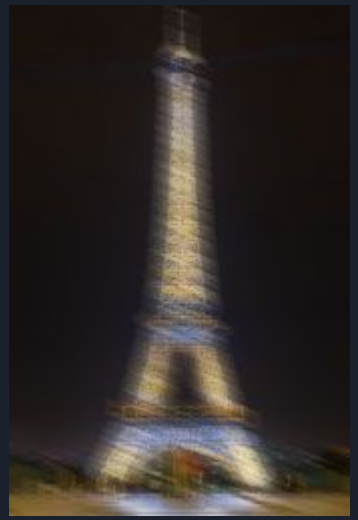
SharpCam

Group D1: Nathan Koch, Sean Pogorelc and Rebecca Jean-Louis

Introduction

- Application Areas: Signals and Software
- Goal
  - SharpCam allows a user to record a video and then have it post-processed on the fly to remove spatially invariant blur, caused by something like a shaking hand , so that the cameraman can have a clean video even under non-ideal conditions.

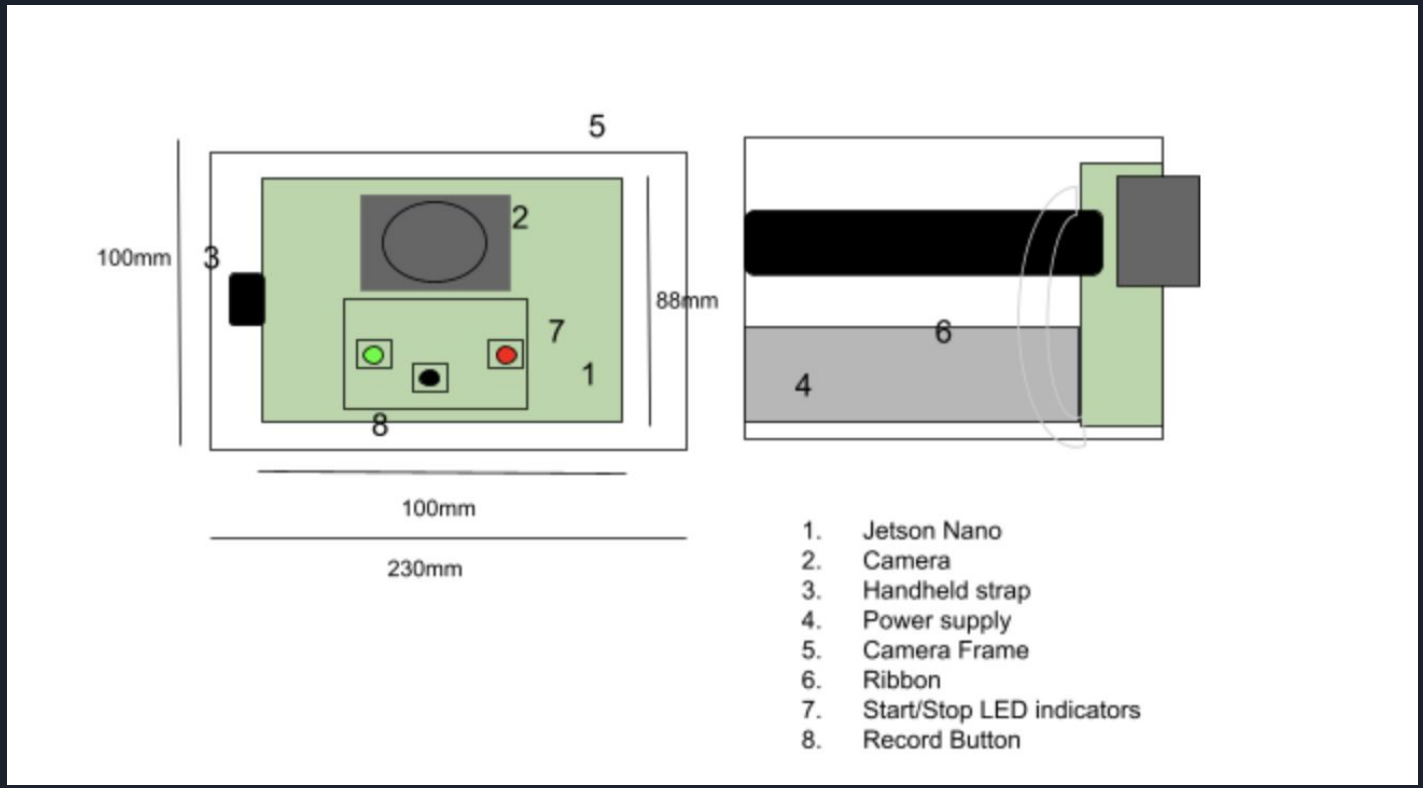


Implementation Approach (Hardware)

