Falcon: the Pro Gym Assistant

Team Ao

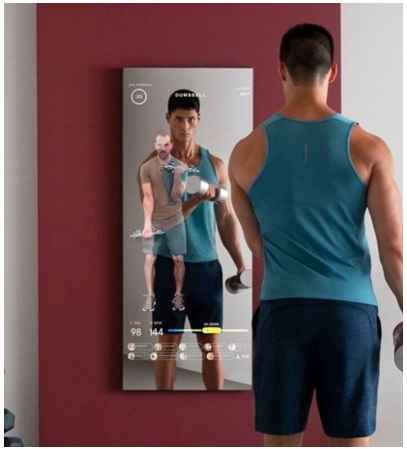
Vishal Baskar

Albert Chen

Venkata Vivek Thallam

Use Case

- Advanced at-home workout system that provides:
  - Demonstration of exercises
  - Rep counter
  - Calorie estimator
  - Live stream of themselves
  - Ability to get customized workouts that comprise of the following:
    - Leg Raises, Pushups, Lunges
  - Real-time feedback regarding posture **(Unique to Falcon)**
- Workout system involves a display and a side camera
- Processing done on an FPGA to address privacy concerns
- Areas Covered:
  - Software Systems, Signals and Systems, Hardware Systems



Mirror: A popular at-home workout tool

Materials/Components

- Materials:
  - Dark Suit + 3M Colored Bandages
  - Xilinx Kintex-7 FPGA KC705
  - Logitech C270 WebCam
- Components:
  - UART Interface
  - Joint Tracking Algorithm
  - Vivado HLS
  - Posture Analysis Algorithm
  - PyGame, OpenCV, SQLite

