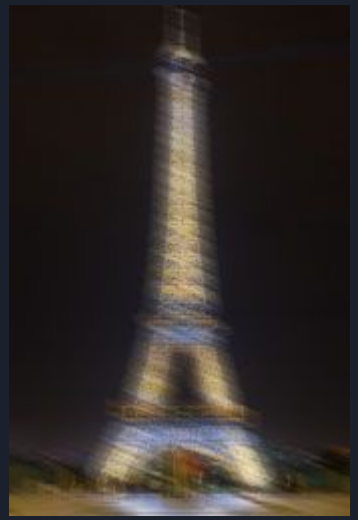
SharpCam

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

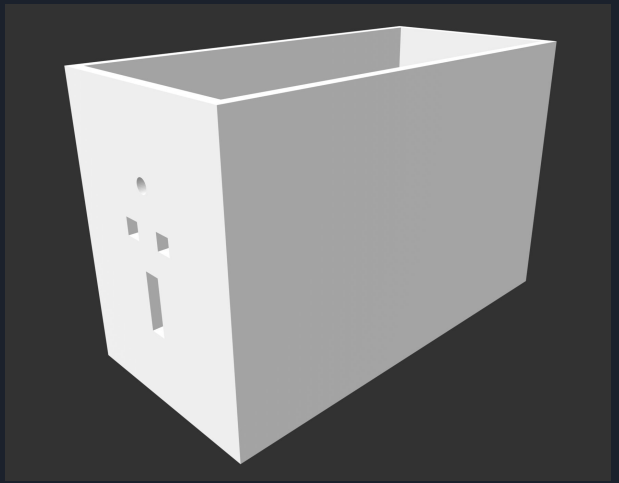
Introduction

- 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
- Raspberry Pi Module V2 Camera
  - Recording @ 1920x1080, 30fps
- 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
- Custom CNC'd encasement to contain all parts of the camera

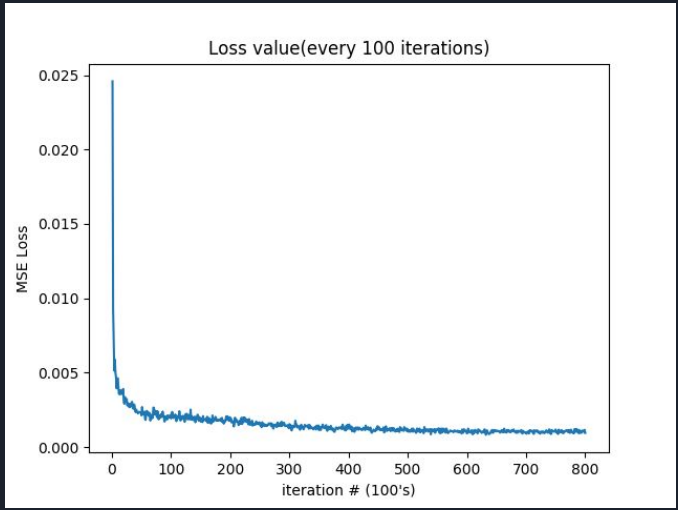


Implementation Approach (Software)

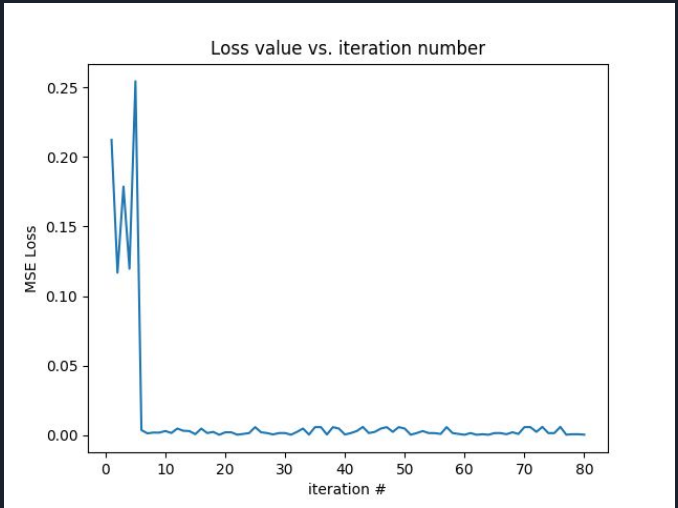
- Matlab
  - Used to help align images before and after they have gone through the DeblurNet
- Python (OpenCV and NumPy modules)
  - Video capture and backend
- BASH shell scripting
  - To start our processes on Jetson Nano boot and create environment variables for clean file management
- Torch Lua

Implementation Approach (ML)

- CNN
  - Using DeblurNet architecture which takes in a set of frames and deblurs them
- Deblurring Metric (Removed)
  - This was removed to try and enhance performance, this way our model can take in all forms of frames.