- Jetson Nano
  - Affordable
  - Capable of running multiple NN in parallel so should more than suffice in terms of processing power
- e-CAM50_CUNX camera
  - Has option to film @ 720p
- Breadboard and LEDs
  - Used for UI, indicate state of camera and its functions
- PD Pioneer 20000mAh
  - Power supply is 5V/3A, within the Nano Specification for its DC Power Jack

Handheld System Design

Image not to scale



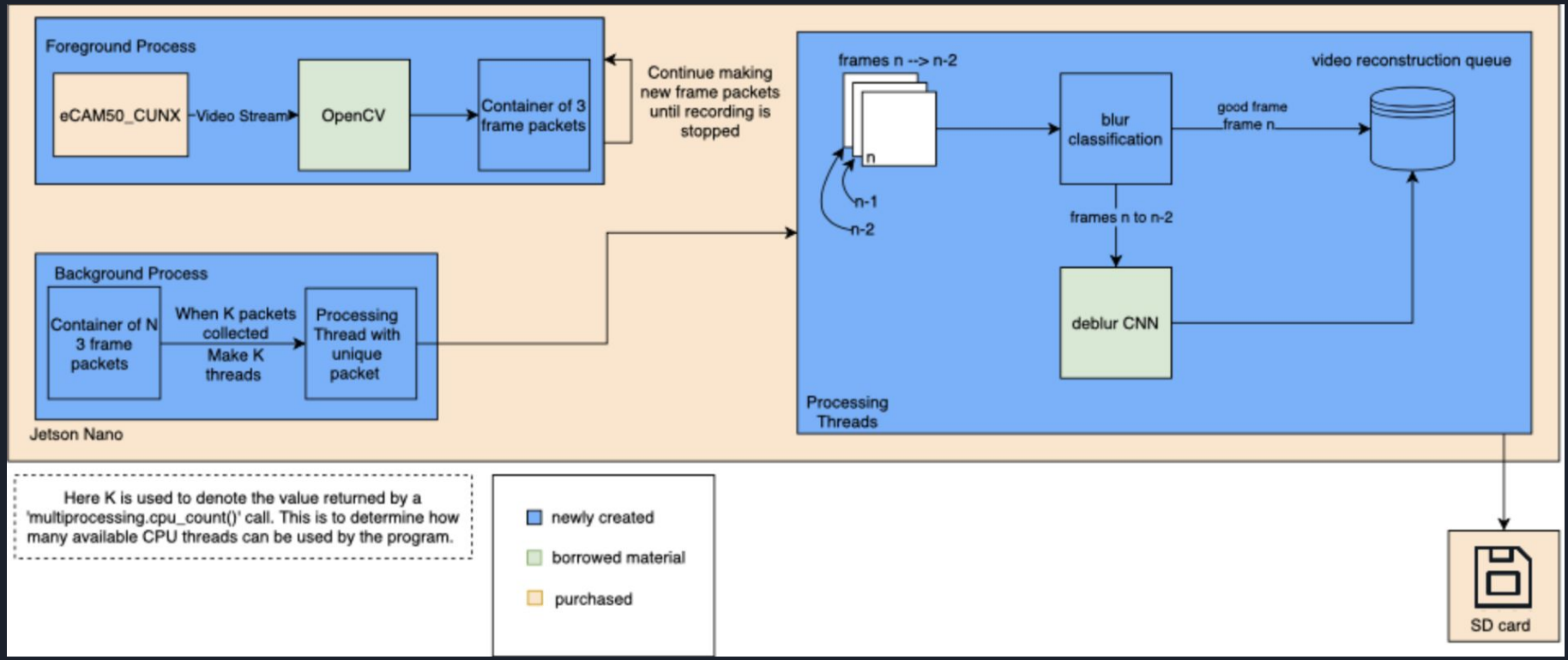
Implementation Approach (Software)

- Matlab
  - Used to create additional spatially invariant blurred images for our dataset
- Python (OpenCV and NumPy modules)
  - Video capture and backend
- BASH shell scripting
  - To start our processes on Jetson Nano boot

Implementation Approach (ML)

- CNN
  - Using tensorflow to create an encoder/decoder model
  - Trained using combination of prebuilt and newly fabricated images
- Deblurring Metric
  - Quickly classify whether the image is blurred via gradient magnitudes

