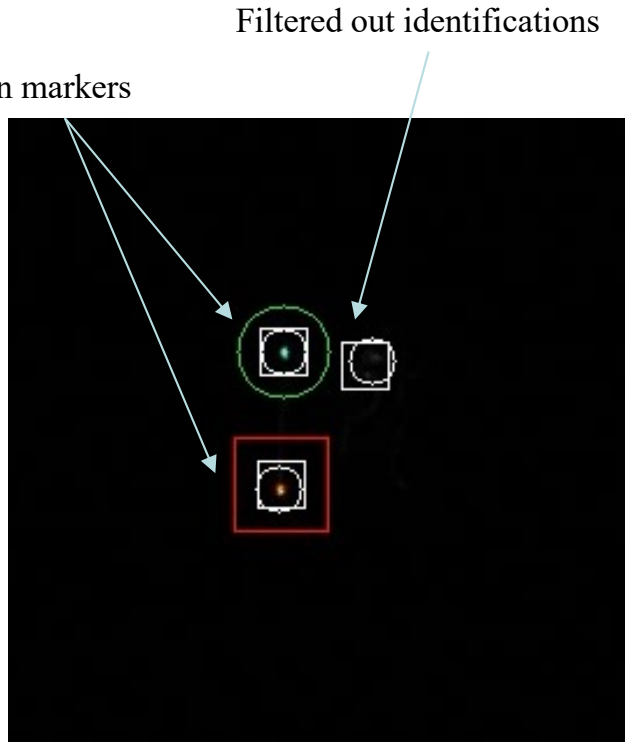
Date Processing Procedure

Overview

- Step 1: Extract calibration markers from raw images
- Step 2: Run easyWand to generate the calibration profile
- Step 3: Convert the calibration data to an OpenLPT-compatible format
- Step 4: Use OpenLPT to obtain 3D particle coordinates
- Step 5: 3D trajectory visualization

Step 1: Extract calibration markers from raw images

- Organize raw image files into subfolders by camera under a single calibration folder
  - E.g., "/root_path/calA/cam1"
- Run "droneMarkerTracking_v3.py" in "calibration image pre-processing " folder
  - GUI pops up and select calibration folder
    - E.g., "/root_path/calA"
  - Images are processed from each camera and the results are displayed
    - White circles and boxes indicate identified markers, while red and green boxes represent the most confident marker identifications. The remaining white circles and boxes are filtered out.
  - Results will be saved in two files
    - wandPts_all.csv (nice to keep)
    - wandPts_clean.csv (cleaned up results with outliers filtered out based on path continuity)



Notes:
• Software path, calibration image pre-processing: ~/Software/3DPTV_Processing_Pipeline/calibration image pre-processing

Step 2: Run easyWand to generate the calibration profile

- Run easyWand5
  - Ensure the software is in " ~/Documents/Matlab".
  - Set this folder as the current directory in Matlab and enter easyWand5 in the command line.
- Compute the initial calibration:
  - Load the previously generated wand points file (wandPts_clean.csv).
  - Click "Load camera profile" and select the appropriate file from the "Camera profiles" folder.
  - Set the wand span to the measured distance between the drone lights.
  - Leave other parameters as default and click "Compute Calibration."
- Refine calibration
  - Reprojection error will likely still be > 1 pixel, and wand score >>1
  - Press "edit outliers" → pick >one point to be removed with high residual → rerun compute calibration
  - Press "Save results" as .mat file
  - If needed, run Matlab script "cleanEasyWand.m" in "easyWand5/easyWand to OpenLPT" folder
    - load .mat file → filter out points with high error → resave a new calibration .mat file
  - In easyWand5, press "Load previous calibration", and select newly saved .mat file
  - Press "Compute calibration"
    - Results should yield < 1 pixel reprojection errors → If so, save these results
    - If not, generate a "cleaner" calibration with a lower threshold value

Notes:
• Software path, easyWand package: ~/Software/3DPTV_Processing_Pipeline/easyWand5

Step 3: Convert the calibration data to an OpenLPT-compatible format

- Create a folder for a new OpenLPT project
  - Add image files, saved in "/cam*/B*****.tif" format, where each camera has its own folder
    - Can convert from .raw to .tif using "convertImage.py" in "calibration image pre-processing/Utils"
  - This will be needed to match the config files generated below
- In "easyWand5/easyWand to OpenLPT ", run "easyWand2OpenLPT.m"
  - Will prompt to load the easyWand results file generated
  - This takes the 3D calibration points that easyWand has solved for, and uses them to recompute a new calibration using the Tsai method employed in the OpenLPT calibration tool
  - This also generates the needed configuration files for running OpenLPT
  - Note that scaling factor of 10 is important for OpenLPT not breaking down due to viewing area being too large

Notes:
• Software path, conversion codes: ~/Software/easyWand5/easyWand to OpenLPT

Step 4 & 5: Obtain 3D coordinates via Open LPT and visualize trajectories

- Installation
  - Follow the instructions in the "OpenLPT Instructions from JHU" folder to complete the installation.
- Parameter setup
  - Configuration files are generated automatically, the following two files require particular attention:
    - *trackingconfig.txt*:
      - Keep the view area setting unchanged
    - *iprconfig.txt*:
      - *mindist2d* and *mindist3d*: Set equal to or greater than the calibration error (e.g., ~0.25)
      - Set the particle size accordingly.
- Executing the Code
  - Open a command prompt in the OpenLPT root directory: "~/ShakeTheBox/Release_windows"
  - Run the executable using the following command with the OpenLPT project root:
    - .\ShakeTheBox "path-to-OpenLPT-project-root"
- OpenLPT results
  - Output files are saved as .txt files in the "Tracks" folder
  - Results can be visualized using the scripts provided in the "3D Trajectory Visualization" folder

Notes:
• The C++ code for OpenLPT needs to be compiled
• In trackingconfig.txt: The view area setting should remain unchanged (even though it appears very small, in mm).
• Software path, OpenLPT: ~/Software/OpenLPT – ShakeTheBox
• Software path, trajectory visualization: ~/Software/3D trajectory visualization

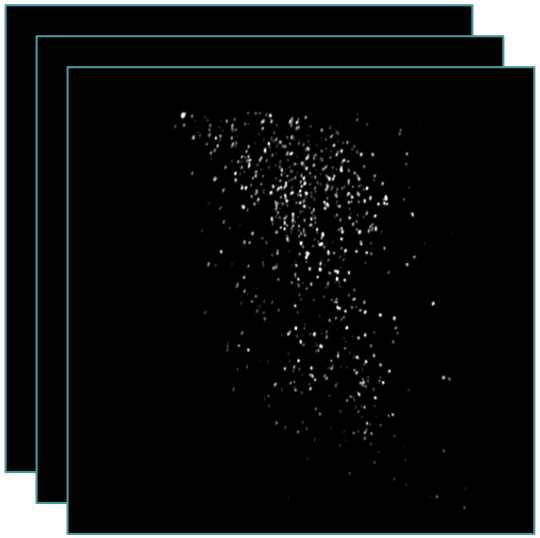
Demonstration Cases

Case I: Confetti Settling

Experimental Setup

- This example serves as a concept validation for the 3D PTV system
- Case description and planning
  - Confetti particles made from 6 mm diameter paper hole punches, hand tossed and fall freely
  - Sampling volume: 3 m × 3 m × 2 m, volume center height: 1 m

Raw particle images

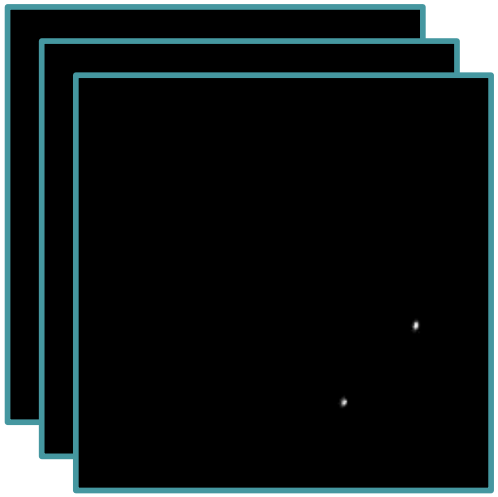


Notes:
• This case was for concept validation in Bristow 2023 EXIF, paper in ~/Field3D_PTV/papers
• Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling
• Reference: Bristow, N., Li, J., Hartford, P., Guala, M., & Hong, J. (2023). Experiments in Fluids, 64(4), 78.

