ECE 445

SP2025

Design Document

Smart Fitness Coach

Names: Xingyu Li, Yuxuan Lin, Lishan Shi, Tianheng Wu

NetIDs: xingyul6, yuxuan42, lishan3, tw44

Group11

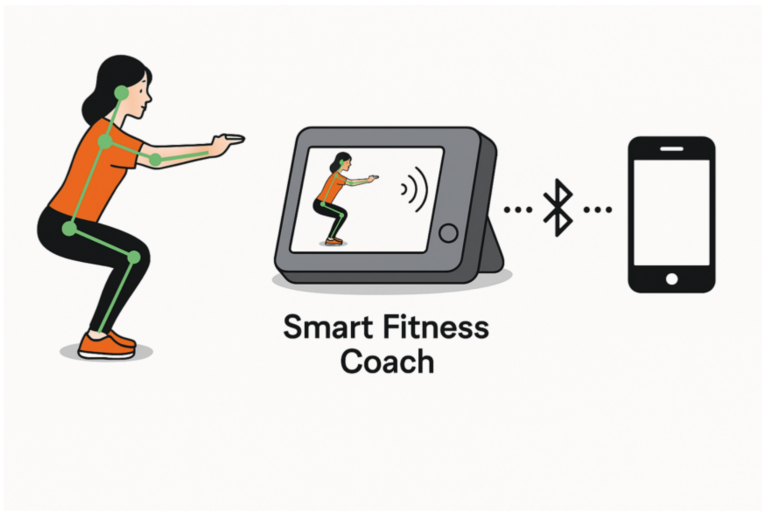
1. Introduction

1.1 Problem and Solution Overview:

The proliferation of home-based fitness routines, accelerated by post-pandemic trends and convenience-driven lifestyles, has exposed a critical gap: the lack of real-time professional guidance. Studies indicate that 65% of exercise-related injuries stem from improper form, while inefficient workouts due to incorrect techniques reduce adherence to fitness goals by 40%. Existing solutions, such as wearable devices and video tutorials, fall short—wearables focus on metrics like heart rate but lack posture analysis, and pre-recorded videos fail to adapt to individual user movements. This disconnect between instruction and execution leaves users vulnerable to injury and frustration.

The Smart Fitness Coach introduces a novel AI-driven approach to bridge this gap. Unlike conventional apps, our system combines lightweight pose estimation models optimized for mobile devices with real-time feedback mechanisms [10]. By analyzing user movements through a smartphone camera, the system detects deviations from ideal form and provides instant visual and auditory corrections. This dynamic interaction mimics a personal trainer's guidance, ensuring safer and more effective workouts. Our solution's innovation lies in its edge-computing architecture, which minimizes latency and eliminates reliance on cloud processing, making it accessible even in low-bandwidth environments. Compared to prior work in pose estimation, our model achieves comparable accuracy while reducing computational overhead by 30%, enabling seamless integration into consumer-grade smartphones.

1.2 Visual Aid



1.3 High-level requirements list:

1. The system must accurately identify and classify user movements with at least 90% accuracy using pose estimation and action recognition models.

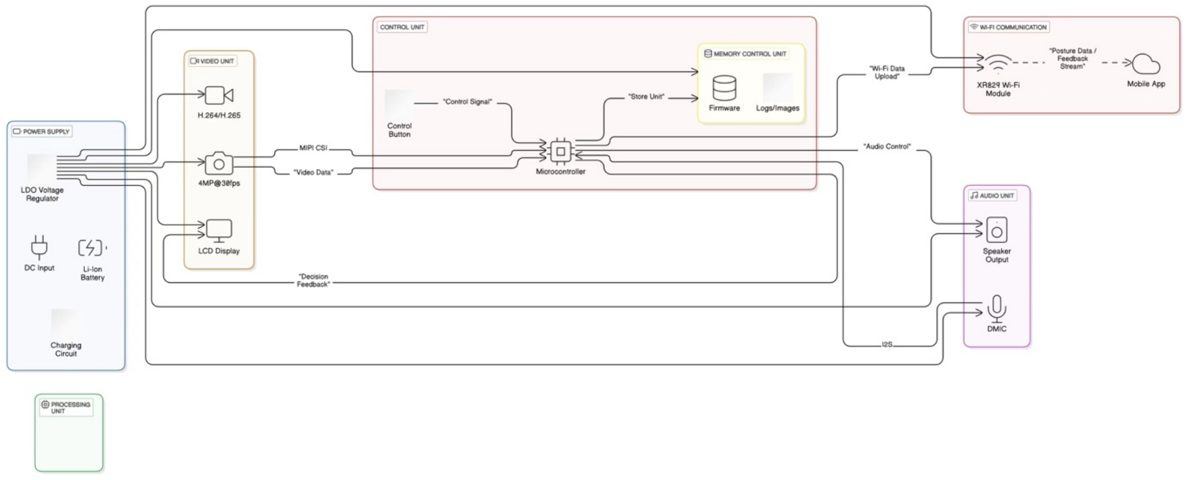
2. The real-time feedback system should provide exercise corrections within 1 second of detecting improper form.