Solution Approach

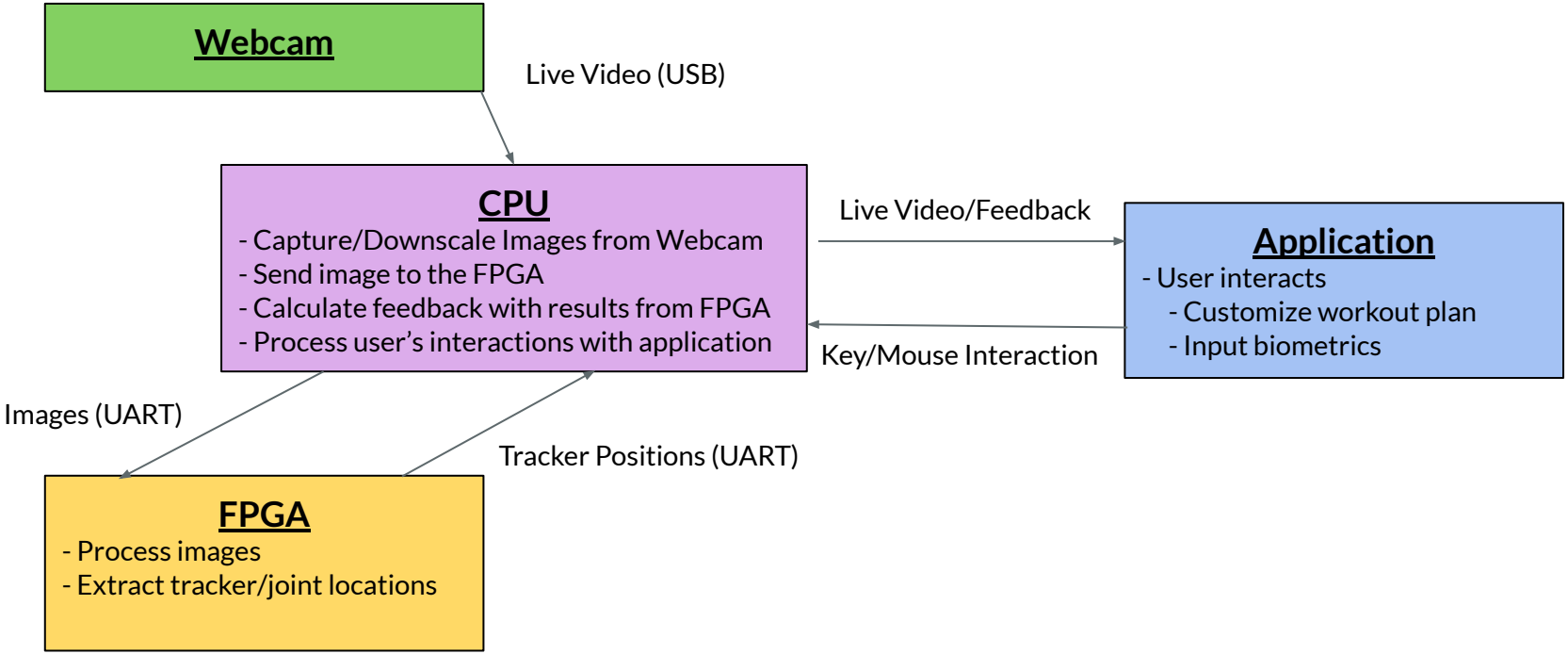


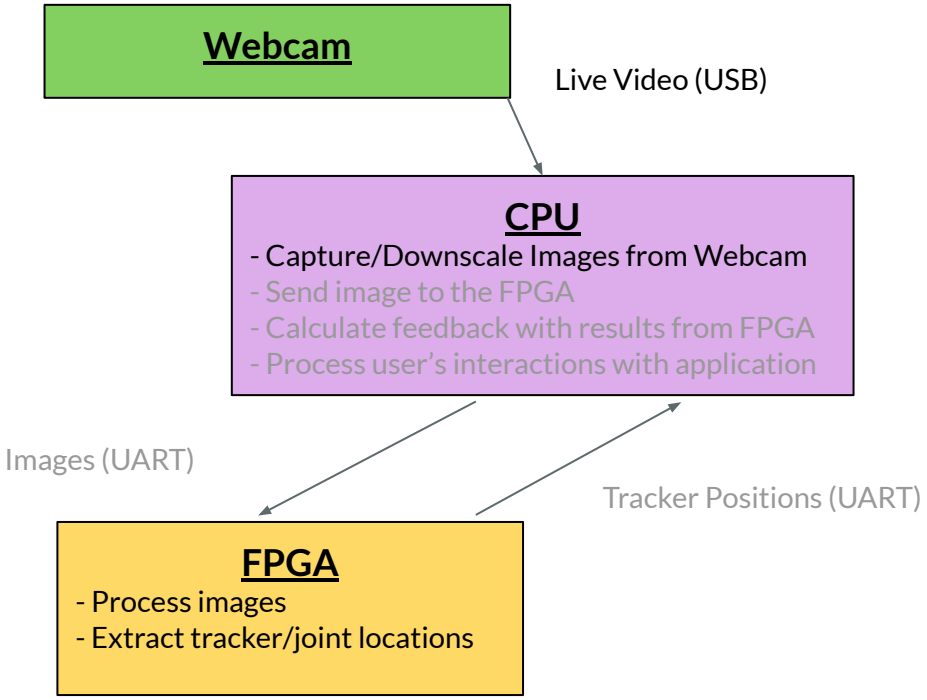
Image Processing

Pre-Processing:

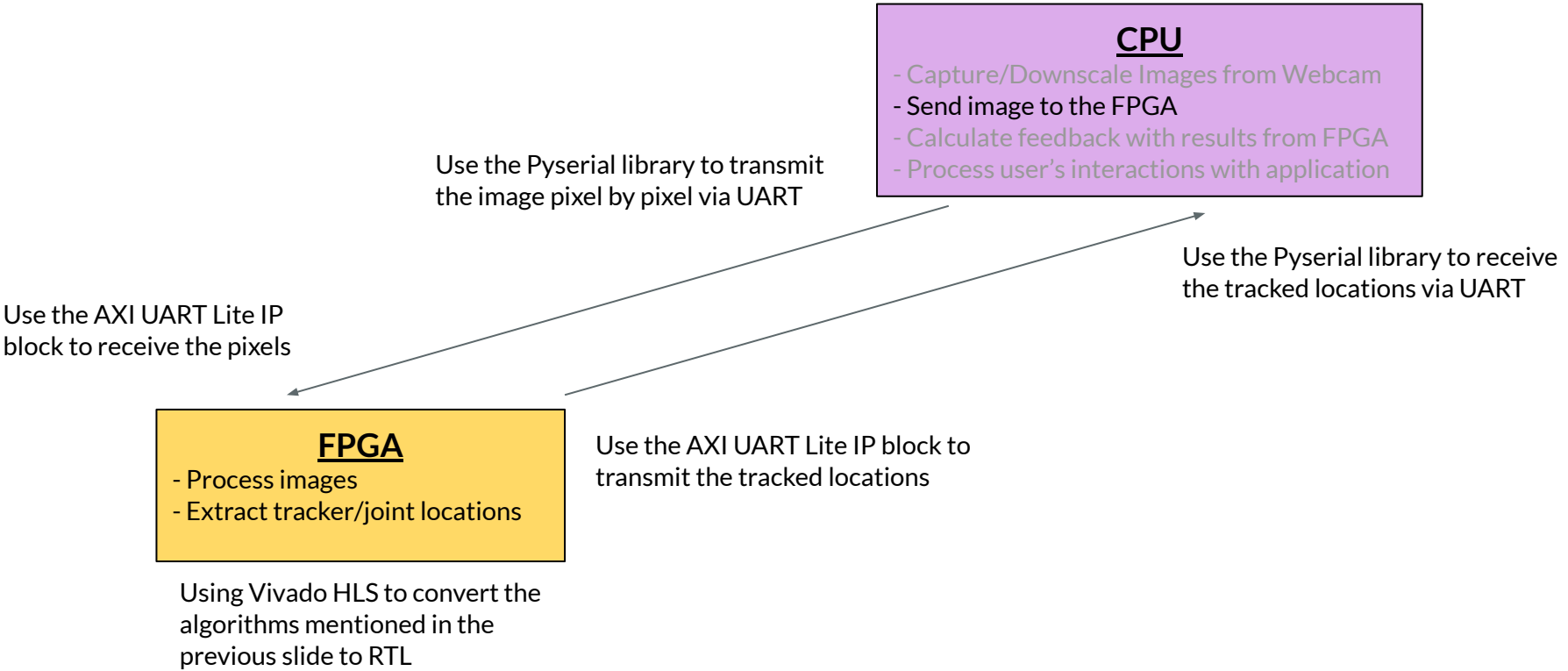
- Downscale + Convert (HSV)

Joint Tracking Algorithm:

- Binary Mask of pixels within HSV bounds
- Morphological Transform
  - Erosion
  - Dilation
- Max Area Rectangle
  - Max Area Histogram
- Output Center of Rectangle



Hardware


CPU
- Capture/Downscale Images from Webcam
- Send image to the FPGA
- Calculate feedback with results from FPGA
- Process user's interactions with application
Use the Pyserial library to transmit the image pixel by pixel via UART
Use the Pyserial library to receive the tracked locations via UART
Use the AXI UART Lite IP block to receive the pixels
FPGA
- Process images
- Extract tracker/joint locations
Use the AXI UART Lite IP block to transmit the tracked locations
Using Vivado HLS to convert the algorithms mentioned in the previous slide to RTL

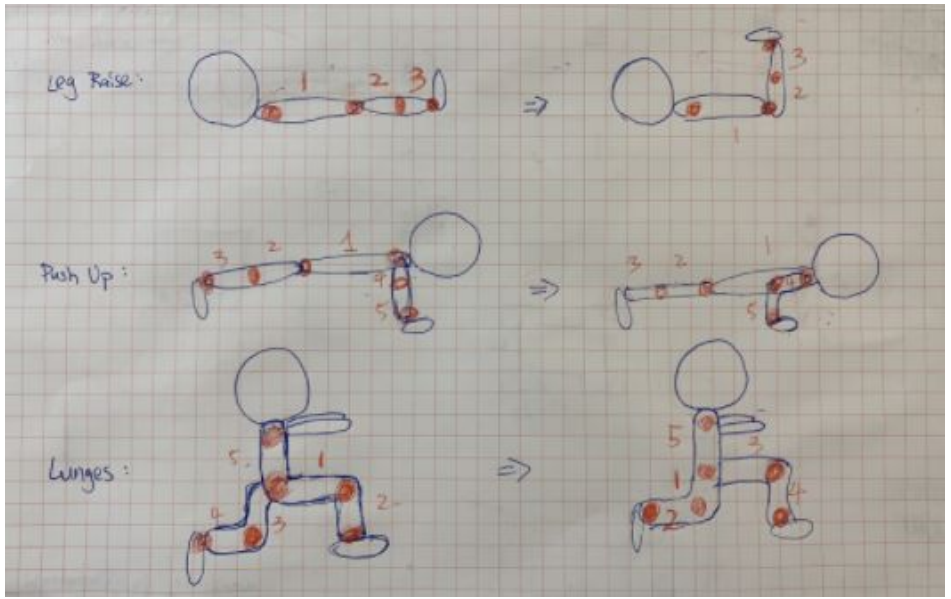
Posture Analysis

Joint Positions → **CPU**
- Capture/Downscale Images from Webcam
- Send Image to FPGA
- Calculate feedback with results from FPGA
- Process user's interactions with application

