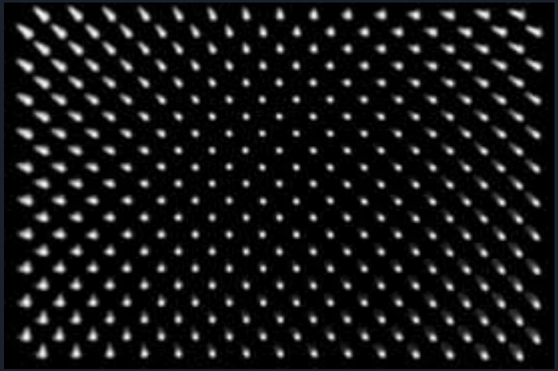
Team D1
Project Sharpcam

Nathan Koch, Rebecca-Jean-Louis, Sean Pogorelc

What is Project Sharpcam?

- Mobile video camera that can record multiple videos and deblur them in real time, specifically spatially invariant blur from camera shake
- Spatially invariant blur because it simplifies the solution space and still offers meaningful results.
- Offers a software solution to a problem typically solved with physical equipment



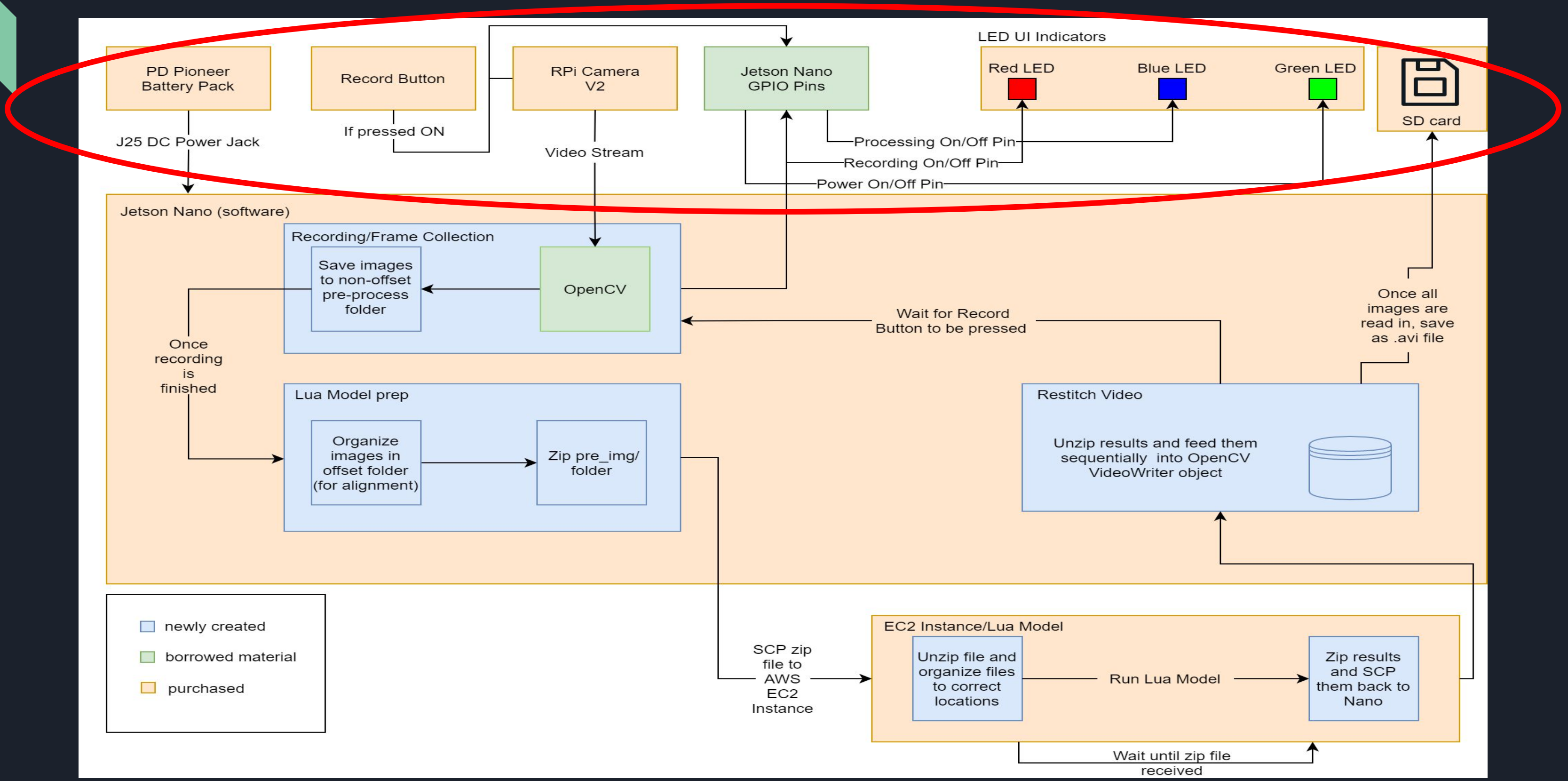
spatially variant blur

spatially invariant blur

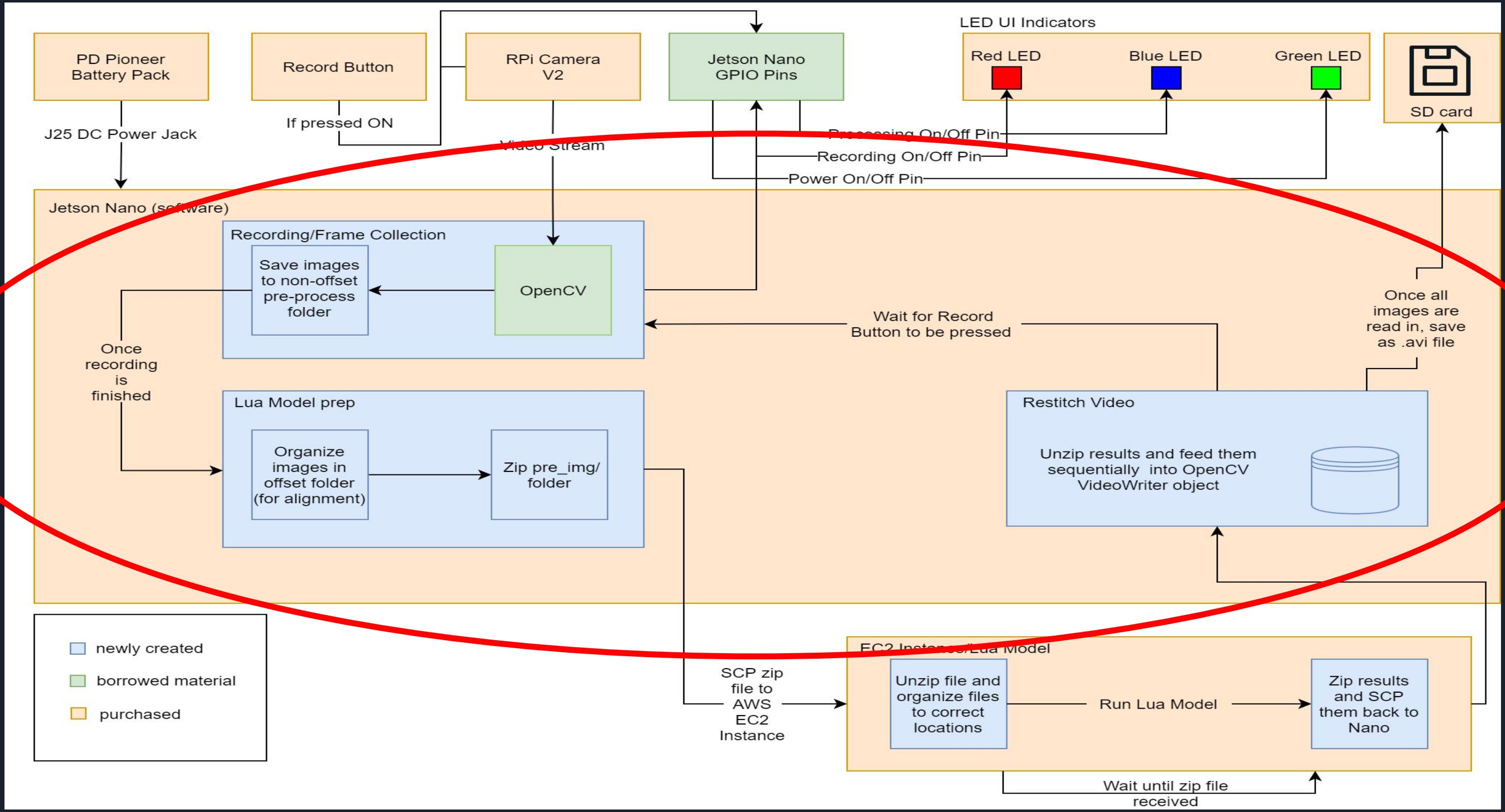
Application Area and Use Case

- Sharpcam can be used in any situation where it may be difficult to capture a video while maintaining a steady hand.