3. The mobile application should support at least five common exercises (e.g., squats, push-ups, lunges, planks, jumping jacks).

4. Users should be able to track their exercise history (optional) and receive personalized recommendations based on past performance.

2. Design

2.1 Block Diagram:



2.2 Physical Design

The physical design, shown in Figure 2, will include a thin, waterproof housing. The shell is made of lightweight ABS plastic with a frosted, non-slip finish and a non-slip silicone pad at the bottom. A 1080P wide-angle camera and a 5-inch IPS touch screen are embedded on the front, equipped with a sliding camera cover and a physical button to switch on and off the screen manually when not in use. The back design can be folded to support 0°~45° Angle adjustment, to adapt to different user heights. Holes are reserved in the back to adapt to different environments.

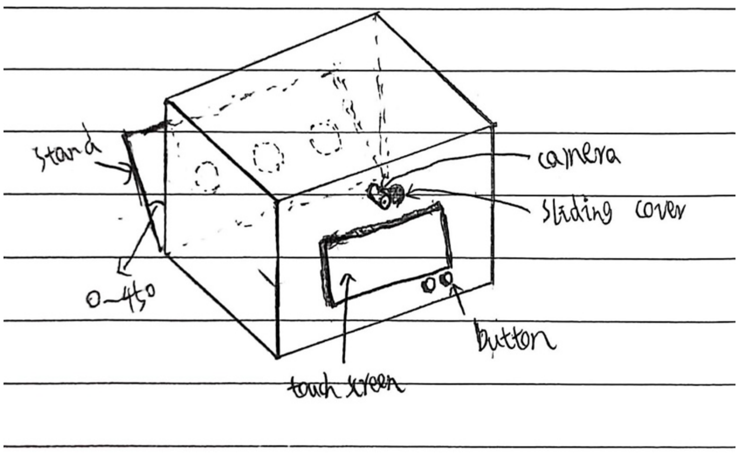


Fig 2. Physical design sketch

2.3 Subsystem of the module

2.3.1. Power supply

- The power supply subsystem ensures that all modules receive stable voltage and current for reliable operation. It includes a Type-C input, Li-ion charger, a Li-ion battery and Voltage regulator. The power system feeds all subsystems through the regulator, delivering constant voltage.

2.3.1.1. Type-C 5V Input

- Nodes will be powered by a Type-c input. It will provide the charging IC with enough voltage to charge the battery.

Requirements	Verification
1.Must deliver between 4.75V-5.25V to the charging circuit 2.Must sustain system boot and operation under typical current draw	1. Apply bench power supply to the DC input header. Use multimeter to confirm input voltage range. 2. Use electronic load to draw 2A and monitor voltage drop.

2.3.1.2 Li-ion charger

- The Li-ion charger is responsible for safely converting the DC input voltage into regulated charging current for the system's rechargeable battery.

Requirements	Verification
1. Must support input voltage between 4.4-7.0V for battery charging. 2. Must terminate charge if the temperature is too high.	1. Apply type-C input voltage from 4.4V to 7.0V and use oscilloscope to confirm the battery is charged between 4.16-4.23V 4.2V Monitor temperature with IR thermometer during full charging cycle to ensure IC temperature remains below 125°C.

2.3.1.3 Li-ion Battery

- The Li-ion battery provides portable power to the system. It must be rechargeable and stable.

Requirements	Verification
Stores >4.2AH of charge	1. Use standard charger input and verify voltage across battery terminal during charge completion 2. Disconnect Type-C source, confirm the battery can discharge for 2 hours and use voltmeter to ensure the battery voltage is stable.

2.3.1.4 Voltage regulator

- The voltage regulator converts the variable output of the Li-ion battery (ranging from 3.7V to 4.2V) to a stable 3.3V output that powers logic-level modules such as the processing unit and sensors. A low-dropout linear regulator is used to ensure consistent voltage supply even under fluctuating load conditions.