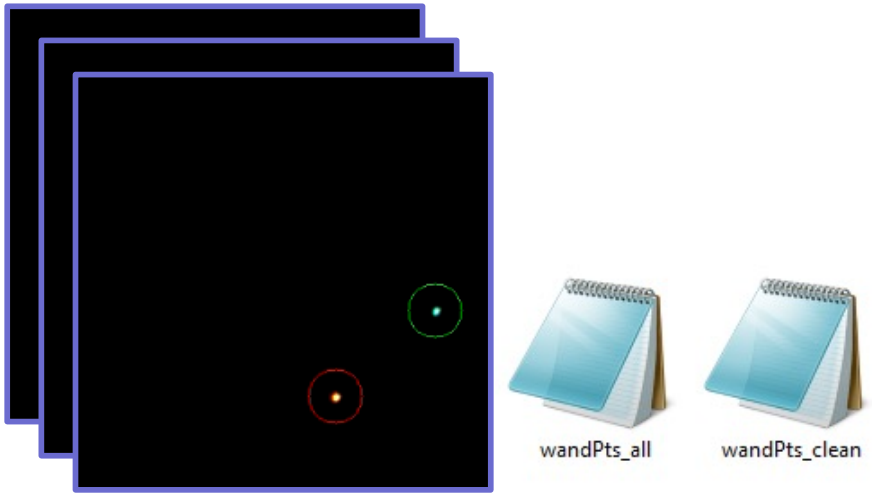
Extract marker coordinates from raw calibration images

- Find “droneMarkerTracking_v3.py” in “calibration image pre-processing” folder
- Select path to raw calibration images and run the code
- Marker coordinates are exported as .csv files for easyWand calibration in the next step

Raw calibration images



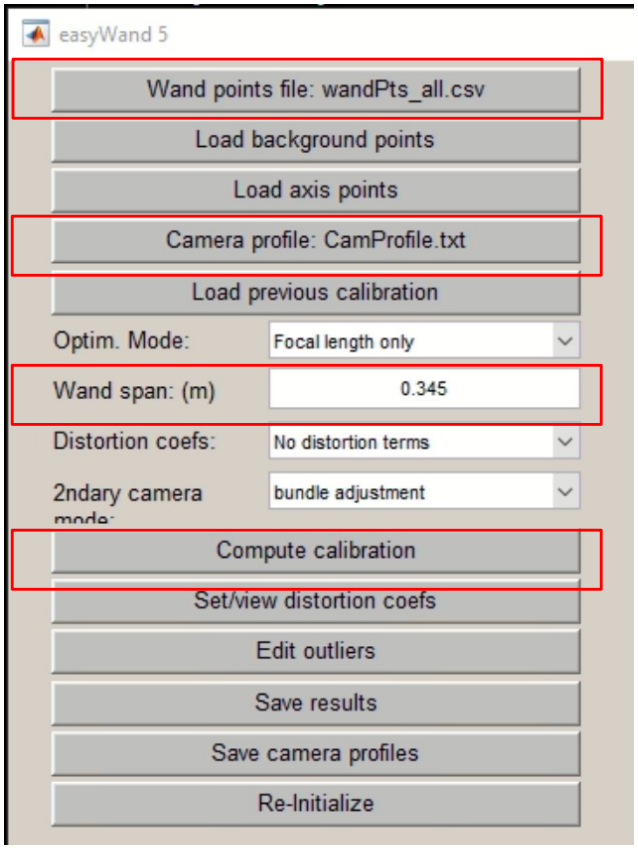
Markers identified and coordinates saved as .csv files

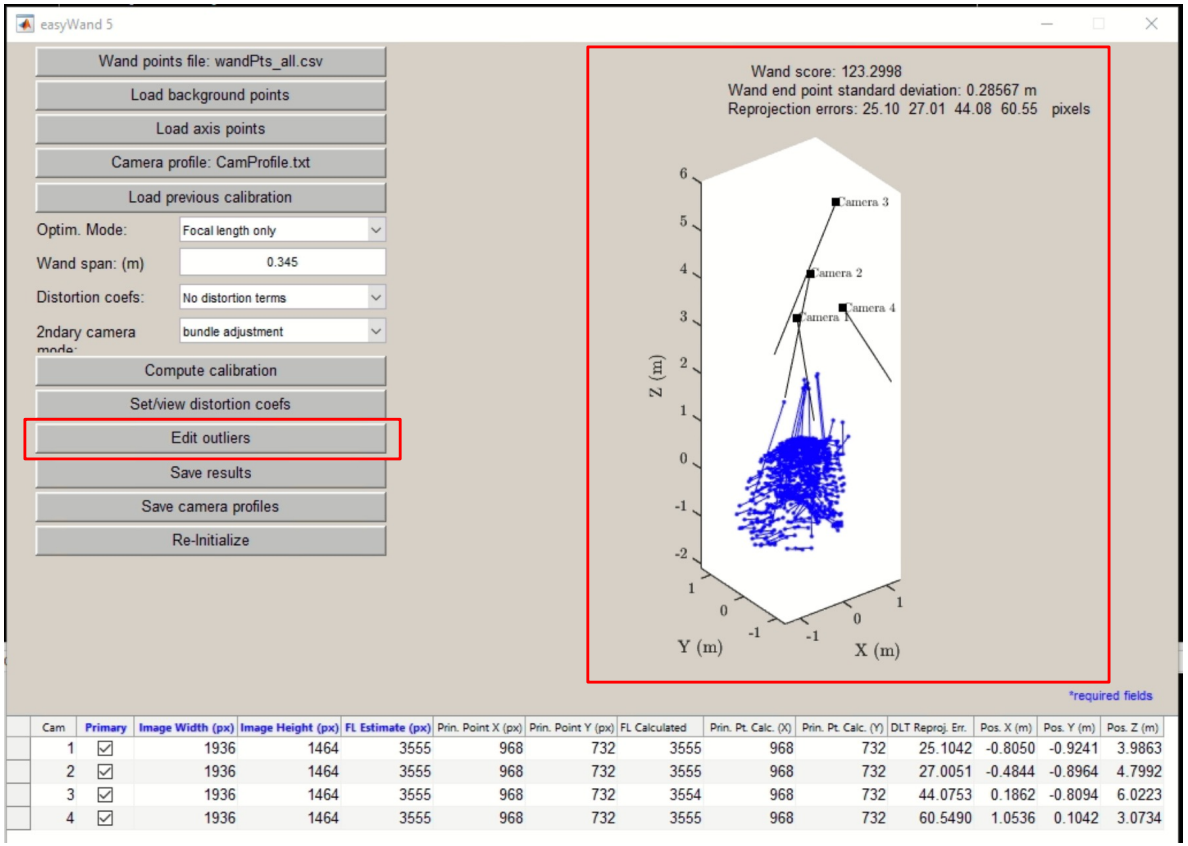


Notes:
- Inout: raw marker image; opt put: wand point coordinates
- Color setting: Calibration images are RGB for identifying red and green dots, tracking images are bw
- More details in “Data Processing Procedure”, the code generates .avi file with marker noted for verification
- The calibration working folder, named “calibration_proc_folder,” is included in each demo case.
- Code: ~/Software/3DPTV_Processing_Pipeline/calibration image pre-processing
- Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling/calibration_proc_folder

Compute Calibration Profile

- Run “easyWand5” in Matlab, load wand points and camera profile, enter wand spam, compute calibration
- The first result can be with large error, use “Edit outliers” to refine results