  - Some examples are: high action events such as sporting events, capturing suspicious activities and trying to track a specific object while moving
- Sharpcam can be used as a software approach for stabilizing video capture, eliminating the need for physical devices such as a gimbal

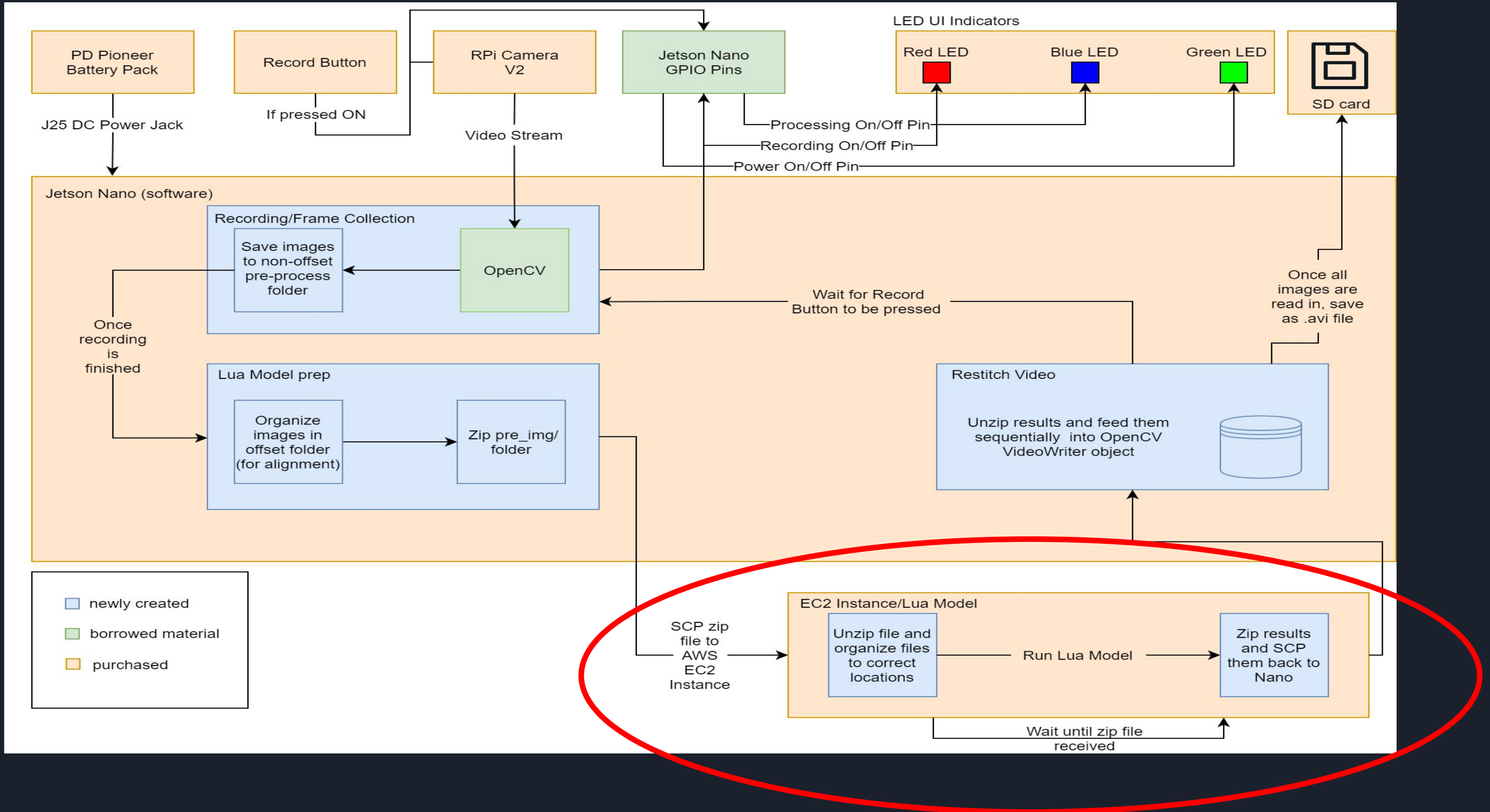
Hardware Components



Software Components (non-CNN)