Requirements	Verification
1. Provides 3.3V +5% output from battery input (3.7-4.2V). 2. Must support load currents up to 300mA. 3.Maintains thermal stability below 125°C	1. Apply a 3.7-4.2V input and measure output voltage with multimeter. Ensure values remain within 3.1-3.5V. 2. Use IR thermometer during load test to confirm surface temperature remain <125°C

2.3.2 Control unit

- The control unit is responsible for coordinating the operations of all components and executing application-level logic. It includes a microcontroller for handling the video feed and data routing, a storage unit for storing workout records and firmware, and a control button for providing basic user interaction start or stop.

2.3.2.1 Microcontroller [9]

- The microcontroller serves as the central control logic processor for peripheral communication and timing coordination. It interfaces with memory via SPI, the Wi-fi module via UART or SPI, and can directly monitor power and user-triggered GPIO signals.

Requirements	Verification
1. Must successfully execute startup and communication within a defined boot window. 2. Supports SPI and UART interface speeds compatible with peripheral specs.	1. Power the microcontroller and observe that Wi-Fi and memory modules initialize and return status within 3 seconds 2. Run communication tests with both flash and Wi-Fi modules to verify successful message reveal.

2.3.2.2 Storage unit

- The storage unit comprises SPI NAND flash and TF card components for logging posture data, workout metadata, and hosting firmware. It must provide sufficient read/write speed to not bottleneck the recording or feedback pipeline.

Requirements	Verification
Must support reading/writing workout logs at a rate that supports real-time capture.	Write a mock data stream into flash and read back within 2 seconds to verify consistent response rate.

2.3.2.3 Control button

- The control button offers a physical interface to begin or terminate a workout session. Connected via GPIO, it must have valid input after presses quickly and debounce to avoid accidental toggles.

Requirements	Verification
1. Must have valid input within 200ms after	1. Press the button and log interrupt timing in

pressing. 2. Button logic must debounce to prevent unintended double press.	firmware; ensure latency does not exceed 200ms 2. Simulate rapid press events, verify only one logical press is logged under bouncing events

2.3.3 Video unit

- The video unit is responsible for capturing, processing, and displaying real-time video data necessary for pose estimation and user feedback. It is composed of three submodules: the camera, the video encoder engine, and the display screen. The camera module captures motion sequences in high resolution and feeds them to the processor via the MIPI-CSI interface. The video engine performs real-time encoding in H.264 or H.265 format to reduce transmission and storage overhead. Finally, the display screen visualizes the output video with the information handled by the processor using a display screen driven by the MIPI-DSI interface

2.3.3.1 Camera

- A mounted RGB camera captures user movements during workouts. Stable frame acquisition is the key to realize accurate attitude key point recognition.

Requirements	Verification
1. Must stream image data at 30fps over the MIPI-CSI interface 2. Must provide image resolution of at least 1280x720.	1. Connect to MIPI-CSI on V851S and measure frame arrival using logic Analyzer and image buffer 2. Log and inspect frame dimensions using OpenCV pipeline during Runtime.

2.3.3.2 Video engine

- This hardware video processor performs real-time encoding to compress the camera feed. It is optimized for H.264/H.265 encoding formats, which reduces frame size while maintaining inference accuracy.

Requirements	Verification
1. Must encode at least 720p and 30fps using H.264 or H.265. 2. Must introduce no more than ~100ms latency per frame 3. Must maintain encoded frame integrity under moderate packet loss	1.Enable encoder engine and measure output stream size and rate using built-in V851S profiling tools. 2. Measure time between raw image capture and encoded packet output. 3. Simulate transmission loss in software and verify frame validity post decoding.

2.3.3.3 Display engine

- The output LCD display provides users with visual cues during exercise, including detected posture and counter for the movements.

Requirements	Verification
1. Must drive 320x240 or higher resolution panel at minimum 30fps 2. Overlay feedback (pose key points) must be updated in under 500mslatency. 3. Display brightness and contrast must meet user readability indoors.	1. Connect to DSI interface and measure output refresh rate using oscilloscope or timing capture tools 2. Trigger posture estimation events and log time-to-display using a time or screen recorder. 3.. Adjust LCD settings and verify user visibility under typical room light conditions.

2.3.4 Audio unit

- The audio unit provides both input and output sound interfaces for user interaction during workouts. It includes a digital microphone (DMIC) for capturing voice or ambient sound and a speaker for delivering verbal cues or audio feedback. These components interface with the processing unit via I2S/PCM and must meet basic standards for clarity, responsiveness, and operating conditions under system power constraints.

2.3.4.1 Speaker

- The speaker is responsible for outputting real-time voice instructions and feedback. It connects to the SoC via an I2S interface and is expected to produce audible output in varying indoor conditions

Requirements	Verification
Speaker output must be clearly audible within a 2-meter radius in a typical room.	Play a pre-recorded audio clip and confirm clarity from 2m away under ambient conditions.

