1. **Installation**
   1. Python components
      1. Anaconda 2.7
      2. OpenCV 3
      3. WX
   2. Go to <https://github.com/laughreyl/PySolo-LL>
2. **Computations**
   1. Aside from the obvious changes to the GUI, I have replaced the distance calculations.
   2. Rather than shrinking the image prior to processing, all calculations are done using the original image size. Conversion to millimeters is done based on the user's input regarding the actual size of the camera's field of view.
   3. Rather than process the whole image at once, this program processes each ROI separately in order to maximize contrast.
   4. The threshold for converting to black & white is selected by finding the lowest threshold that produces one more contour than an empty cell. That contour is assumed to be the fly.
   5. A box is fitted to the contour, and the middle of the box is selected as the position of the fly.
   6. The distance is calculated between the position of the fly in one frame compared to the previous frame. Distances less than 3 pixels are ignored.
   7. Condensation or anything other than the fly, that causes movement in the ROI can be misinterpreted as fly movement.
3. **On start**, a file selection dialog window will open so you can select a saved configuration.
   1. If you don't have one you want to start with, you can cancel the dialog and a generic configuration will be created, but not saved.
4. **Setting up the** Configuration
   1. The program will open on the main **configuration** page.
   2. There are **tabs on the left side** that allow you to switch to monitor pages.
   3. **Load a configuration** at any time by typing in the name or using the browse button.
   4. **Save Configuration** - saves the configuration and all monitor settings to a file.
   5. **Add Monitor** - adds another monitor to the configuration
      1. a maximum of 9 monitors is allowed).
      2. The new monitor will be a copy of the last monitor in the list.
   6. **Start Acquisition** - uses current configuration to collect data.
      1. **It is essential to have the correct settings for the source size and source fps** for each monitor in order to get accurate distance calculations.
      2. **The output** is a collection of files with the names shown in "Output Prefix" on the table, each with 32 ROIs, and numbered consecutively.

e.g. for output prefix "c:\pathname\Monitor1":

"c:\pathname\Monitor11.txt" will contain data for ROIs 1-32,

"c:\pathname\Monitor12.txt" will contain data for ROIs 33-64, etc.

* + 1. If files with the output names already exist, you will be asked for a new prefix.
       1. The name of the monitorthat you are choosing a prefix for is on the label at the top of the save file dialog window.
       2. You cannot overwrite a file. If you want to reuse the prefix, delete the old files.
    2. There may be long pauses during Acquisition from time to time. Information appearing on the consoles will let you evaluate progress.
  1. **View Monitors**  - returns to monitor panels after acquisition.
  2. **DAMFileScan 110X** - runs the DAM FileScan program in a separate window.
  3. I was unable to add a button to start **SCAMP** since I don't have MATLAB at home. Code for the button is in the cfgPanel.py file, and is commented out.

1. Click on a monitor tab on the left side to open a **monitor configuration page**
   1. Two types of **sources** are available.
      1. To change the source, click on a field or button and browse for or type the file name.
      2. The real-time camera option has been commented out because there is no end of video that would cause Acquisition to stop.
   2. A different **mask file** may be loaded by browsing or typing in the file name.
   3. The **output folder** is the folder where the acquired data will be saved.
   4. **Time Settings** refer to the date and time that the camera started recording.
   5. **Tracking parameters** - I've only maintained code for distance tracking, and all monitors in the configuration will be tracked.
      1. A panel of controls for tracking is in the maskPanel.py code and commented out if someone wants to add them back.
   6. **Mask Generator**
      1. The diagram shows what each of the terms means.
      2. Clicking on the video image will change the "Top Left" settings to the coordinates of the pixel that was clicked. These numbers are based on the full sized image, not the image shown on the screen.
      3. After entering the mask's values, click the generate mask button to show the mask on the video.
      4. To save the mask use the save mask button.
   7. The preview **font** changes the size of the numbering above the top row of the ROIs. The font size is not related to typesetting font sizes. Best settings will vary based on the size of the original image and size of the preview frame.
   8. Changing the **frame size** and **preview fps** only affects the image on the screen. All tracking is done using the full size image.
   9. The **ROI line thickness** changes the thickness of the lines displayed on the video but has no effect on data acquisition.
      1. If your ROIs seem to have missing parts, increasing the line thickness will probably resolve that.
   10. **Watch Video Tracking** will show images used during acquisition.
       1. For each ROI you will see the original image, black & white image, and contour drawings on your screen for 2 seconds.
       2. Press "p" to pause on the current image, "q" to quit viewing images, and "s" to save the current image.
       3. This settings drastically increases processing time due to the 2 second wait between frames. Leave it off, or quit to acquire data in a timely fashion.
   11. **Save Configuration** button saves all monitor configurations to a file.
   12. **Delete Monitor** - removes the monitor from the configuration.