System Diagram



Metrics and Validation

Metric	Validation Process
Video Storage Capacity (> 10 minutes)	Recording video and separately timing length of recording, when complete uploading video from SD card to check length of video
Multiple Video Storage (min 5 unique videos)	Recording 5 short videos, when complete uploading videos from SD card to check that we have 5 different videos present
Battery Life of System (> 15 minutes)	We will turn on system and record and process data for 2 sprints of ~5 minutes each with a 5 minute interval between them. Any time the system is not processing in the interval will result in the system idling until the next recording
Total frame processing time (< 3s)	Measuring the time from a single frame packet input to frame output with multiple different image scene

Metrics and Validation

Metric	Validation Process
Correctly classify type of blur given random image (90% accuracy)	Randomly select images from different scenes, then apply the corresponding blur kernel (motion, gaussian, haze, defocus, no blur) to use as our validation set
Deblur frames successfully 70% of the time (30% validation error)	Use the prior blur classification process to attempt to classify the image after going through the model.

Risk and Mitigation

Risk	Mitigation
Blur classification CNN incorrectly classifies a blurred image as sharp	Require that all images recorded be put through a 2D convolution with an averaging kernel, have their difference taken from the original image, and find the average difference in pixel value. If the value is above 0.035, force the image to be processed as blurred
User recording in hazardous environments (rain, snow, extreme temperature).	Print the camera frame with stronger material and add insulation around the external exposures (charging port and camera lens), so the components are no affected
CPU overload due to multiple threads	There is a Thread CPU count to ensure there are available threads. We also have a maximum threshold and maximum number of threads in order to mitigate the amount of threads being processed.

Project Management

Tasks	Sean	Nathan	Rebecca
Training Data Acquisition (created and downloaded)		✓	
Equipment, component, part Acquisition		✓	
CNN: development and training	✓	✓	
OpenCV/Shell Scripting/Backend development	✓		
Deploying Software to Jetson Nano			✓
Create a 3D model to act as camera’s frame		✓	✓
System Integration	✓	✓	✓
Developing deblurring metric and quantifying it	✓	✓	

Updated Schedule

New Tenative Schedule									Final Tweaks
Tasks	Week 6 (03/08 - 03/14)	Week 7 (03/15 - 03/21)	Week 8 (03/22 - 03/28)	Week 9 (03/29 - 04/04)	Week 10 (04/05 - 04/11)	Week 11 (04/12 - 04/18)	Week 12 (04/19 - 04/25)	Week 13 (04/26 - 05/02)	Week 14 (05/03 - 05/09)
Research possible altercations to proposed CNN	SP, NK - 3	SP, NK - 3							
Setup simple CNN that can take video as an input	SP, NK - 3	SP, NK - 3							
Create training data for blur classification	SP, NK - 2	SP, NK - 2							
Research/setup/test camera	SP, NK - 2	SP, NK - 2							
Research deploying models on Jetson Board	SP, NK - 3	SP, NK - 3							
Review different image alignment methods			SP, NK - 5	SP, NK - 5					
Develop Python code for Jetson backend	RJL - 3	RJL - 3	RJL - 3	RJL - 3					
Develop Matlab Code to create training data	SP, NK - 2	SP, NK - 2							
Develop Python OpenCV code for image handling			SP - 3	SP - 3					
Setup LED indicators on breadboard	RJL - 2	RJL - 2							
Integrate OpenCV code with Matlab code					SP, NK - 3	SP, NK - 3			
Integration between the Jetson Board and SD card			RJL - 4	RJL - 4					
Develope Image Alignment Matlab code					SP, NK - 4	SP, NK - 4			
Implement our CNN model across multiple threads					NK, RJL, SP - 4	NK, RJL, SP - 4			
Model camera frame in Solidworks			RJL - 3	RJL - 3					
3D print camera frame					RJL, NK - 1	RJL, NK - 1			
Integrating switches/button with GPIO pins	RJL - 2	RJL - 2							
Create testing metric for measuring blur							SP - 3	SP - 3	
Program conditionals for LED lights							NK-1	NK-1	
Test CNN model with verification data					SP, NK - 2	SP, NK - 2			
Extensive Full Scope Testing							RJL,NK,SP - 1	RJL,NK,SP - 1	
Final presentation							RJL,NK,SP - 1	RJL,NK,SP - 1	
Final Report							RJL,NK,SP - 1	RJL,NK,SP - 1	**Margin**

Team Members are assigned to task by initials
Difficulty Scale: 1 (easier) - 5 (hardest)