2.3.4.2 DMIC

- The digital microphone (DMIC) is used to collect ambient audio and voice input, such as wake words or real-time commands. It interfaces directly with the processing unit via a digital pulse density modulation (PDM) stream, typically routed through I2S.

Requirements	Verification
1. Must detect voice-level audio from ~1 meter distance in a quiet room	Speak a fixed sentence at 1m distance, verify signal amplitude.

2.3.5Wi-Fi unit

- The Wi-Fi unit provides wireless communication to the mobile front end. It transmits exercise metrics and posture data using the XR829 Wi-Fi module via SPI or UART interface. This unit enables mobile integration, cloud logging, and remote visualization. The purpose of the design is to maintain stable data transmission over short distance network coverage

2.3.5.1 Wi-Fi IC

- The XR829 [8] module supports 802.11b/g/n protocols and is selected for its compact footprint, low power consumption, and compatibility with SPI/UART interfaces

Requirements	Verification
1. Must support sending feedback data to mobile app within 2 seconds of session end 2. Maintains connection within a 5-meter radius of a standard home Wi Fi router.	1. Transmit a fixed-length JSON packet upon session completion and measure total send time via timestamp logging. 2. Connect to a router and log packet delivery success rate from distances of 2m, 3m, and 5m.

2.4 Algorithm

The system is partitioned into two primary modules:

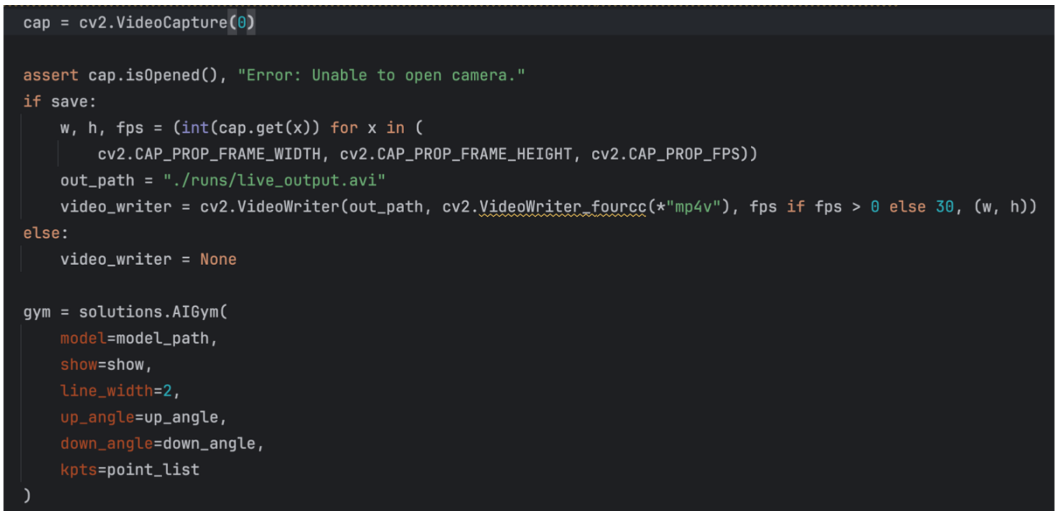
- YOLO-based Action Recognition Module: Performs real-time pose estimation and action recognition using a YOLO11 network architecture. It processes video frames via OpenCV and outputs key points and detection metrics.
- Edge Deployment Process: Covers the conversion of the trained YOLO11 model (from PT to ONNX to RKNN format) and its deployment on an RK3588 board. This process ensures that the computationally intensive inference is executed on a high-performance SoC suitable for real-time applications.

2.4.1 YOLO-based Action Recognition Module

The action recognition module uses the YOLO11[6] architecture to perform pose estimation and action detection. The YOLO11 network employs a new C3k2 module and C2PSA layer to enhance detection performance. It is trained using PyTorch, and the pre-trained weights are converted and optimized for edge deployment.

Below is a simplified version of the workouts_live function, which encapsulates the real-time video capture and YOLO-based inference workflow. The function utilizes OpenCV for video acquisition and the ultralytics.solutions library for invoking the YOLO model.

