Concept: one of the inputs of parafoil control theory. Detect collapse of wing relative to ATV/camera

- High frame rate for machine vision - Mario
- **Goals**:
    - procure jetson + camera hardware
    - Setting up a paraglider inflation test
        - Fix parafoil wing
        - Look into setting up a parafoil test stand
    - Apply hassans MV software to paraglider project
    - Include paraglider centroid detection into MV software

Linear Actuator Control:
Concept: response/control mechanism of parafoil
Current State:
Acquire raspberry Pi for LAC

Summer Weekly Updates/Progress

Machine Vision Footage: Try to use more footage/pictures from paraglider wings, compare the footage with other pieces and find best results

Plan for Improvements: Improvements on the masking program with morphological operations to help with the specks/white noise. Address the graininess and holes, contour filters to help focus on segmentation

Improved Segments: color segmentation, white noise, contour detection, more visual for image detecting

Program Includes:

- Global Variables and Helper Functions
    - Store IMG.# (set up for global access)
    - Create access to IMG.#
    - Show values on video_capture for color tuning
- Main code
    - Set current_hsv as global function
    - Store IMG.# (Paraglider Footage)
    - Create VideoCapture on video_path AND return function if not opened successfully
    - Print (Video Capture Frames) for looping process
    - Set ‘q’ to quit looping process and ‘space bar’ to pause/resume video footage
    - Morphological Kernel Operations (LINK) Intro to OpenCV (LINK)
- Footage Loop Processing

- while running:
    if not paused:
        (Loop continuously to process each frame of the video while video is running)

- Current_hsv = hsv # read a single frame from IMG.# The video is read in (HSV) Hue Saturation Value.

- Np.array # HSV ranges from 0-255

- cv2.inRange #Create **Binary Masks** based on paraglider color ranges.

- Mask_red_cleaned = cv2.morphologyEx()
- Mask_yellow_cleaned = cv2.morphologyEx()
- Mask_orange_cleaned = cv2.morphologyEx()
- # Morphological Closing/Opening Clean up the masks using morphological operations (LINK).

- Contours_red, _ = cv2.findContours()
- Contours_yellow, _ = cv2.findContours()
- Contours_orange, _ = cv2.findContours()
- # **Filter** out small contours that are likely noise or specks.

- min_contour_area
- filtered_contours_red = [ ]
- filtered_contours_yellow = [ ]
- filtered_contours_orange = [ ]

- Result_masked_original = cv2.bitwise_and(frame, frame, mask = mask_combined)

Step 10:
Wait for a key press to handle pause/resume or quit commands.
cv2.waitKey(25)

- Helper Functions
    - cap.release()
    - cv2.destroyAllWindows()

Fall 2025 GM #1

*** Machine Vision Footage**: Try to use more footage/pictures from paraglider wings, compare the footage with other pieces and find best results

Plan for Fall: Use NEW footage for Machine Vision, Implement Adaptive Thresholding (Auto Track), Object tracking using **Kalman Filter**, Log Mission Data

- Reliable autonomy for air/ground terrain
- Stage 3 on GoAero
-

Workday #1

- Intro to ASCEND Team

Workday #2

- Jetson Orin Nano Setup

Workday #3

- Jetson Orin Nano setup + camera set up

Workday #4

- Setup for Jeson Orin Nano failed - looking forward to someone else fixing Jetson setup

Workday #5

- Look into camera set up, and necessary library packages. Tested old jetson on Monitor

Workday #6

- No Workday

Workday #7

- Jetson Orin Nano NVIDIA boot completed by Mario.
- Nanocamera updated to Jetson. Must download VSCode, capture live-footage with CSI camera.
- Imported Nanocamera module to start the CSI Camera installation
- Must complete by next workday: setting up VSCode, register Jetson into utexas-iot WIFI, install Nanocamera module, set up CSI Camera live footage, capture live footage with already made program.
- Research Started: MV Paraglider Collapse Detection - Docs

Workday #8

BEFORE

- Set up VSCode and CSI Camera footage
- Complete Research
- Repair Paraglider?

AFTER

- Research - MV Paraglider Collapse Detection
- Uninstalled: opencv-python, opencv-contrib-python, opencv-python-headless
- Re installed: opencv-python, opencv-contrib-python, opencv-python-headless

Workday #9

No Work Day SHPE National

Workday #10

- CSI Camera Set Up Complete
- VS Code complete
- Run Camera on Terminal to capture live footage
    - `sudo /opt/nvidia/jetson-io/jetson-io.py`.
    - Configure 24-pin
    - Configure
    - Configure IMX 219 C
    - Save and Reconfigure
    - LIVE CAMERA FOOTAGE

Downgraded numpy to function with CSI Camera
ParagliderMasking.py debugging to function with different masking code
Research battery pack, paraglider collapse detection, live footage detection

Workday #11

- Live footage detection (track and identify different hue saturation w/ live footage)
- Research battery pack - portable chargers
- Paraglider collapse detection
- Integrating machine learning Reinforcement Learning (RL)

Research - Linux vs. Unix
New jetson - ascend
Old jetson - MASSlab

- 120 fps
- Video,
- Check installation of cv
- G-steamer to open camera
- AutoFocus testing
- Guthublinks.txt - shell commands

Starting camera on simple_camera.py

- Python3 and python simple_camera.py does not work
    - Reboot with 10-second fix - sudo services nvargus-daemon restart
- Tried python3 simple_camera.py but showed same error
- Ran v4l2-ctl - -list-devices to verify hardware connection and imx477 is connected, NOT imx219
- Now checked if GSteamer captures camera
- Checked if camera is even working

- Now checking because both GStreamer and camera function Force restarting camera camera test and trying fakesink again
- "dmabuf_fd -1" means the camera service is failing to get a valid memory buffer from the system [meaning nvargus-daemon is broken]

Workday #12:

- Since camera is connected and verified (v4l2-ctl - - list-devices) but service is failing: **Restart the Camera Service**
    - Bash: sudo service nvargus-daemon restart
- **Verify GStreamer Pipeline:** Ensure your GStreamer pipeline string is correct for your **IMX477 camera** (not IMX219). A robust GStreamer pipeline for a CSI camera on the Jetson often looks like this:
    - CURRENT INPUT DISPLAY IS NOT SUPPORTED

Verification & Testing

1. Check Device: After rebooting, check if the camera is detected: ls /dev/video*. You should see /dev/video0 or similar.
2. Test with GStreamer: Use a command like this to stream video (adjust resolution/framerate as needed):

BASH:
gst-launch-1.0 nvarguscamerasrc ! 'video/x-raw(memory:NVMM),width=1920, height=1080, framerate=30/1' ! nvvidconv ! nvegltransform ! nveglglessink -e

CAMERA WORKS!!!
IMX 219A - 477C

Spring 2026 GM #1

Workday #13:
Intro + Koopman Operator Based Program

- Create software pipeline
- Integrate MV with live footage