**GenerateTimingFigure.m Script Overview**

The `**GenerateTimingFigure**.m` script is designed to visualize and assess the temporal precision of data recorded from light sensors. It processes the data, identifies unexpected frame sequences, plots the light sensor data, and highlights points of temporal imprecision. Below, I provide a detailed description of the key fields output into `**figureData**`, the algorithms used to extract these fields, and guidance for navigating the scripts.

**Key Fields in figureData**

1. **analog\_orig**:

- Description: Original analog data from the light sensors.

- Extraction: Directly retrieved from the input data (`serialRecvData`).

2. **DetectedFlipsL** and **DetectedFlipsR**:

- Description: Details of detected frame flips (transitions) for the left and right light panels.

- Extraction: Obtained by detecting peaks and troughs in the light sensor data (analog\_orig.LightL/LightR) using the `**FindFrameTransitions**` function.

3. **DetectedFramesL** and **DetectedFramesR**:

- Description: Classification of frames as valid or invalid based on expected patterns for the left and right light panels.

- Extraction: Processed using the `**FlashPanelPatternProcessor**` function, which matches frame sequences to expected patterns and identifies deviations.

4. **exceptionDetailsL** and **exceptionDetailsR**:

- Description: Details of any exceptions found during the frame classification process.

- Extraction: Generated within the `**FlashPanelPatternProcessor**` function as frames are classified and exceptions are detected.

**Algorithms**

**ProcessFlashPanelData**

1. Function: `**ProcessFlashPanelData**`

- Purpose: Processes the analog data recorded from light sensors to extract and analyze frame onset data.

- Algorithm:

- Load and prepare the data.

- Identify frame flips by detecting peaks and troughs in the light data (`**FindFrameTransitions**`).

- Upsample the data to achieve higher temporal resolution.

- Match the detected flips to the expected pattern using `**FlashPanelPatternProcessor**`.

**FindFrameTransitions**

1. Function: `**FindFrameTransitions**`

- Purpose: Detects transitions between frames in the light sensor data.

- Algorithm:

- Normalize the light data.

- Slide a window over the data to detect peaks (indicating frame transitions).

- Classify transitions as trough-to-peak (indicating a rise in light intensity) or peak-to-trough (indicating a fall in light intensity).

**FlashPanelPatternProcessor**

1. Function: `**FlashPanelPatternProcessor**`

- Purpose: Classifies frames as valid or invalid based on expected patterns.

- Algorithm:

- Generate a matrix of all possible orders of the expected pattern (`patternTemplate`).

- Initialize indices and storage for frame details and exceptions.

- Loop through the frames to find matches (`**FindNextMatch**`) or identify exceptions (`**FindNextException**`).

- Assign `**FlipIndex**` to map frames back to their original indices.

**FindNextMatch** and **FindNextException**

1. Function: `**FindNextMatch**`

- Purpose: Identifies the next segment of frames that match the expected pattern.

- Algorithm:

- Compare segments of the data to columns of the `patternTemplate`.

- Update `frameDetails` with validity and pattern column information.

2. Function: `**FindNextException**`

- Purpose: Identifies exceptions in the frame sequences.

- Algorithm:

- Compare segments of the data to smaller patterns derived from `patternMatchers`.

- Record exceptions in `exceptionDetails`.

**FindClosestWithWindow**

1. Function: `**FindClosestWithWindow**`

- Purpose: Finds the closest time points in the upsampled time vector that match the original detected flip times.

- Algorithm:

- Loop through each flip time to find the closest corresponding time point in the upsampled time vector within a defined window.

**FindIdealFlipTimes**

1. Function: `**FindIdealFlipTimes**`

- Purpose: Corrects the upsampled flip times to align them with the expected frame rate.

- Algorithm:

- Use an iterative process to adjust the detected flip times to minimize the deviation from the expected frame duration.

**GenerateDiscretizedFrameVector**

1. Function: `**GenerateDiscretizedFrameVector**`

- Purpose: Converts corrected flip details into a discretized frame vector for pattern matching.

- Algorithm:

- Create a vector where each frame is represented as valid or invalid based on its status and original indices.

**Navigating the Scripts**

1. Start with `**GenerateTimingFigure**.m`:
   1. This script is the entry point and calls other functions to process the data and generate the figure.
2. Understand Data Processing in `**ProcessFlashPanelData**.m`:
   1. This function processes the analog data and prepares it for further analysis.
3. Frame Classification in `**FlashPanelPatternProcessor**.m`:
   1. Learn how frames are classified as valid or invalid, and how exceptions are handled.
4. Detection Functions:
   1. **FindFrameTransitions**.m: Understand how frame transitions are detected from the analog light data.
   2. **FindNextMatch**.m and **FindNextException**.m: Understand how patterns are matched or exceptions are identified.
5. Upsampling and Correction Functions:
   1. **FindClosestWithWindow**.m: Learn how the closest time points in the upsampled time vector are found.
   2. **FindIdealFlipTimes**.m: Understand how the upsampled flip times are corrected to match the expected frame rate.
6. Discretizing Frames:
   1. **GenerateDiscretizedFrameVector**.m: Learn how corrected flip details are converted into a discretized frame vector for pattern matching.

Example Workflow

1. Load and Prepare Data:

- Load the raw analog data from the light sensors from the processed serialRecvData.

- Extract synchronization timestamps.

2. Detect and Correct Flip Times:

- Use `**FindFrameTransitions**` to detect frame flips.

- Upsample the detected flips using `**FindClosestWithWindow**`.

- Correct the flip times using `**FindIdealFlipTimes**`.

3. Classify Frames and Identify Exceptions:

- Convert the corrected flip details into a discretized frame vector using `**GenerateDiscretizedFrameVector**`.

- Classify the frames using `**FlashPanelPatternProcessor**` and identify any exceptions.

4. Generate and Interpret the Figure:

- Use `**GenerateTimingFigure**` to create a plot that visualizes the light sensor data and highlights points of temporal imprecision.

By following these steps, you can effectively navigate the scripts, understand the relationships between the structs, and assess the temporal precision of the light sensor data.