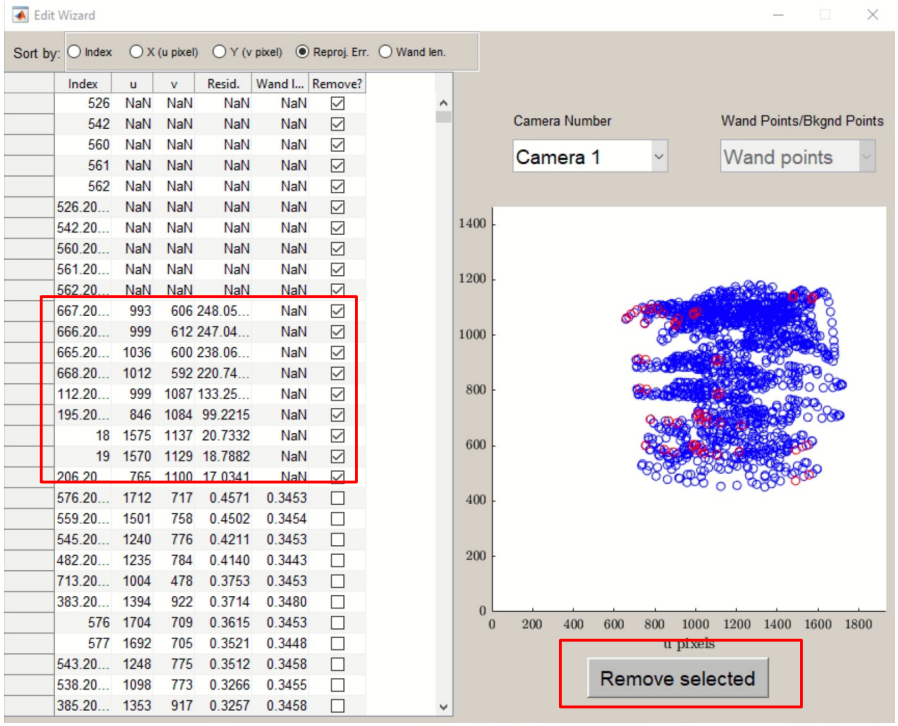


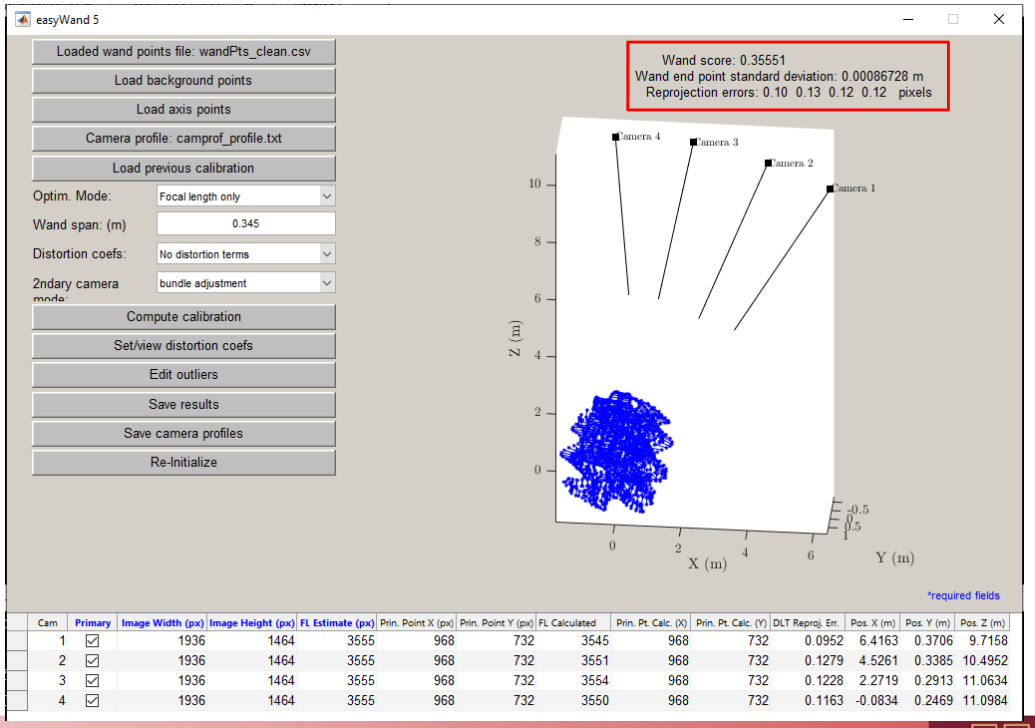
Cam	Primary	Image Width (px)	Image Height (px)	FL Estimate (px)	Prin. Point X (px)	Prin. Point Y (px)	FL Calculated	Prin. Pt. Calc. (X)	Prin. Pt. Calc. (Y)	DLT Reproj. Err.	Pos. X (m)	Pos. Y (m)	Pos. Z (m)
1	☑	1936	1464	3555	968	732	3555	968	732	25.1042	-0.8050	-0.9241	3.9863
2	☑	1936	1464	3555	968	732	3555	968	732	27.0051	-0.4844	-0.8964	4.7992
3	☑	1936	1464	3555	968	732	3554	968	732	44.0753	0.1862	-0.8094	6.0223
4	☑	1936	1464	3555	968	732	3555	968	732	60.5490	1.0536	0.1042	3.0734

Notes:
- Input: wand point coordinates, camera profile, wand span
- Wand span: 34.5 cm
- Camera profile: ~/Software/3DPTV_Processing_Pipeline/Camera profiles
- Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5
- Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling/calibration_proc_folder

Refine Calibration Profile

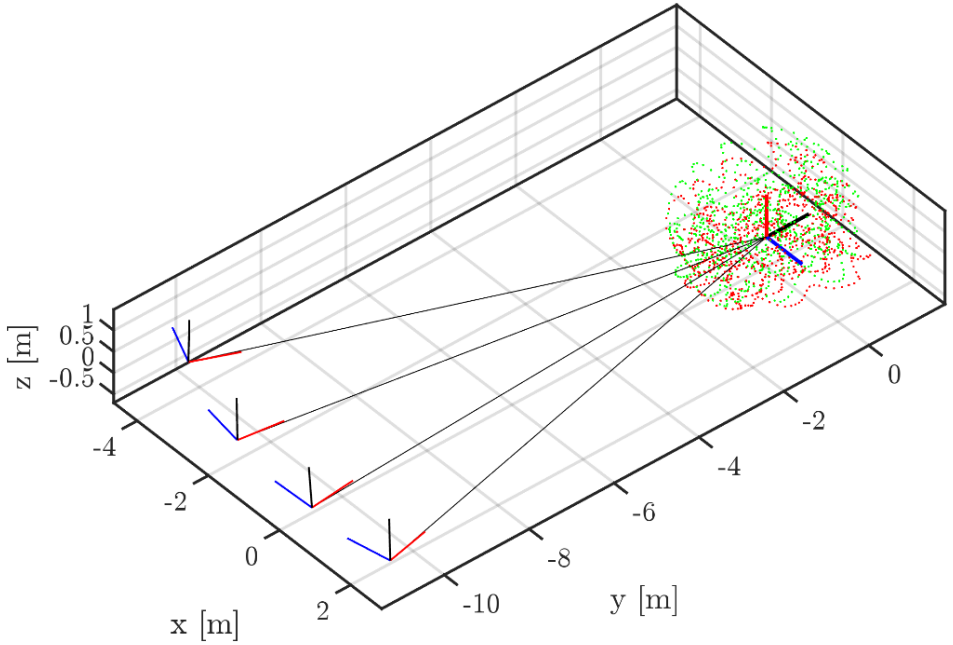
- Sort calibration points by the residual, remove results that are significantly larger
- Compute calibration again, check wand score and errors, repeat previous step again
- Use “cleanEasyWand.m” in “easyWand5/easyWand to OpenLPT ” folder to filter results if needed
- Save results (.mat file) when the wand score and error meet the desired criteria





Notes:
- Should target for wand score <1 and pix error <<1
- Output: .mat file from easyWand calibration
- Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
- Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling/calibration_proc_folder

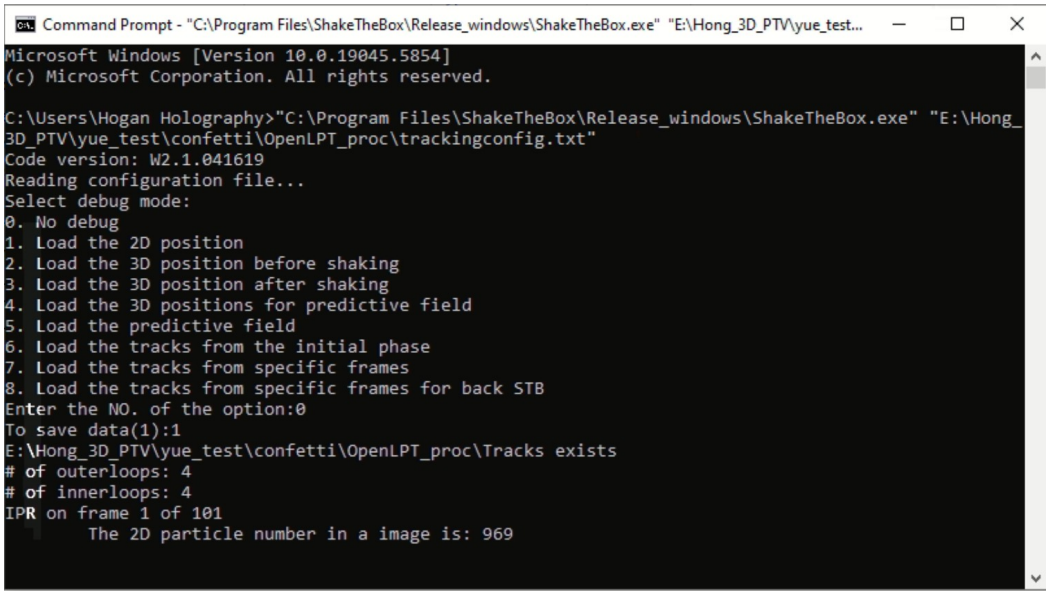
Calibration Results



- Wand quality score: 0.35
- Wand end point standard deviation: 0.8 mm (total wand length: 34.5 cm)
- Reprojection error of each camera (Cam 1: 0.1 pix; Cam 2: 0.13 pix; Cam 3: 0.12 pix; Cam 4: 0.12 pix)
- Wand score of 1 or less indicates a good calibration (STD/mean = 1%)
- Run code “easyWand2OpenLPT.m” to export config files for OpenLPT

Notes:
• Input: .mat file from easyWand calibration; Output: config files for OpenLPT
• The code also converts the coordinate system
• Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
• Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling/calibration_proc_folder

