

**ReadMe for the RMC Video Sync Tool**

This project was created to streamline the process of analyzing real world DELTA motion systems with the use of an RMCTool’s csv file and synced video footage.

Version: 1.0 | Last Updated: April 1, 2021 | Supports: Video Formats (.mp4, .avi, .mov), File Format(s) (.csv)

## Contents

[Resources 2](#_Toc68190714)

[Libraries 2](#_Toc68190715)

[Initial Procedure 2](#_Toc68190716)

[How it Works 3](#_Toc68190717)

[Using the Program 4](#_Toc68190718)

[Known Issues 5](#_Toc68190719)

[Future Changes 5](#_Toc68190720)

# Resources

**main.py****–** Project executable that initializes the sizing of GUI elements along with callback handling for plot data and button press event handling

**ui\_main.py****–** Contains the GUI framework for the main window

**ui\_functions.py****–** Handles the main functions of the program, this includes data parsing, plot drawing, user input event handling, and GUI animation

**ui\_loading.py****–** Processes video data using OpenCV and commits it to a GUI compatible list of frames

# Libraries

**Matplotlib –** Library that handles everything plot related; it provides an object-oriented API for embedding plots into applications

**OpenCV****–** Library that handles everything video related

**PySide2****–** Library that handles everything GUI related, it is a Python binding of the GUI toolkit Qt

**NumPy****–** Library that manages plot data and array manipulation

# Initial Procedure

1. **–** Import the video file and the RMCTool’s plot file

*Note: make sure the video file is synchronized with the data file*

1. **–** Select between one and three data parameters to be displayed

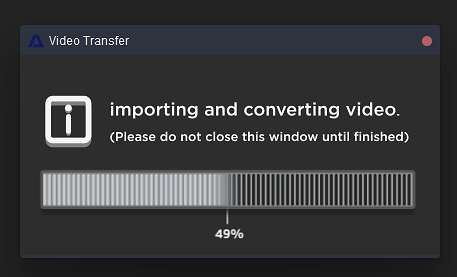
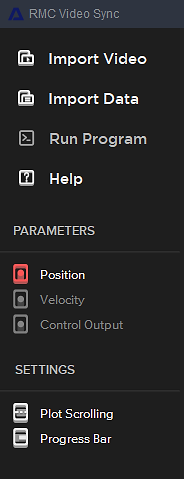
*Note: this cannot be changed after ‘run program’ has been executed, additionally, selecting more parameters decreases performance*

1. **–** Select whether axis scrolling is to be applied

*Note: axis scrolling reduces the overall smoothness of the plot as it must redraw every plot element, i.e. blit enabled. By disabling axis scrolling, only the elements found In the artist packet will be redrawn*

1. **–** Run the program and wait for the video to be uploaded and converted
2. **–** Control the program using the arrow keys, spacebar, and click regions

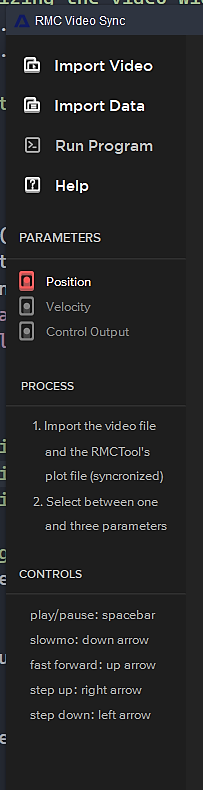
# How it Works

1. **–** When the program begins, main.py will animate menu items and grab screen geometry information from the operating system to properly size the video footage
2. **–** When Import Video is selected in the side menu, main.py branches to ui\_loading.py which then spawns a sub-program (Video Transfer) that collects information regarding the footage’s frame rate and frame count. While the footage is being uploaded, a loading bar tracks the progress and the Import Video icon in the main window subtly rises and falls to let the user know it is being processed. Along with this, an array of QT compatible frames is stored and sent to ui\_functions.py to be used in the main window (RMC Video Sync). If the footage was successfully uploaded, the Import Video button will illuminate. Note that the main window is inactive during this process as this entire function is handled by the Video Transfer sub-program