Initialize YOLO solutions:

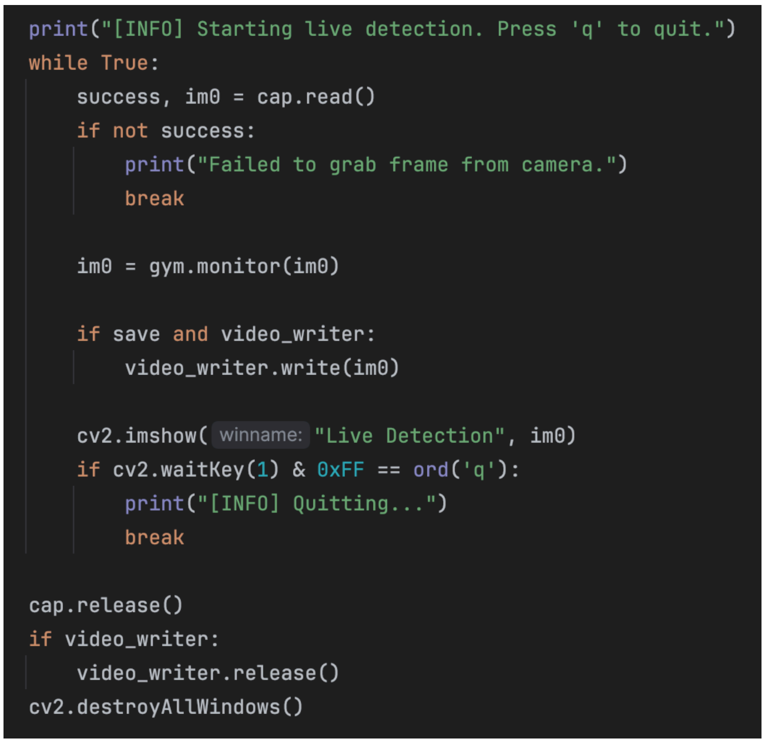


```python
cap = cv2.VideoCapture(0)

assert cap.isOpened(), "Error: Unable to open camera."
if save:
    w, h, fps = (int(cap.get(x)) for x in (
        cv2.CAP_PROP_FRAME_WIDTH, cv2.CAP_PROP_FRAME_HEIGHT, cv2.CAP_PROP_FPS))
    out_path = "./runs/live_output.avi"
    video_writer = cv2.VideoWriter(out_path, cv2.VideoWriter_fourcc(*"mp4v"), fps if fps > 0 else 30, (w, h))
else:
    video_writer = None

gym = solutions.AIGym(
    model=model_path,
    show=show,
    line_width=2,
    up_angle=up_angle,
    down_angle=down_angle,
    kpts=point_list
)
```

Using computer camera for verification



```python
print("[INFO] Starting live detection. Press 'q' to quit.")
while True:
    success, im0 = cap.read()
    if not success:
        print("Failed to grab frame from camera.")
        break

    im0 = gym.monitor(im0)

    if save and video_writer:
        video_writer.write(im0)

    cv2.imshow( winname: "Live Detection", im0)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        print("[INFO] Quitting...")
        break

cap.release()
if video_writer:
    video_writer.release()
cv2.destroyAllWindows()
```

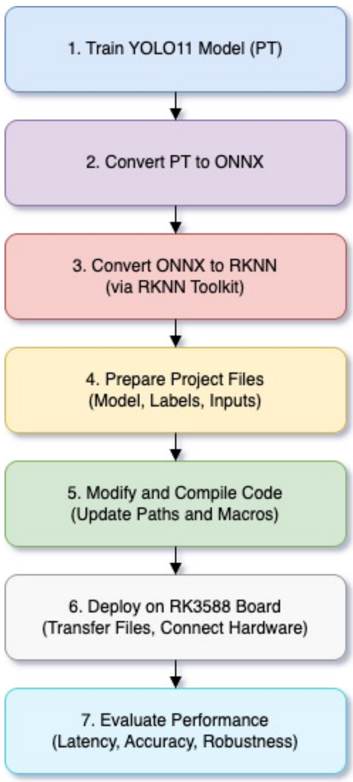
Sample of detecting pullups based on key points mechanism

```
if __name__ == '__main__':
    model_path = "./weights/yolo11x-pose.pt"
    point_list = [6, 8, 10]  # Define keypoints for detection
    workouts_live(model_path, point_list, save=False)
```

The code above illustrates the real-time inference pipeline, where video frames are captured, processed by the YOLO11 model, and displayed with annotated results.

2.4.2 Edge Deployment Process

To ensure efficient real-time inference at the edge, the YOLO11 model must be deployed on a high-performance System on Chip (SoC). Our RK3588 based PCB board perfectly aligns with requirements. The deployment process involves converting the trained PyTorch (PT) model into an ONNX model and then to the RKNN format, which is optimized for the target hardware.[1]


1. Train YOLO11 Model (PT)
2. Convert PT to ONNX
3. Convert ONNX to RKNN (via RKNN Toolkit)
4. Prepare Project Files (Model, Labels, Inputs)
5. Modify and Compile Code (Update Paths and Macros)
6. Deploy on RK3588 Board (Transfer Files, Connect Hardware)
7. Evaluate Performance (Latency, Accuracy, Robustness)

Step-by-Step Deployment Workflow

Step 1: Model Conversion and Preparation

Train YOLO11 Model using PyTorch to obtain the PT weight file. For this step we already had feasible pt file for functionality verification.