3D Coordinate Calculation using OpenLPT

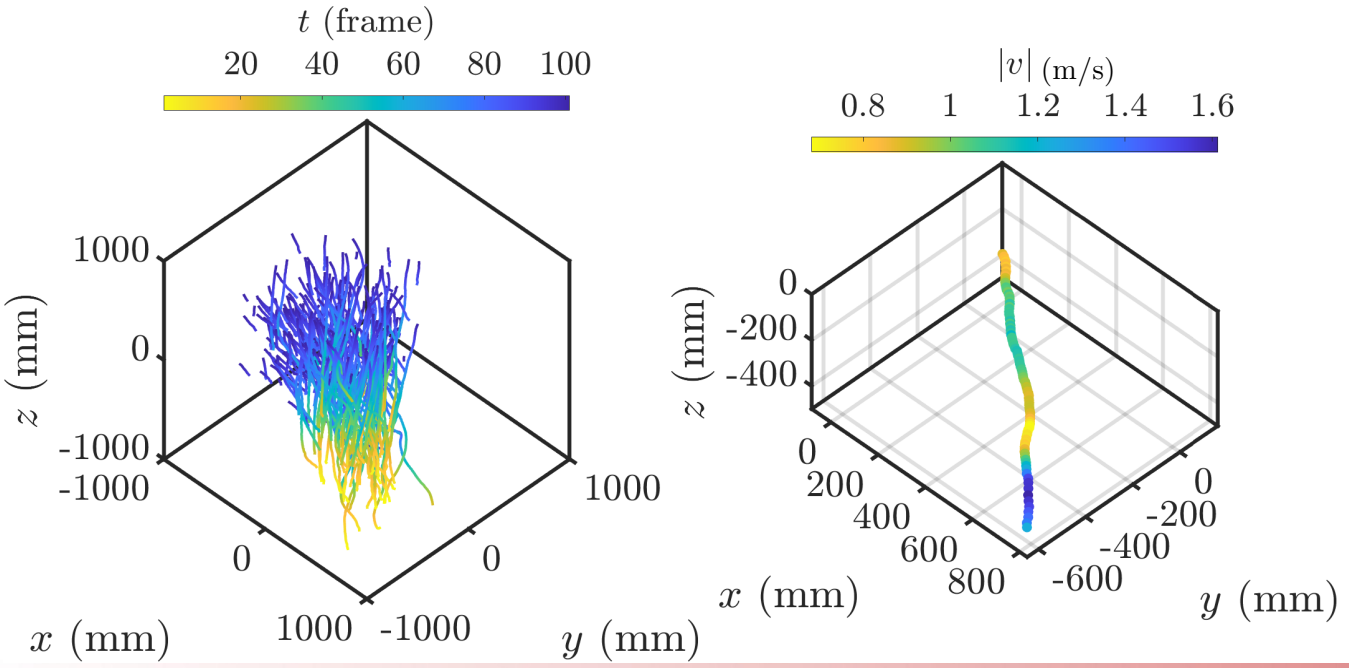


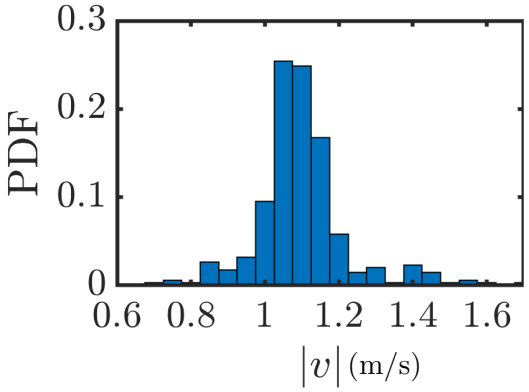
- Create project folder for OpenLPT, organize .tif images into the /cam*/ folders
- Convert easyWand results to OpenLPT configuration files (Use "easyWand2OpenLPT.m" in "easyWand5/easyWand to OpenLPT ")
- Run OpenLPT (in command window, enter "path-to-ShakeTheBox\ShakeTheBox.exe" "path-to-data\OpenLPT_proc\trackingconfig.txt")
- Results are saved in "Tracks" folder as txt files

Notes:
• Make sure path in the configuration files are correct
• The OpenLPT working folder is "OpenLPT_proc_folder", provided in each demo cases
• Data: ~/Field3D_PTV/Demonstration cases/Case I_Confetti settling/OpenLPT_proc_folder

Trajectory Visualization and Post-processing

- Trajectory yield: 93% of particles successfully captured by the system (464/501)
- Fall speed evaluation: PTV fall speed measurements closely matched independently measured values for individual confetti particles (~1m/s)
- Trajectory analysis: Captured fine-scale motions accurately, reveals details of tumbling motions of the confetti particles



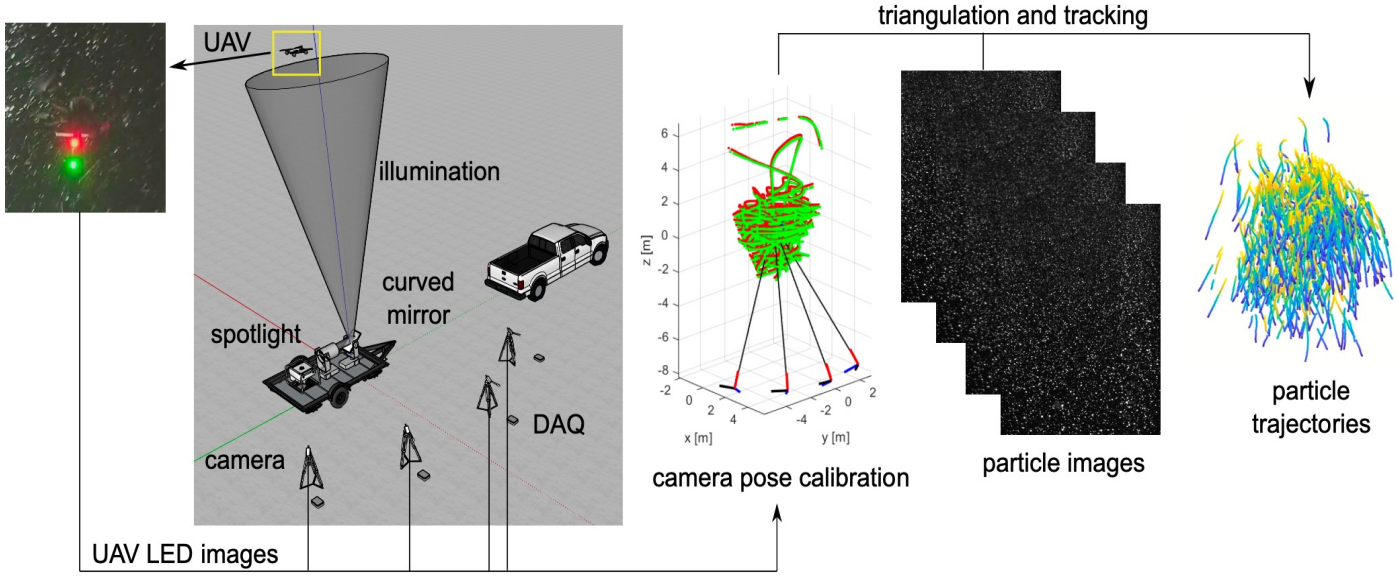


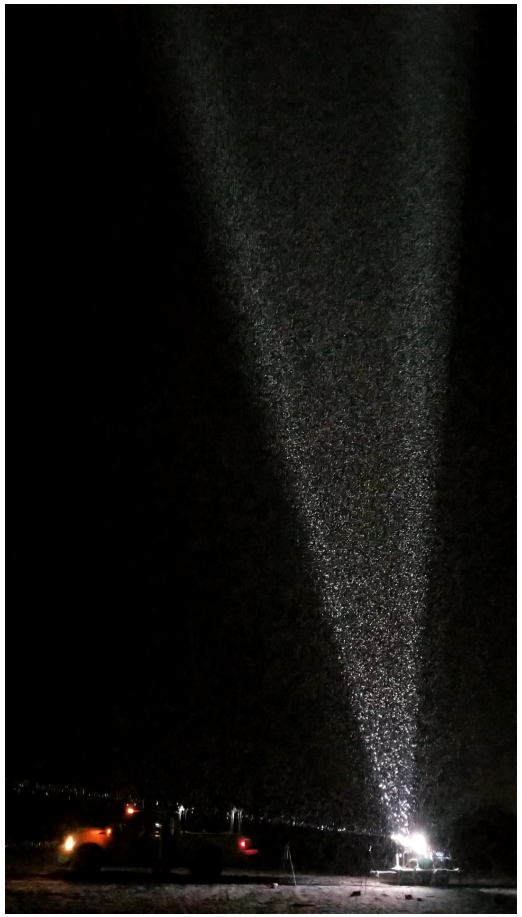
Notes:
• Run VisualizeResults_2.m to visualize tracks and analyze velocity distribution
• Location: ~\Code\3DPTV_Processing_Pipeline\3D trajectory visualization\VisualizeResults_3

Case II: Snow Settling

Experimental Setup

- Investigation on snow settling under different atmospheric conditions
  - Low turbulence (snow morphology play a role)
  - Imaging 4 m × 4 m × 6 m volume, ~ 10 m above ground
  - Spatial resolution 6.3 mm/voxel; temporal resolution 200 Hz
  - Duration: in total 26 50-second sequences