*ui\_loading’s Video Transfer window*

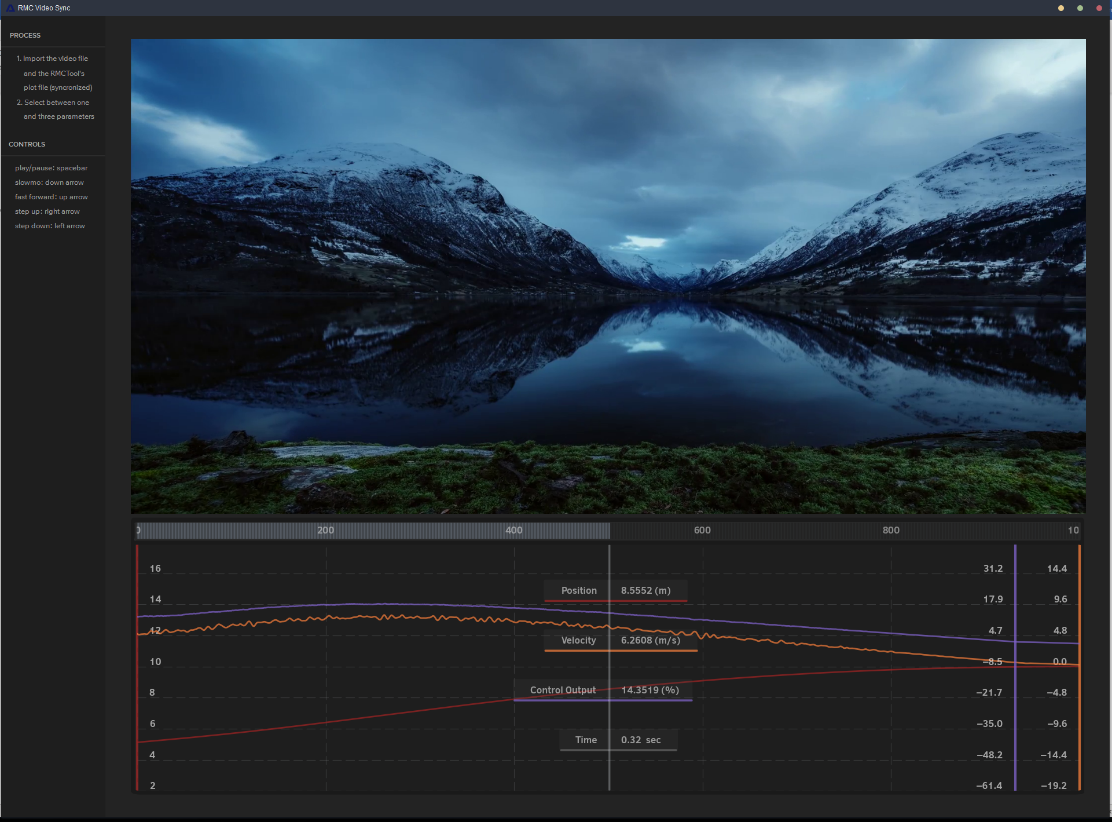
1. **–** When Import Datais selected in the side menu, main.py branches to ui\_functions.py which then handles all the data parsing using python’s csv library along with NumPy. If the file was successfully uploaded, the Import Data button will illuminate, and the Parameters sub-menu will provide a list of parameters that can be displayed on the plot viewer
2. **–** When Helpis selected in the side menu, a Process, and Controls sub-menu will appear describing how to use the program along with the controls
3. **–** If axis scrolling is enabled, the entire plot viewer will be updated with every data increment. If axis scrolling is disabled, the plot is animated using the artist method which increases overall performance, as it only updates the drawn line and on-screen text.
4. **–** If the progress bar is enabled, a visual of the current position in the data is displayed behind the x-axis at the top of the plot viewer

*RMC Video Sync side menu*

1. **–** If the progress bar is enabled, a visual of the current position in the data is displayed behind the x-axis
2. **–** Once the Import Video and Import Data buttons have illuminated and at least one parameter has been selected from the drop-down list, the Run Program button can be pressed to initiate the data/video analysis
3. **–** Packets of data are sent using a signal-slot mechanism, this includes data surrounding the current position, both before and after to draw a centered plot.
4. **–** The data\_send\_loop function found in ui\_functions.py, is checking for user input and adjust the signal frequency, video frame output, and data position. Zooming out will increase the data packet size from 201 entires per frame to 1001 entries per frame

*RMC Video Sync help dropdown*

# Using the Program

**play/pause –** spacebar

**slow motion****–** down arrow

**fast forward****–** up arrow

**step forward****–** right arrow

**step backward –** left arrow

**hide labels****–** right click on the left side of the plot, where the labels are displayed

**zoom out****–** right click on the right side of the plot for a 5x zoom outward

*Plot Viewer and Video Display*

# Known Issues

**Parameter drop down****–** when selecting parameters, if they are rapidly pressed or dragged, false selections may occur

**Video Transfer window****–** closing the video transfer window results in the main window being locked until the file is processed, preferably the transfer would simply end if the window is closed before the video transfer has completed

**Step forward and back****–** after altering the method of plotting, the step forward and step backward function does not work as intended, holding down the left and right arrows will scroll the plot in those directions, but pressing them once will not advance the plot, only the video

**Occasional crash window****–** this happens rarely, currently the cause is unknown

# Future Changes

**Smoother plot scrolling****–** if this is to be executed, it will most likely result in sourcing another library for animated plot functionality

**Slider bar control****–** likewise, if this is to be executed, it will most likely result in sourcing another library as the Matplotlib library cannot smoothly execute all functions without noticeable choppiness

**More thorough file inspection****–** this includes checking for missing pieces of data, corrupt files, and representing higher levels of detail when plotting, this will also including testing a variety of different files

**On screen feedback****–** originally, play, pause, fast forward, slow motion, and other icons appeared in the side menu when these playback methods were activated, however because this causes the focus to be drawn to the side menu of the program rather than the video/plot, displaying these transparently on the video will be far more intuitive

**Live settings options change****–** this feature will enable/disable the plot scrolling feature and progress bar while plotting

**Live parameter selection –** this feature will allow the user to hide/show parameters while plotting

**Support for multiple data array axis –** axis 0 is only currently supported

**Better keyboard detection –** the current method of handling keyboard inputs is fairly barebones