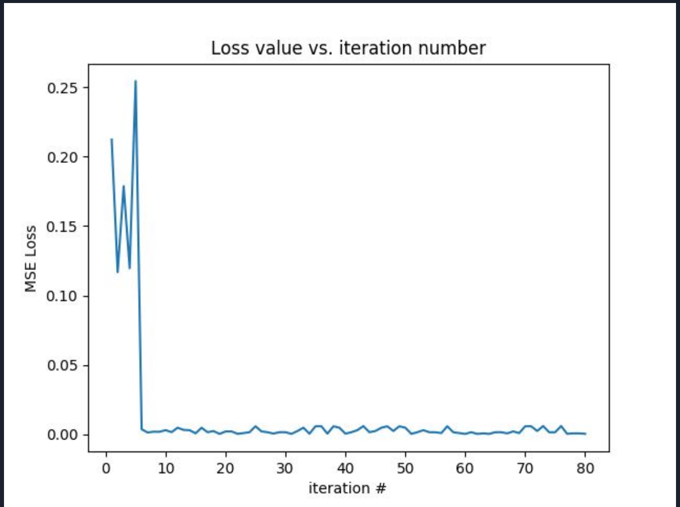
Software Components (CNN)

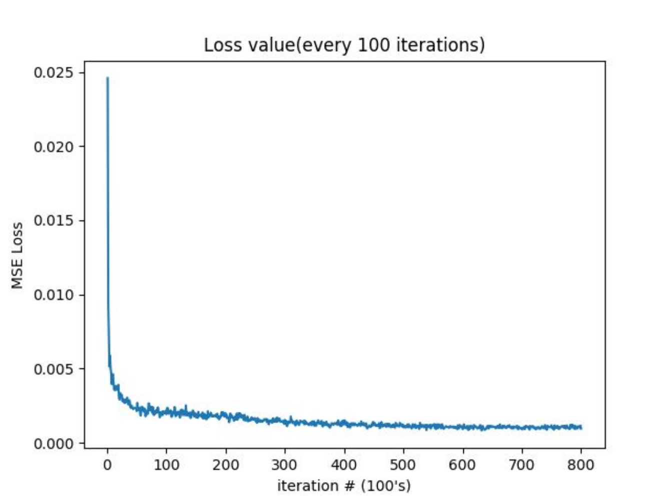


Challenges Overcome (CNN)

- Original CNN model architecture (DeblurNet) did not learn from original training data, so a switch to the provided model was needed

- DeblurNet architecture has 22 layers, which overloaded the Nano CPU, so processing was offloaded to aws ec2