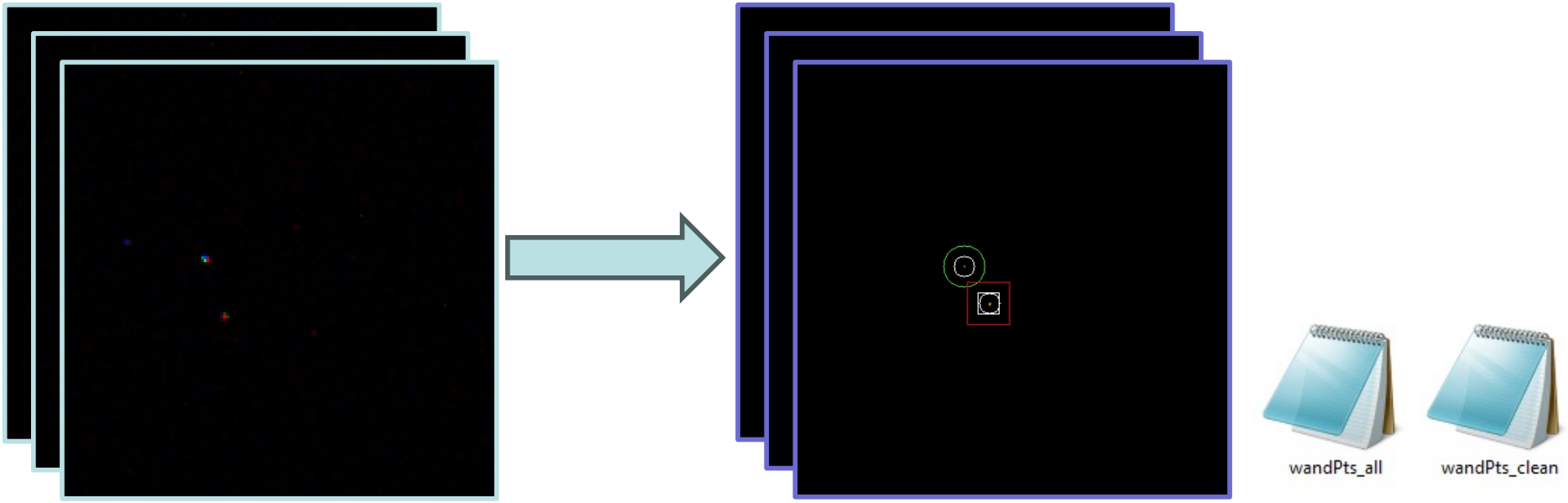




Notes:
• This case was the main experiment in Bristow 2023 EXIF, paper in ~/Field3D_PTV/papers
• Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow
• Reference: Bristow, N., Li, J., Hartford, P., Guala, M., & Hong, J. (2023). Experiments in Fluids, 64(4), 78.

Extract marker coordinates from raw calibration images

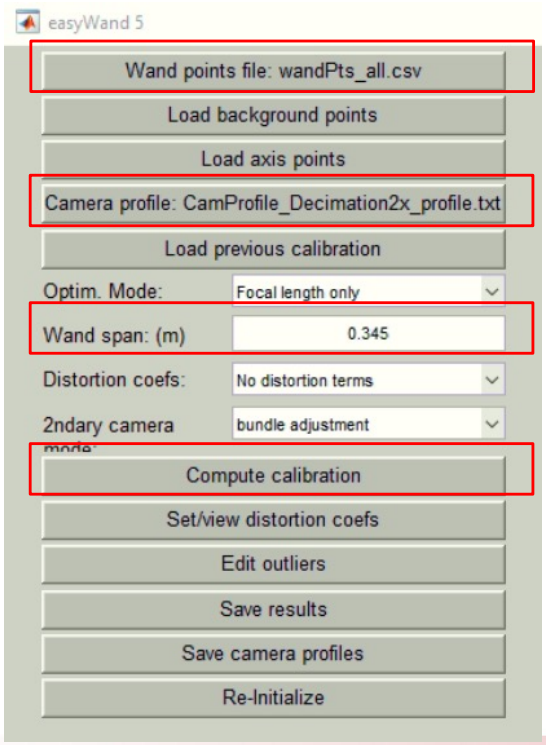
- Extract marker coordinates from raw calibration images
  - Find “droneMarkerTracking_v3.py” in “calibration image pre-processing” folder
  - Select path to raw calibration images and run the code
  - Marker coordinates are exported as .csv files for easyWand calibration in the next step

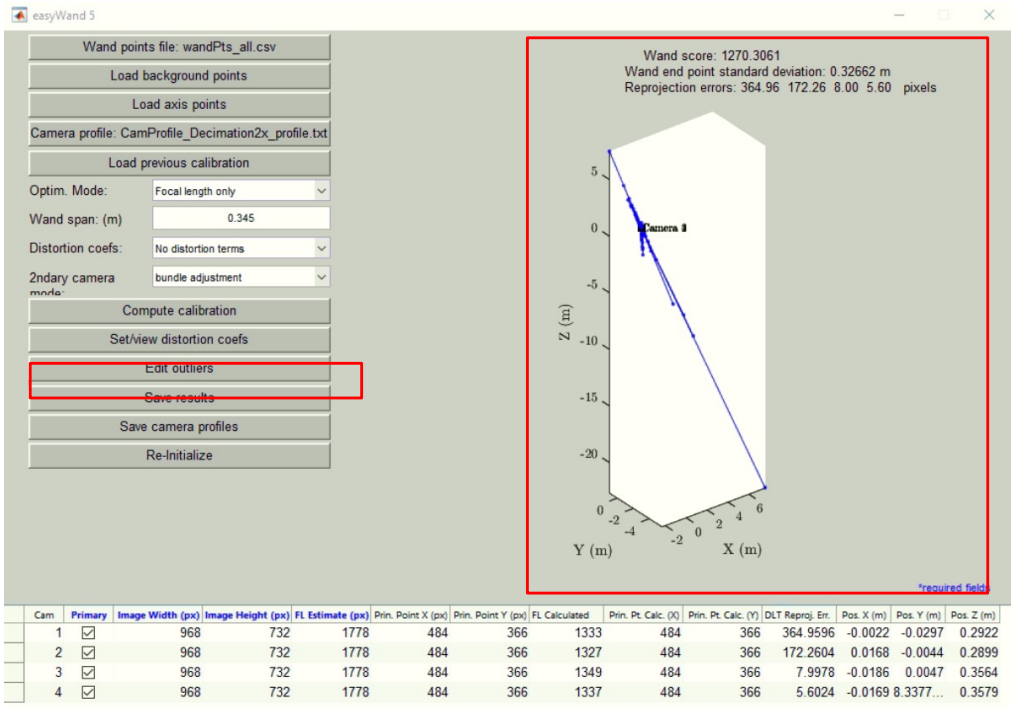


Notes:
• Inout: raw marker image; opt put: wand point coordinates
• Color setting: Calibration images are RGB for identifying red and green dots, tracking images are bw
• More details in “Data Processing Procedure”, the code generates .avi file with marker noted for verification
• The calibration working folder, named “calibration_proc_folder,” is included in each demo case.
• Code: ~/Software/3DPTV_Processing_Pipeline/calibration image pre-processing
• Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow/calibration_proc_folder

Compute Calibration Profile

- Compute calibration profile
  - Run “easyWand5” in Matlab, load wand points and camera profile (Decimation2X), enter wand spam, compute calibration
  - The first result is with large error, use “Edit outliers” to refine results