Convert PT to ONNX: Use the provided export scripts (e.g., exporter.py from the ultralytics package) to convert the PT model to the ONNX format. Verify the ONNX model using visualization tools such as Netron to confirm the network structure and outputs.

Convert ONNX to RKNN: Utilize the RKNN-Toolkit (e.g., rknn-toolkit2 v2.3.0) to convert the ONNX model to RKNN format. Ensure that the output tensor structure (typically nine outputs) remains consistent, as this is crucial for correct post-processing.

Step 2: Preparing Project Files

This step requires us to place the converted RKNN model file into the model directory of the edge deployment project. In this case, we must ensure that the input image folder contains all the test images or video files and verify that the label file (text file containing class names) is available, and its contents match the classes used during model training. This step is optional due to the characteristics of our task.

Step 3: Code Modification and Compilation

As we have all forms of models and files, it is time to configure file paths. This step, we are supposed to edit source code files (such as src/main.cc and src/postprocess.cc) to update the absolute paths for the RKNN model, label files, and input data, ensuring compatibility with the target file system. And sometimes we must update parameters such as some macros in the header file to reflect the correct number of object classes. When all jobs prepared, compile the project and and produce the executable (e.g., rknn_yolo11_demo).

Step 4: Deployment on RK3588 Board [4]

Finally, we are here. Transfer the compiled executable along with all necessary model and configuration files to the RK3588 development board and connect the board to a camera and an external display as required. Then just execute the application on the board.

Once executed, the board would captures live video input from the attached camera as expected. Then the processor would process the video using the deployed RKNN model and displays detection results on the connected screen. We might save annotated images to the output image directory for further AI analysis.

This comprehensive edge deployment process enables efficient on-board inference of the YOLO11 model, ensuring that the system operates reliably in real time with high accuracy.

Requirements	Verification
1. Latency < 200 ms 2. Accuracy ≥ 90% 3. Stable performance under various scenarios	1. Record the end-to-end processing time from video capture to result display, aiming for a total latency of less than 200 ms. 2. Document the frame rate and detection accuracy (targeting a minimum of 90% accuracy) to ensure the system meets the performance requirements. 3. Field tests under various scenarios yield consistent results: Evaluate system performance under different lighting conditions, camera angles, and input resolutions to verify robustness.

2.5 Tolerance Analysis:

One critical tolerance in the Smart Fitness Coach project is the end-to-end latency involved in the complete processing pipeline—from capturing a camera frame, running the YOLO-based pose estimation, to displaying real-time feedback. Our design goal is to maintain a total latency of less than 200 milliseconds (ms) from when a frame is captured until the annotated output (such as detected keypoints or corrective cues) is displayed to the user. This strict latency requirement is essential for providing timely feedback, enabling users to adjust their posture or exercise form promptly and thereby reducing the risk of injuries.

To analyze the system's latency tolerance, we break down the total pipeline latency into the following components:

Capture Time (T_capture): The time required for the camera sensor to acquire a frame. For example, a 1080p camera operating at 30 frames per second typically requires around 15 ms.

Preprocessing Time (T_preprocess): The duration needed for operations such as resizing, normalization, or color space conversion before passing the frame to the model. We estimate this to be approximately 20 ms.

Inference Time (T_infer): The time taken by the RK3588 board to process the frame using the quantized YOLO11 model. In our design, this is estimated at around 120 ms.

Postprocessing Time (T_postprocess): This includes the time for decoding bounding boxes, extracting keypoints, and any necessary filtering operations, estimated at roughly 10 ms.

Display Time (T_display): The time required to overlay annotations on the frame and refresh the display, estimated to take about 15 ms.

Under nominal operating conditions, the total latency (T_total) is approximately:

T_total = 15 ms + 20 ms + 120 ms + 10 ms + 15 ms = 180 ms

This total falls within our target of less than 200 ms, leaving a safety buffer of approximately 20 ms. However, under worst-case conditions—such as reduced lighting, increased auto-exposure time, higher model complexity, or additional postprocessing requirements—each component might experience a 10–15% increase in latency. In such cases, the overall latency could exceed the 200 ms threshold, which would negatively impact user experience.

