RTES Project Goals and Objectives (MINIMUM, TARGET, STRETCH)

Please read the following general description of the RTES Project OBJECTIVES. Note that you must verify and analyze your system so that you can accurately estimate the real-time feasibility, safety and how predictable your response is for each test run.

Synchronome & Time-Lapse Image Acquisition GENERAL OBJECTIVES

Your system solution must acquire external clock synchronized images where all images show a unique state of the hour, minute and second hands on an ANALOG CLOCK using a standard resolution (or better) resolution USB camera.

From your acquired and saved image set, you should be able to encode an accurate time-lapse of the images where only unique non-blurry clock states are shown. It is required that you use a 320x240 resolution or higher digital camera as a minimum, that interfaces via USB to an embedded Linux system. Here are some detailed goals and objectives for the syncronome and time-lapse RTES Project:

1. Resolution must be at least 320x240, but VGA (640x480) or Standard Definition (720x480 or 640x480) or better is recommended as documented here. Other higher definition resolutions such as 1080p (1920x1080) or higher are allowed if you can find a camera that provides this.
2. You must acquire INDIVIDUAL frames from your camera - NOT an MPEG encoded stream. The frames must be acquired and then selected at 1 Hz for verification purposes (higher acquisition rates are fine, but 1 Hz and 10 Hz must be supported frame selection and storage modes). They should be PPM or PGM format, but options for PNG, JPG, or other compressed formats are ok to use if your frames are visibly time-stamped in a header or on the image.
3. Your camera and interface must be able to acquire frames faster than the 1 Hz rate (or 10 Hz for full credit) and **must continuously timestamp frames at a rate that is at least 20 to 30 frames/sec (20 Hz or better)** from which your **frame selection service samples for storage at 1 Hz (or 10 Hz)**. Your processing rates should ideally be based on an interval timer (hardware PIT or software timer services POSIX clock). Frame rates for cameras can vary quite a bit (and are affected by start-up and lighting) - please refer to documentation on your camera and frame rates, so part of the challenge is reliable acquisition of images over time.
4. Verification image file format must be PPM "P3" or "P6" RGB or "P2" or "P5" Graymap format as documented for Netpbm format - if your camera encodes into a different format, you will need to convert in your acquisition service so that you can apply simple machine vision algorithms to the raw frame data.
5. Your frame acquisition and timestamps should be accurate so that there are no "observable" errors in 1800+1 frames (or 1800 seconds of continuous operation over a wall-clock time-period of 30 minutes and 0 seconds). **Errors are** saved frames that have hour, minute, or **second hand (or 1/10th second for full credit) skips, repeats or blurs**.

6. Your timestamps must be embedded in the images themselves (e.g. use OpenCV "putText") or in the PPM or PGM header of EVERY FRAME as a comment field (using "#" prefaced line in your header).
7. Your HOST or TARGET platform "uname -a" output must be embedded in the image or the PPM header of EVERY FRAME as a comment field (using "#" prefaced line in your header) to clearly identify the system on which your test was run.
8. **You must pass the 1 Hz verification (no skips, blurs, or repeats) with an ANALOG CLOCK, and you must pass the full-credit STRETCH goal for 10 Hz verification with a DIGITAL STOPWATCH over 1800+1 frames (30 minutes for 1 Hz, 3 minute for 10 Hz).**

Synchronome & Time-Lapse Image Acquisition VERIFICATION MINIMUM OBJECTIVES

1. You must verify that your image acquisition is accurate enough so that you can observe an ANALOG EXTERNAL WALL CLOCK for a period of 30 minutes with high accuracy (no second hand skips, blurs or repeats), or EXACTLY 1801 frames and that you capture the second and minute hand display with no glitches (hand skips, blurs, repeats). So, for example, I should see 6:06 PM in the first frame (frame 0) and 6:36 PM in the last (frame 1800), which requires 1801 frames (0 to 1800). Your solution should have less than 1 second of error over the 30 minute period, you should count the initial frames that show the starting time and minute (e.g. 6:06 PM) and the number of frames that show the final stop time and minute (e.g. 6:36 PM) - the second, minute and hour hands should appear stationary in all frames. You must compute and plot your average frame jitter and drift and determine if you have truly monotonic processing and timing.
2. As an added test, you must ALSO observe some physical process like the melting of a cube of ICE alongside your EXTERNAL CLOCK so that it proves this is a true time-lapse sequence when you encode it. Put this in your final report.
3. You must encode your final video which includes the EXTERNAL CLOCK and the observed physical process into an MPEG2 or MPEG4 program stream using ffmpeg and include this on your final Zip files along with your PPM frames. If upload of all 1801 frames is not possible based on Canvas file size limitations, upload the first 181 and the last 181 frames.

Synchronome & Time-Lapse Image Acquisition TARGET GOALS

Please choose to add one or more of the following features to your project and re-run your verification:

1. Demonstrate that as an option, you can save frames at 10 Hz rate for 1800+1 frames over 180 seconds without any stability issues in your code (note that this must be glitch free for STRETCH credit, but here it just needs to run and work).
2. Image Processing (e.g. Canny or Sobel edge finder, PSF Sharpen, etc.) or Compression of frames (not using OpenCV) on your target so you can store more than 1800+1 frames and so that you have less to transfer over Ethernet. **If you do compression, and you use**

OpenCV built-in functions to compress, this will not be counted as a FULL CREDIT additional feature.

3. Continuous streaming or periodic automatic download of frames over Ethernet so that you can run indefinitely and never run out of space on your flash file system, which should maintain only the last 1800+1 frames.
4. Any other image processing features you can turn ON/OFF that you can dream up to show that you can capture unmodified and modified frames and/or stream them from your Linux embedded system. Other examples include target of interest detection and designation (adding graphics such as a bounding box), image enhancement (resolution up or down conversion), advanced transformations (e.g. Hough line transform) or image segmentation.
5. After adding one of the above features for real-time image processing, re-verify your frame jitter and accumulated latency as described above and describe and account for any differences you see. You should also be able to demonstrate how the increased load for your image processing service is handled on a specific AMP core and show loading with "htop" or similar profiling tool.

Synchronome & Time-Lapse Image Acquisition STRETCH-GOALS

Please add the following features to your project:

1. Run at a higher frame acquisition rate than 20 Hz with continuous download at 1 Hz or 10 Hz with your selective download of images with no glitches (skips, repeats, blurs) so the quality of your image selection can be assessed at both rates.
2. Verify that you can observe a DIGITAL CLOCK which shows 1/10th of a second elapsed time (as well as hours, minutes and seconds) from a device such as a stop-watch with no glitches (skips, repeats, or blurs).
3. After adding the 20 Hz or faster frame acquisition and 1/10th second frame selection and storage, re-verify your frame jitter and accumulated latency as described above and describe and account for any differences you see.
4. Any other frame by frame image processing you dream up that can run at 1 Hz AND faster 10 Hz rate beyond a basic compression or image transformation feature you already created to meet the INDIVIDUAL TARGET additional feature goal.
5. Ability to turn ON/OFF features one by one or in combination with analysis of the impact on jitter, drift, and monotonicity of your processing.