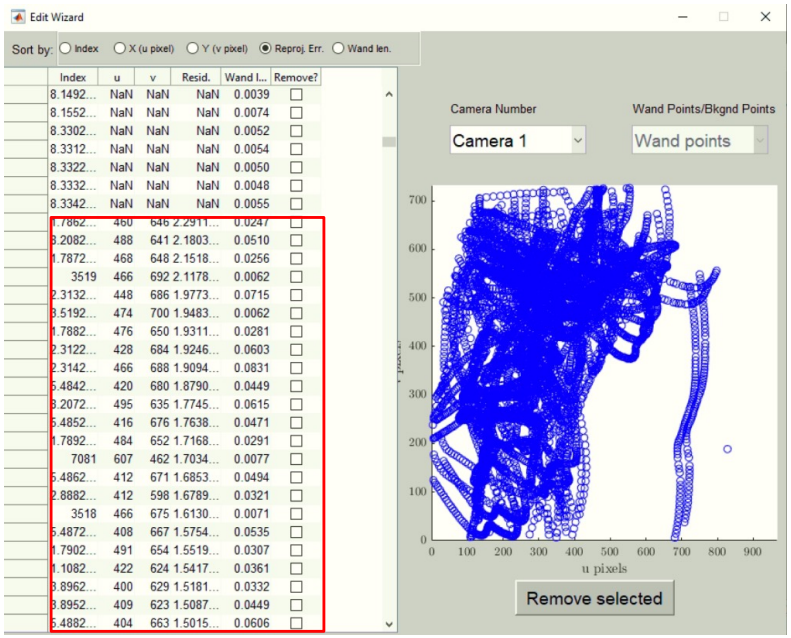


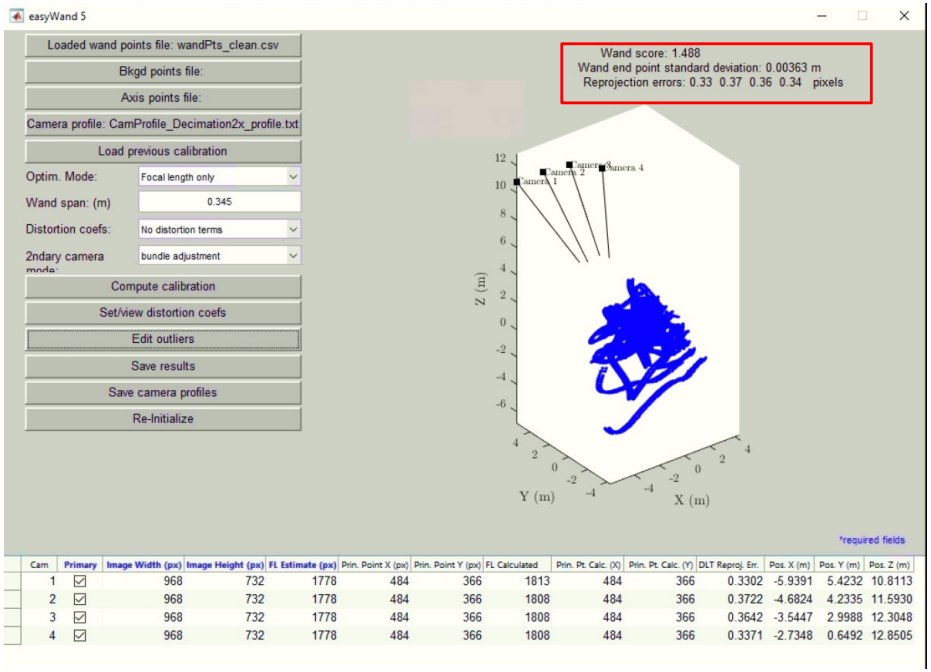
Cam	Primary	Image Width (px)	Image Height (px)	FL Estimate (px)	Prin. Point X (px)	Prin. Point Y (px)	FL Calculated	Prin. Pt. Calc. (X)	Prin. Pt. Calc. (Y)	DLT Reproj. Err.	Pos. X (m)	Pos. Y (m)	Pos. Z (m)
1	☑	968	732	1778	484	366	1333	484	366	364.9596	-0.0022	-0.0297	0.2922
2	☑	968	732	1778	484	366	1327	484	366	172.2604	0.0168	-0.0044	0.2899
3	☑	968	732	1778	484	366	1349	484	366	7.9978	-0.0186	0.0047	0.3564
4	☑	968	732	1778	484	366	1337	484	366	5.6024	-0.0169	8.3377...	0.3579

Notes:
- Input: wand point coordinates, camera profile, wand span
- Wand span: 0.345m
- Camera profile: ~/Software/3DPTV_Processing_Pipeline/Camera profiles
- Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5
- Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow/calibration_proc_folder

Refine Calibration Profile

- Refine calibration profile
  - Sort calibration points by the residual, remove results that are significantly larger
  - Use “cleanEasyWand.m” in “easyWand5/easyWand to OpenLPT ” to filter results
  - When desired Wand score is achieved, save results (.mat file) when the wand score and error meet the desired criteria

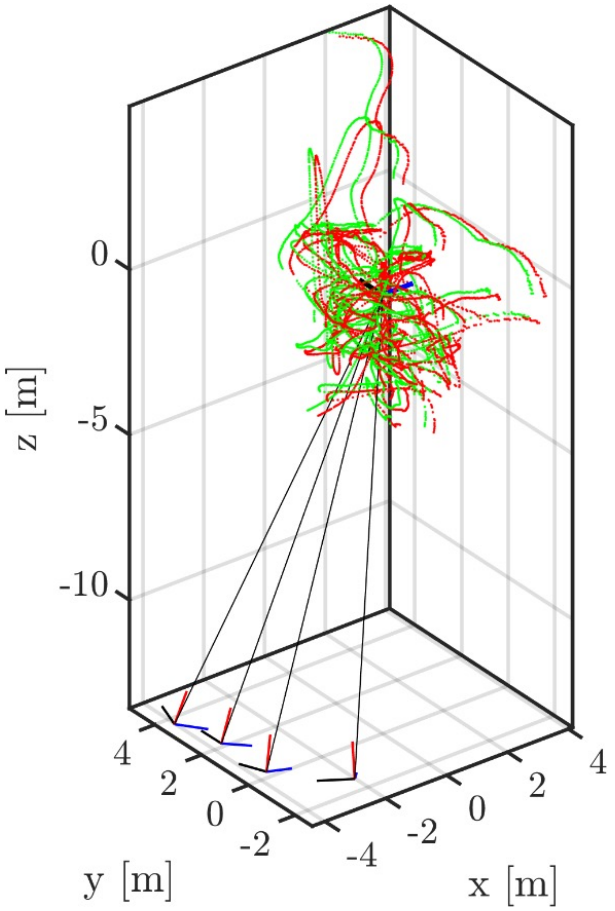




Notes:
- Should target for wand score ~1 and pix error <1
- Output: .mat file from easyWand calibration
- Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
- Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow/calibration_proc_folder

Calibration Results

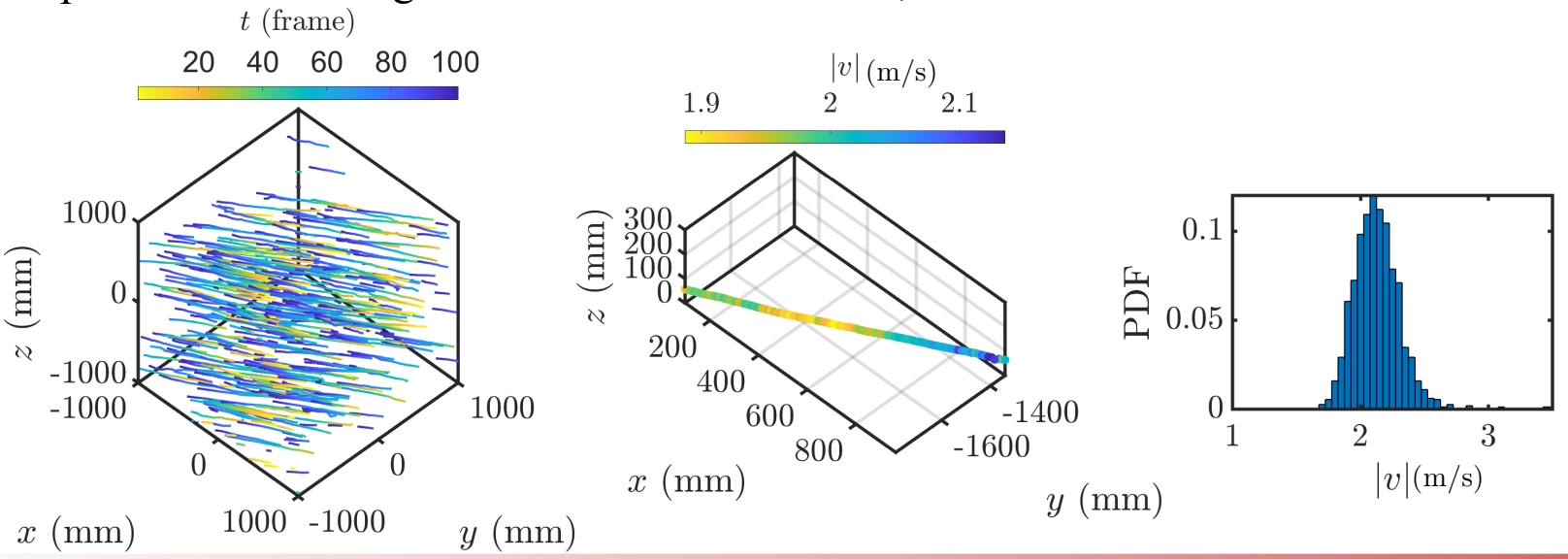
- Calibration result analysis
  - Wand quality score: 1.48
  - Wand end point standard deviation: 3.6 mm (total wand length: 34.5 cm)
  - Reprojection error of each camera
    - Cam 1: 0.33 pix
    - Cam 2: 0.37 pix
    - Cam 3: 0.36 pix
    - Cam 4: 0.34 pix
- Wand score of 1 or less indicates a good calibration (STD/mean = 1%)
- Run code "easyWand2OpenLPT.m" to convert coordinate system and export config files for OpenLPT



Notes:
• Input: .mat file from easyWand calibration; Output: config files for OpenLPT
• The code also converts the coordinate system
• Every 10th image was included in the data to reduce size. This subset is sufficient for calibration, with minor result variations possible.
• Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
• Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow settling/calibration_proc_folder

Trajectory Visualization and Post-processing

- Process images using OpenLPT
  - Create project folder for OpenLPT, organize .tif images into the /cam*/ folders
  - Convert easyWand results to OpenLPT configuration files (Use “easyWand2OpenLPT.m” in “easyWand5/easyWand to OpenLPT ”)
  - Run OpenLPT (in command window, enter “path-to-ShakeTheBox\ShakeTheBox.exe” “path-to-data\OpenLPT_proc\trackingconfig.txt”)
  - Results are saved in “Tracks” folder as .txt files
  - Further process the tracking data to extract information; additional details are available in the paper.