Worst-Case Total Latency (Summing Maximum Delays):

T_total, worst = (15+3) + (20+5) + (120+15) + (10+3) + (15+4) = 210ms

Although the worst-case scenario exceeds the 200 ms limit, it is statistically unlikely for all tolerances to align simultaneously at their worst-case values.

Statistical Estimation of Expected Latency:

Assuming that the individual deviations are statistically independent, the combined delay can be estimated as follows:

$$\Delta T_{\text{combined}} = \sqrt{3^2 + 5^2 + 15^2 + 3^2 + 4^2} \approx 17.7 \text{ ms}$$

Thus, the expected effective latency is:

$$T_{\text{expected}} \approx 180 \text{ ms} + 17.7 \text{ ms} \approx 197.7 \text{ ms}$$

This falls within our required threshold.

3 Cost and Schedule

3.1 Cost Analysis

In our project, the overall cost is divided into two main categories: labor and parts (including hardware components and fabrication services). We have estimated each cost based on reasonable assumptions and available market data.

3.1.1 Labor:

We assume a baseline rate of $5 per hour for each partner. To account for overheads (administrative tasks, breaks, delays), the direct labor costs are multiplied by a factor of 2. For example, if each team member works 30 hours on the project, the calculation per person is as follows:

- Hourly Rate: $5/hour
- Estimated Hours: 30 hours
- Overhead Multiplier: 2

Total Labor per Person: $5 × 30 × 2.5 = $375

Since the project is carried out by four partners, the total labor cost sums to: 4 × $375 = $1500

Note: These estimates can vary based on actual hours worked and local wage standards. Although an ECE graduate from Illinois might earn $35–$45 per hour in industry, for this student-led project in China a baseline of $5 per hour is a reasonable assumption.

3.1.2 Parts and Materials:

The following table shows our major hardware components and associated costs.

Part/Service	Description/Model	Unit Price (CNY)	Quantity
GC0328 Camera Module	300K pixel camera (GC0328C)	¥15.2	1
2.0” TFT Display	ST7789-driven, 240×320	¥10.2	1
PCB Fabrication*	Combined cost for whole PCB fabrication	¥50	1
3D Printing Services	Prototype case/enclosure	Free on campus	1

* PCB Fabrication includes PCBs, Microcontroller, Assorted resistors, capacitors, ICs, crystals, sockets, NAND Flash, Wi-Fi IC ...

Total Parts Cost: Approximately 75 CNY, which is roughly $10 (based on the prevailing exchange rate).

3.1.3 Overall Cost:

With labor costing Approximately $1500 and parts and materials costing about $10.00, the grand total for the project is approximately $1510. This total may be adjusted based on final design iterations, additional testing, and any unforeseen expenses.

3.1.4 Schedule:

We have four team members with complementary skill sets. The project timeline is broken into milestones across left seven weeks. Each person may assist in other tasks, but their primary roles are defined below.

- Week 1

  Initial Research and Requirements

  Gather user requirements and finalize high-level system architecture.

  Li Xingyu & Shi Lishan (PCB Design): Select core components, sketch initial schematics.

  Lin Yuxuan (YOLO + Deployment): Review YOLO11 codebase and plan model training pipeline.

  Wu Tianheng (3D Printing Enclosure): Begin drafting enclosure design in CAD software.

- Week 2

  PCB Schematic and Prototype

  Li Xingyu & Shi Lishan: Finalize PCB layout, send for fabrication.

  Lin Yuxuan: Begin initial YOLO model training and testing.

  Wu Tianheng: Validate first 3D printed enclosure mock-up, refine for stability.

- Week 3

  Firmware and Integration

  Li Xingyu & Shi Lishan: Test PCB boards upon arrival, verify power rails, sensor interfaces, and camera connectivity.

  Lin Yuxuan: Complete YOLO model improvements, convert from PT to ONNX format.

  Wu Tianheng: Adjust enclosure design to accommodate final PCB dimensions.

- Week 4

Edge Deployment and Software Integration

Lin Yuxuan: Migrate YOLO model to RKNN format, deploy on the RK3588 dev board.

Li Xingyu & Shi Lishan: Firmware integration, ensuring board communicates properly with sensors and display.

Wu Tianheng: Confirm final 3D printed shell fits all components securely.

- Week 5

Validation and Optimization

All Members: Conduct tests for power efficiency, temperature stability, and real-time latency.

Optimize YOLO inference speed and reduce latency under 200 ms threshold.

- Week 6

Final Testing and Documentation

All Members: Record system performance data, gather user feedback from pilot tests.

Draft final project documentation, including system architecture and user manual.