→ Feedback

Leg Raise:

- **Lines:** Shoulder - Hip - Knee - Ankle
- **Angles:** Hip & Knee
- **Checks:**
  - Angle of Knee = ~180 degrees
  - Angle at Hip = ~90 degrees
- **Feedback:**
  - "Knees are bent"
  - "Raise your Legs Higher"



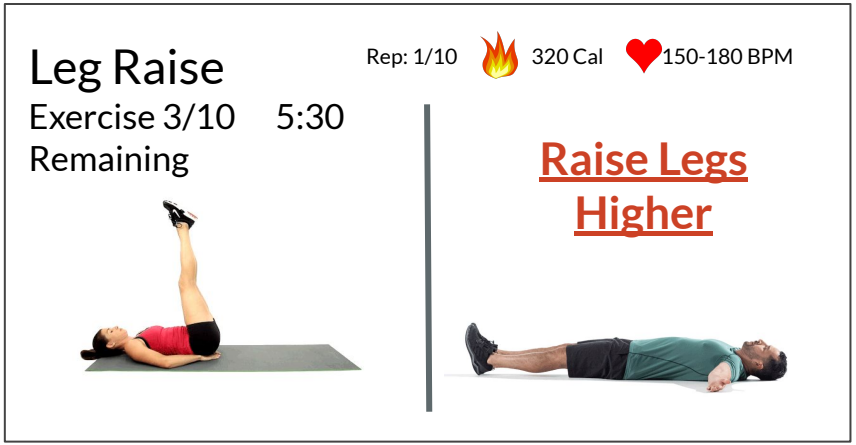
Application/UI

- Application hosted in pygame using the GUI library
- Image Capture and streaming through OpenCV
  - Save images periodically for Pyserial Library to stream
- Mouse + Keyboard Interaction
- Workout data stored in SQLite

Feedback from CPU →

Application
- User interacts
  - Customize workout plan
  - Input biometrics

→



Testing, Verification and Metrics

Requirement	Testing Strategy	Metrics
Downscaling & Conversion	Software testbench (analyze size and quality of resulting image)	100% size match
Detect trackers	Software testbench (analyze trackers over various images)	1 rep to be misclassified every set
Communication between computer and FPGA	Hardware testbench (analyze various packets of data sent)	Latency < 1s & 100% data accuracy
Posture Analysis	Software testbench (analyze various positions to extract info)	100% accuracy according to our designed models
UI (Workout Data + Feedback)	Human Eye (analyze the metrics are met from what is done)	Workout Data: ~tracker accuracy Feedback Delay < 1.5 secs

Risk Factors + Mitigation

- Unable to track the joints due to downscaled image -> more pixels to be sent -> increase I/O.
  - Plan to remove portions of the image based on workout
  - More defined trackers
- Unable to synthesize the C code to a high enough frequency -> results in lower baud rate -> influences overall latency.
  - Optimize as much as possible
  - Modify exercises to those that can handle slower feedback
- Having live stream, model video, and feedback at same time may be choppy
  - Separate live stream and feedback/workout app into separate applications

Work Distribution

- Signal Processing
  - Pre-Processing to downscale and convert the image (Python) -> **Albert**
  - Learn and implement algorithm to extract trackers and determine joints (Python) -> **Albert**
  - Implement posture analysis from the joint positions for feedback (Python) -> **Albert/Venkata**
  - Finetune thresholds to provide feedback regarding posture given joint locations -> **Albert/Venkata**
- Hardware
  - Learn how to communicate efficiently between the FPGA and CPU and vice-versa via UART -> **Venkata**
  - Learn and convert software algorithm to optimized synthesizable FPGA code (Vivado HLS) -> **Venkata/Albert**
- Software
  - Design the computer application interface (PyGame) -> **Vishal**
  - Integrate live video feed and capture image -> **Vishal**
  - Integrate recorded exercises and create timed workout with feedback -> **Vishal**

Updated Schedule