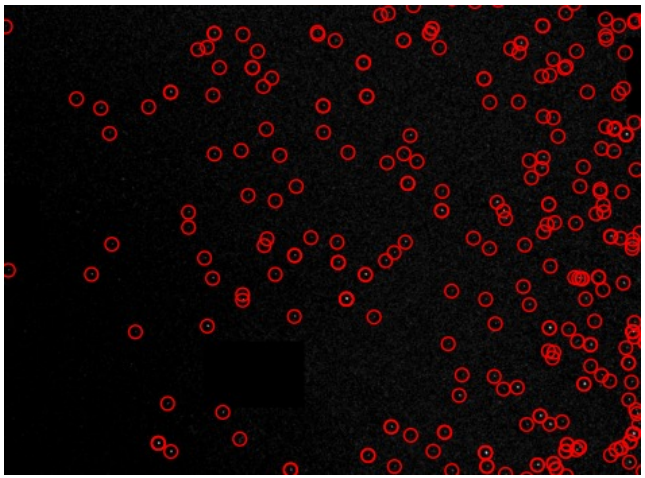
Notes:
• Code: ~/Field3D_PTV/Demonstration cases/Case II_Snow settling/3D trajectory visualization
• Data: ~/Field3D_PTV/Demonstration cases/Case II_Snow settling/results_fig
• Reference: Bristow, N., Li, J., Hartford, P., Guala, M., & Hong, J. (2023). Experiments in Fluids, 64(4), 78.

Case III: Pollen Dispersion

Experimental Setup

- Investigation into pollen particle emission from a cottonwood tree
  - Deployments of the system done over Summer 2022 during a time of high pollen density
  - Used 3D PTV system to obtain particle tracks of cottonwood pollen during the daytime
  - A better understanding of pollen emission can potentially improve forecasting for people with allergies





Notes:
• Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion

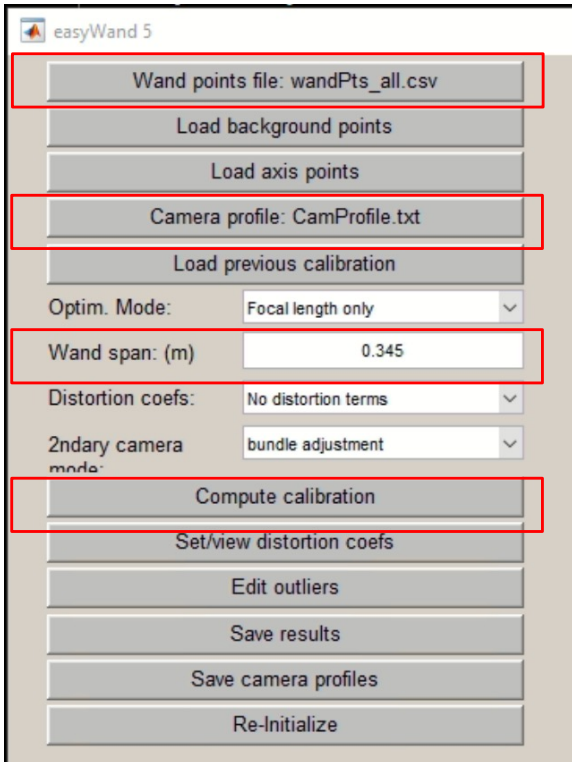
Extract marker coordinates from raw calibration images

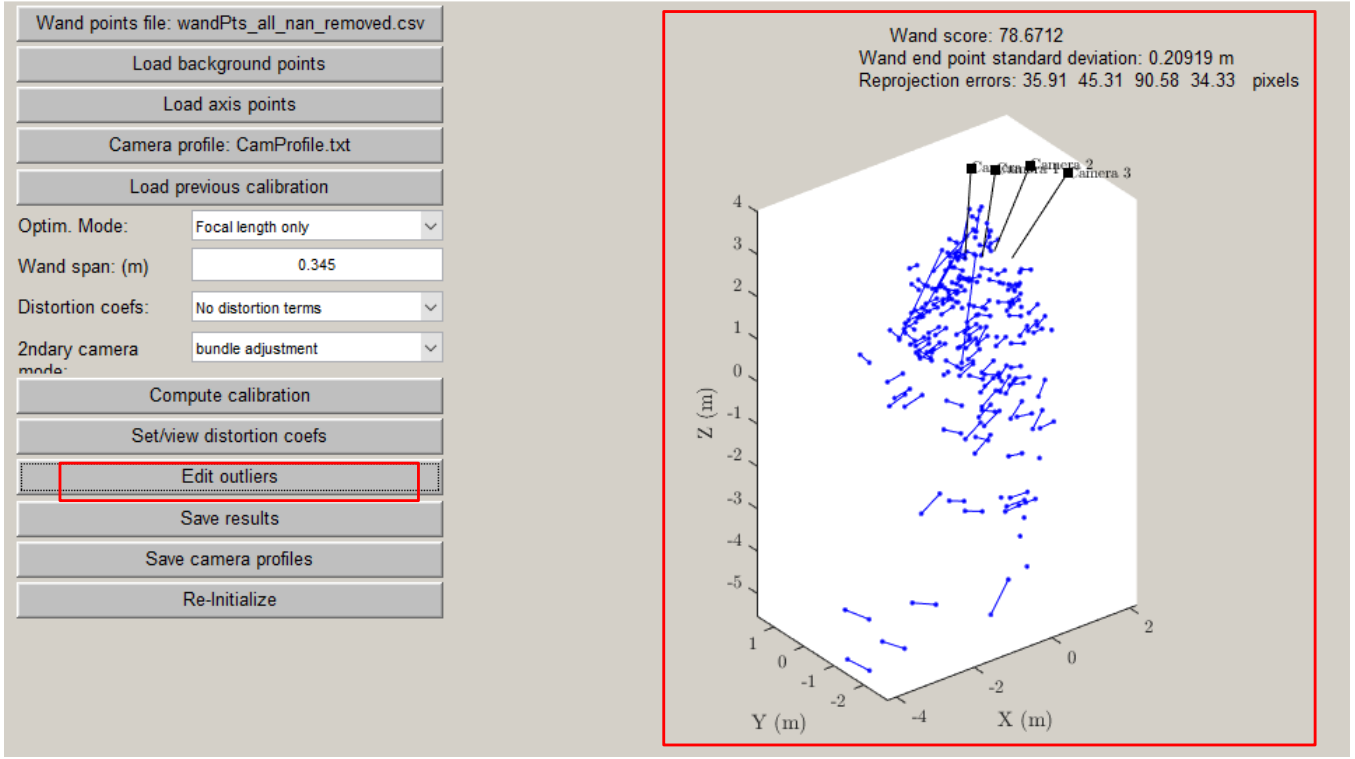
- Extract marker coordinates from raw calibration images
  - Find “droneMarkerTracking_v3.py” in “calibration image pre-processing” folder
  - Select path to raw calibration images and run the code
  - Marker coordinates are exported as .csv files for easyWand calibration in the next step

Notes:
• Inout: raw marker image; opt put: wand point coordinates
• Color setting: Calibration images are RGB for identifying red and green dots, tracking images are bw
• More details in “Data Processing Procedure”, the code generates .avi file with marker noted for verification
• 1 out of every 10 images was used to reduce the pack size. This reduced set has been shown to be sufficient for identifying the calibration matrix.
• The calibration working folder, named “calibration_proc_folder,” is included in each demo case.
• Code: ~/Software/3DPTV_Processing_Pipeline/calibration image pre-processing
• Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/calibration_proc_folder

Compute Calibration Profile

- Compute calibration profile
  - Run “easyWand5” in Matlab, load wand points and camera profile (Decimation2X), enter wand spam, compute calibration
  - The first result is with large error, use “Edit outliers” to refine results