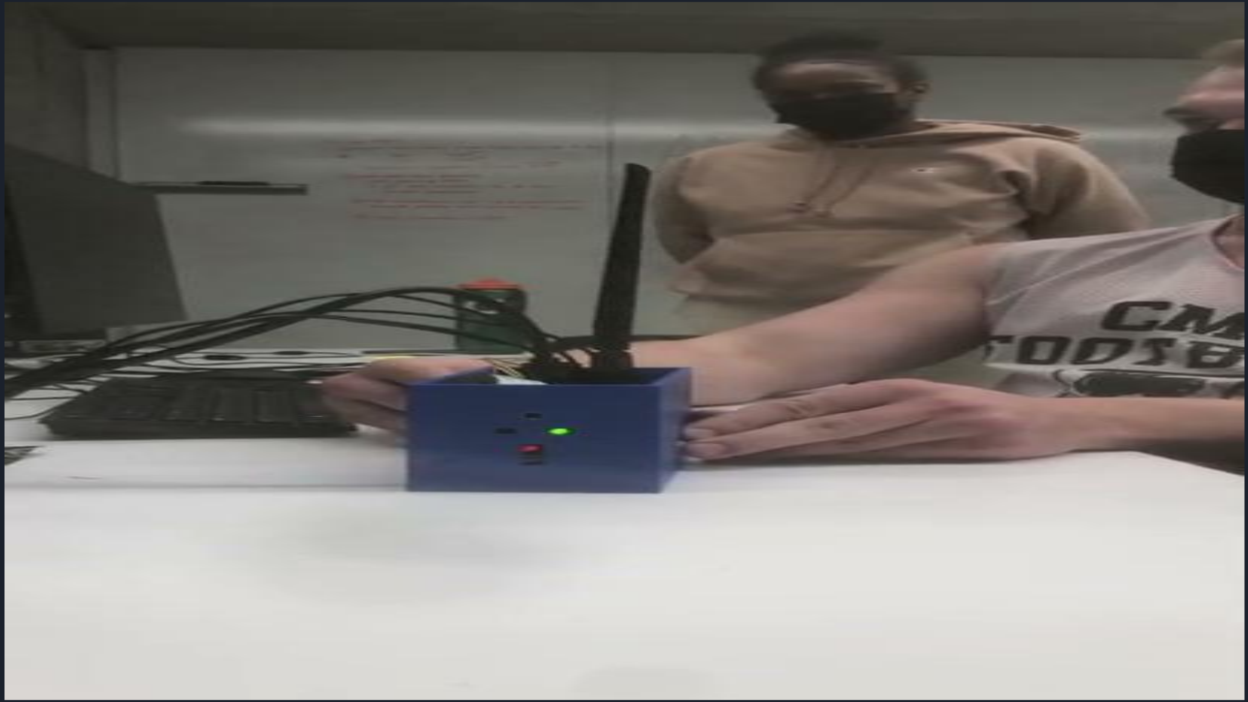
Challenges Overcome (Nano System Memory)

- Original design required use of shared memory
  - Very slow when trying to read/write to it
  - Made attaining 30fps impossible
  - Caused Nano to become unresponsive after short periods of recording
- Next iteration used a serial approach to the processing and stored frames in data
  - Would get process ‘killed’ after storing ~700 frames this way
    - Result of triggering the kernel’s “OOM” (out of memory) killer
- Kept the serial approach but now used write calls to store frames initially and os.system calls to move frames around
  - Threaded initial writes around fps regulating wait call in main process
    - Allowed for writes to not noticeably affect process runtime or fps
  - Used os.system to run “cp ...” commands to limit number of read/write calls made elsewhere

Challenges Overcome (Physical Hardware)

- Jetson Nano
  - There was an initial issue with our first Jetson Nano, where it wouldn't display nor turn back on
  - Our solution was to use our budget and order another one
- Battery
  - When the Nano is connected to the battery it will turn off after 35 seconds
  - Our solution was to remake the model stationary without the need of the power supply attached
- 3D Printed Case
  - It wasn't exactly what we envisioned especially because the length of the case was cut off too short, therefore not allowing us to have both the Nano and Battery pack held within
  - The LEDs and Camera cut out holes were pretty difficult to place the parts
  - Our solution was to just enclose the Jetson Nano, Camera, breadboard, LED’s, and buttons into the case

Fully Integrated Run Through



Side by Side Comparison

Pre Deblurring

Post Deblurring

What We Achieved

- Tactile buttons were able to control when to record and the system power, while LEDs were able to signal the power, recording, and processing stages of the script
- System capable of recording/storing multiple videos without needing to restart
- System capable of exchanging files with AWS EC2 instance
- System capable of recording at a consistent 30fps for a reasonable (>15 minutes) amount of time
- System capable of storing as many videos as SD card memory will allow (each with unique auto generated identifier)
- **Overall, created an all in one solution to recording a video and deblurring it in a reasonable amount of time (<2 seconds per frame)**

Lessons Learned/Takeaways

- Rebecca
  - Any created enclosing definitely needs precise measurements and physical mock confirmation before being made, otherwise the product will look wrong
  - Sometimes functionality takes precedence over aesthetics
- Sean
  - Just because a solution seems elegant doesn't mean it's also computationally efficient
  - Careful organization early on makes later additions faster and easier to implement/integrate
- Nate
  - Although it may be long and monotonous a deep dive into component specs can make a massive difference in the long run.
  - Sometimes the best way to learn is to fail, and in our case it would have been to fail faster and to fail a little harder, this way we could hash out some of our problems before it was too late.

Thank You CMU ECE Faculty!