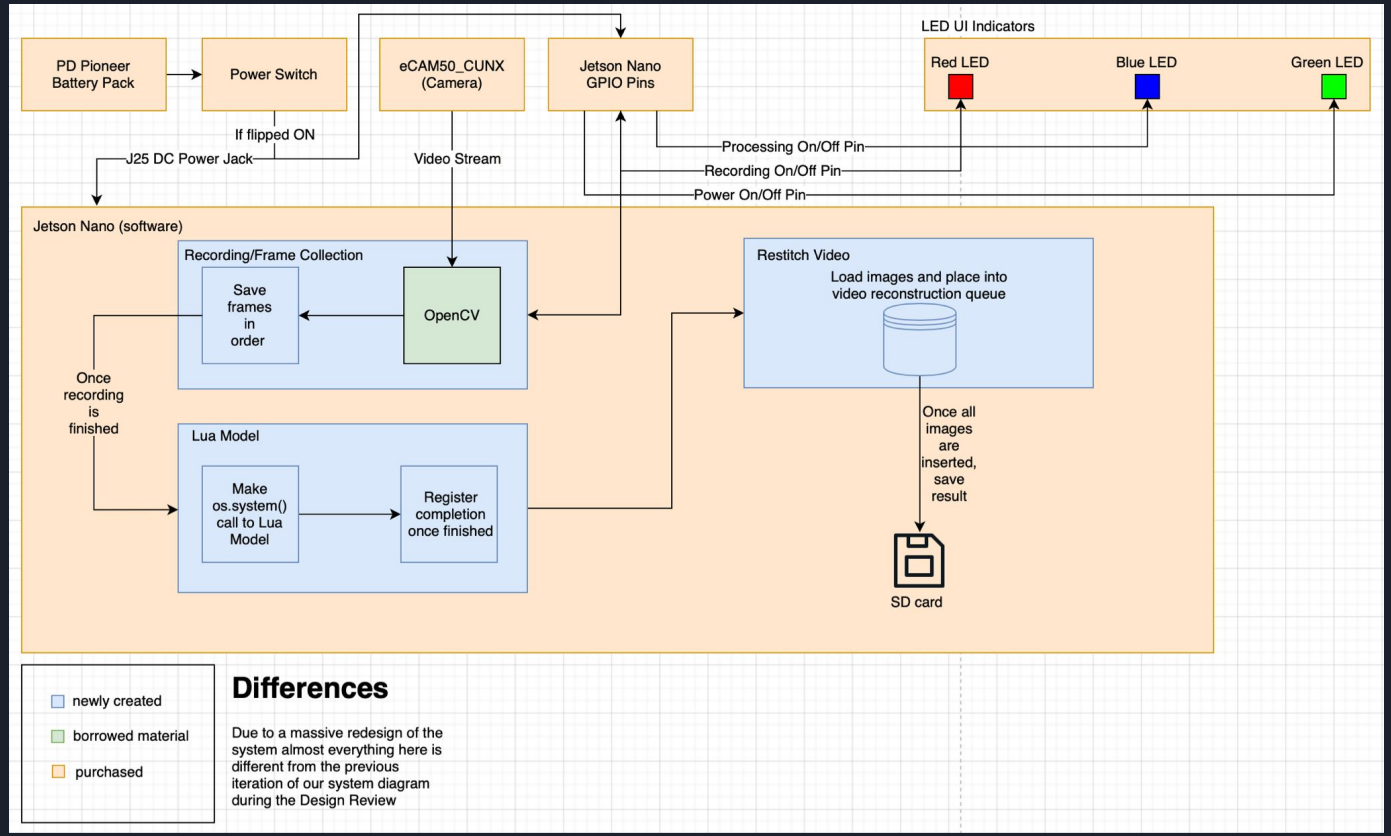


Pytorch Model Training loss after 80k iterations



Pytorch Model Validation loss every 2k iterations

System Diagram



Complete Solution

- The camera system will be contained inside the CNC'd case, powered externally through a wall outlet to ensure power stability
- At least 1 video will be taken, prepared, and processed by the CNN
- We will then playback the footage of the unprocessed video alongside the processed footage to compare and check results

Metrics and Validation

Metric	Validation Process	Performance	Pass?
Video Storage Capacity (0-10 minutes)	Recording video and separately timing length of recording, when complete uploading video from SD card to check length of video	When recording off of the Nano it can store any recording from 0 - 10 minutes easily, when on and connected to a power supply.	Y
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	The videos can be saved on the Nano itself, they now need to be saved explicitly on the SD for customer use afterwards.	Y
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	When the Nano is connected to a socket, the Nano will stay on for > 15 minutes. The power supply itself, shuts off after 30 seconds, which is an issue. We need to adjust the power supply.	In process
Time for System to run to completion (without counting CNN processing time) (~ 5 frames per 1 sec)	Keep track of number of frames recorded, comment out code interfacing with Lua model, and determine time for rest of code to run to completion	Using print outs we were able to get the amount of time it took (in seconds) and number of frames in the video, on average we had a ratio of ~ 5.75 frames per 1 second	Y

Metrics and Validation

Metric	Validation Process	Performance
Total video processing time (< 3 sec per frame)	Measuring the time for the video to be processed by the Lua model given a known number of frames	Still being tested
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.	Still being tested

Design Trade Offs

- Camera
  - We needed to switch the camera to save on time (reading through poor NVIDIA documentation)
  - The Raspberry PI camera integration was much smoother and quicker
- CNN
  - Using an older version of pyTorch on Lua
  - Hopeful improvement on model training
- Camera case
  - 3-D printed open case, with a removable top, for easy access to components

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		
Tasks	Week 14 (05/03 - 05/09)	Week 15 (05/10 - 05/14)
Setup LED indicators and integrate GPIO pins code	RJL - 2	
Integration between the Jetson Board and SD card	SP, RJL - 2	
Finetune the CNN model	NK - 3	
Setup Lua Torch on Nano	NK - 2	
Integrate Lua Script with python script	NK,SP - 2	
Configure projects items into the case	RJL - 2	
Test CNN model with verification data	NK, SP - 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