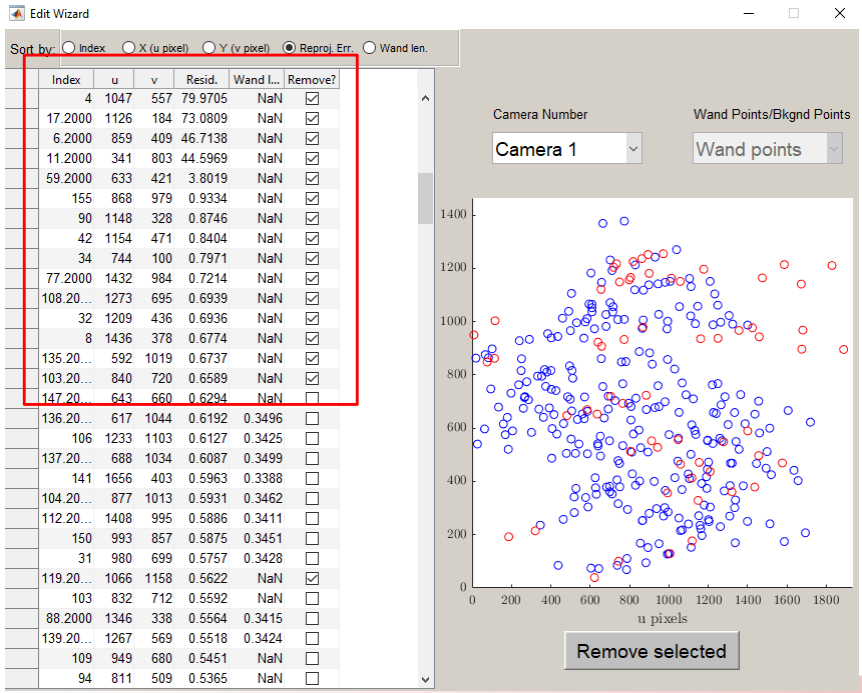


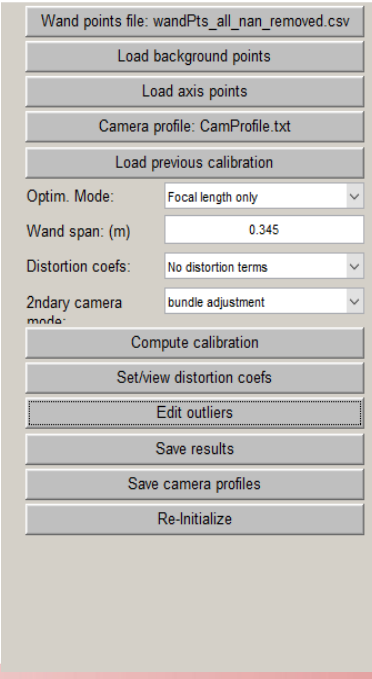


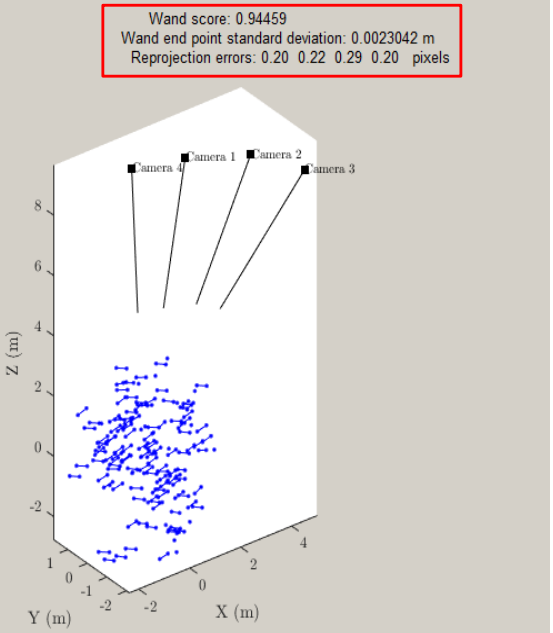
Notes:
- Input: wand point coordinates, camera profile, wand span
- Wand span: 0.345m
- Camera profile: ~/Software/3DPTV_Processing_Pipeline/Camera profiles
- Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5
- Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/calibration_proc_folder

Refine Calibration Profile

- Refine calibration profile
  - Sort calibration points by the residual, remove results that are significantly larger
  - Use “cleanEasyWand.m” in “easyWand5/easyWand to OpenLPT ” to filter results
  - When desired Wand score is achieved, save results (.mat file) when the wand score and error meet the desired criteria



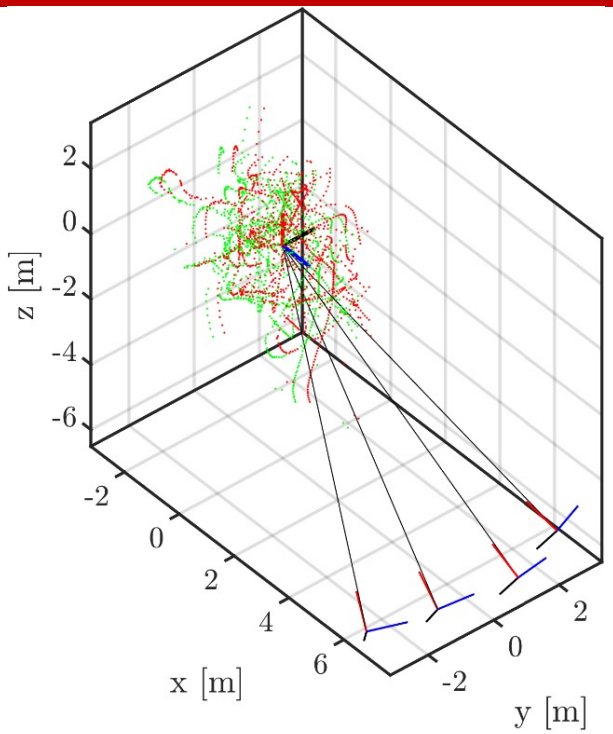




Flow Field Imaging Lab

Notes:
• Should target for wand score ~1 and pix error <1
• Output: .mat file from easyWand calibration
• Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
• Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/calibration_proc_folder

Calibration Results

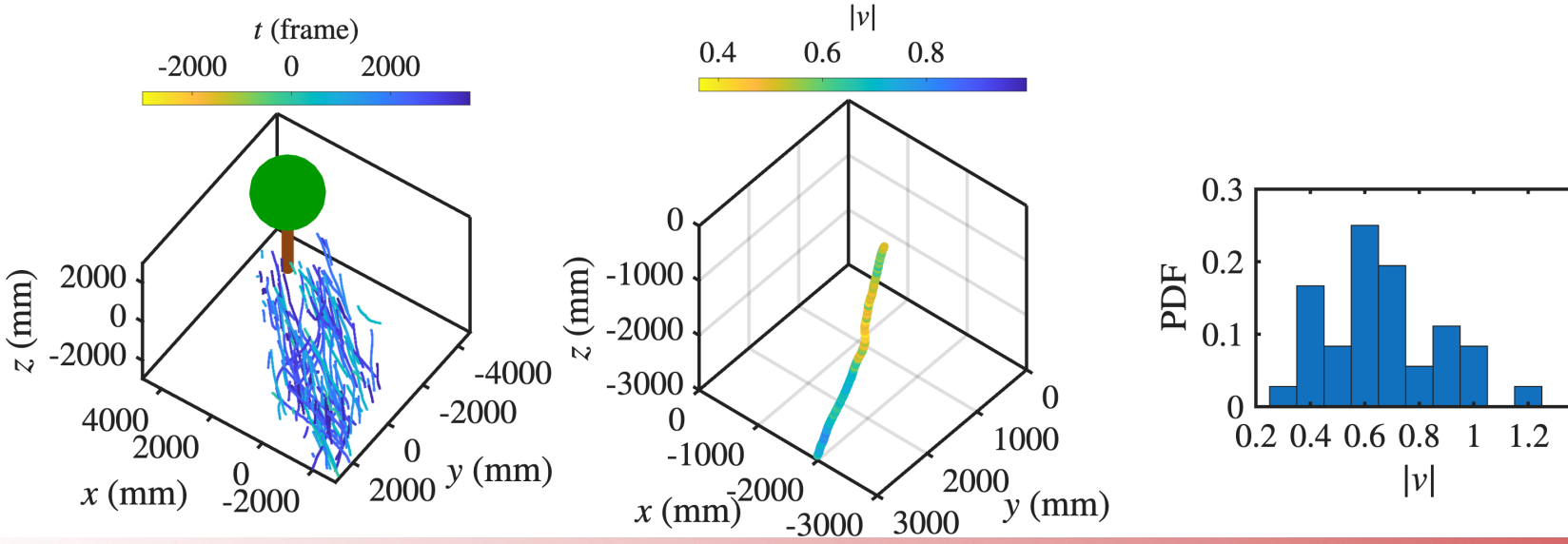


- Wand quality score: 0.94
- Wand end point standard deviation: 2 mm (total wand length: 34.5 cm)
- Reprojection error of each camera (Cam 1: 0.2 pix; Cam 2: 0.29 pix; Cam 3: 0.29 pix; Cam 4: 0.20 pix)
- Wand score of 1 or less indicates a good calibration (STD/mean = 1%)
- Run code “easyWand2OpenLPT.m” to export config files for OpenLPT

Notes:
• Input: .mat file from easyWand calibration; Output: config files for OpenLPT
• The code also converts the coordinate system
• Code: ~/Software/3DPTV_Processing_Pipeline/easyWand5/easyWand to OpenLPT
• Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/calibration_proc_folder

Trajectory Visualization and Post-processing

- Process images using OpenLPT
  - Create project folder for OpenLPT, organize .tif images into the /cam*/ folders
  - Convert easyWand results to OpenLPT configuration files (Use “easyWand2OpenLPT.m” in “easyWand5/easyWand to OpenLPT ”)
  - Run OpenLPT (in command window, enter “path-to-ShakeTheBox\ShakeTheBox.exe” “path-to-data\OpenLPT_proc\trackingconfig.txt”)
  - Results are saved in “Tracks” folder as .txt files
  - Further process the tracking data to extract information; additional details are available in the paper.



Notes:
- Tree shape and location for illustration only
- Due to file size constraints, the demo case includes 200 raw image frames for user testing. Tracking results for these 200 frames are included.
- The full tracking results are also included for visualization demo purposes
- Full tracking results location: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/OpenLPT_proc_folder/Tracks_Full
- Data: ~/Field3D_PTV/Demonstration cases/Case III_Pollen Dispersion/3D trajectory visualization