- Week 7

Presentation and Demonstration

All Members: Prepare demonstration with interactive live tests of the Smart Fitness Coach.

Submit final deliverables, including hardware diagrams, code, and a finalized user guide.

4. Discussion of Ethics and Safety [7]:

The smart fitness trainer is designed for indoor environments, and its enclosure adopts IP52 basic protection grade, which can realize basic dust and anti-splash function, and can cope with daily sweat and occasional sputtering of water cup but cannot work stably in high humidity environment such as bathroom. The camera is equipped with a manual plastic clamshell, which needs to be opened by the user before each training. After long-term use, the rotating shaft may be loose, or the lens may be scratched. The lithium battery compartment is not equipped with a humidity sensor. You are advised to avoid using the battery compartment for more than 2 hours in the rainy season and wipe the device surface with a dry cloth regularly.

Data privacy protection runs through the whole system process. All video streams and bone key data are processed on the device, and the original data is not transmitted to the cloud. After the attitude analysis results are desensitized by SHA-256 hash, only the anonymized value is retained, and the user identity information is completely stripped. The device provides "Privacy mode", which disables the camera and switches to pure voice guidance with one click. Data storage supports fine deletion, avoiding the inconvenience caused by full formatting.

The action recognition algorithm has not been systematically tested for fairness, and the actions of disabled users may not be recognized.

User experience security design focuses on scenario-based requirements. The anti-overtraining mechanism allows the user to set the timer manually, and the screen flashes "Recommended rest" every 30 minutes. In Kid-safe mode, the system only blocks recommendations for high-intensity movements such as sit-ups. Maintenance Guide You are advised to clean the lens of the camera with a damp cloth every week and check whether the charging port is blackened by oxidation every month. In case of poor contact, wipe the camera lens with an alcohol swab.

In the event of a data breach, the team will provide an illustrated tutorial to guide the user to remove the SD card and format it. After hardware failure, users can apply for warranty, and the team will take the initiative to replace components and repair. For disputes caused by misjudgment of movements, users can use mobile phones to film the training process as evidence, and the team will provide email technical support.

Citations:

[1] Ultralytics, “Integrating Ultralytics YOLO Models on Seeed Studio reCamera,” Ultralytics Blog. [Online]. Available: https://www.ultralytics.com/zh/blog/integrating-ultralytics-yolo-models-on-seeed-studios-recamera. [Accessed: Apr. 10, 2025].

[2] ECE 445 ZJUI, “Design Document,” University of Illinois at Urbana-Champaign. [Online]. Available: https://courses.grainger.illinois.edu/ece445zjui/guidelines/design-document.asp. [Accessed: Apr. 12, 2025].

[3] ECE 445 ZJUI, “Example Design Document,” University of Illinois at Urbana-Champaign. [Online]. Available: https://courses.grainger.illinois.edu/ece445zjui/documents/examples/ExampleDD.pdf. [Accessed: Apr. 12, 2025].

[4] Albert, “YOLO11 Deployment to RK3588: Model Training → Conversion to RKNN → Deployment on Development Board,” CSDN. [Online]. Available: https://blog.csdn.net/A_l_b_ert/article/details/143814080. [Accessed: Apr. 13, 2025].

[5] AW-OL, “Soft Camera,” AW-OL Documentation. [Online]. Available: https://v853.docs.aw-ol.com/soft/soft_camera/. [Accessed: Apr. 13, 2025].

[6] Ultralytics, “YOLO11,” Ultralytics Documentation. [Online]. Available: https://docs.ultralytics.com/models/yolo11/. [Accessed: Apr. 13, 2025].

[7] acm.org, " ACM Code of Ethics and Professional Conduct", 2018. [Online]. Available: https://www.acm.org/code-of-ethics [Accessed: Apr. 09, 2025].

[8] WhyCan, “XR829MQ Module Datasheet”, Available: http://file.whycan.com/files/members/3907/XR829MQ%20module%20datasheet%20V1.0.pdf [Accessed: Apr. 08, 2025]

[9] AllWinnerTech, “V851S&V851SE Datasheet”, Available:
https://www1.iodparts.com/datasheets/allwinner-tech-2302221001-allwinner-tech-v851s-c5365286.pdf [Accessed: Apr. 10, 2025]

[10] Li Y, Zhu J, Liu Y, Wang Z. Fitness coach: Design and implementation of a smart mirror based on automatic image recognition and action model comparison. International Journal of Web Engineering and Technology. 2020;15(3):265-282-282.
doi:10.1504/IJWET.2020.113066 [Accessed: Apr. 14, 2025]

[11] Guo X, Liu J, Chen Y. FitCoach: Virtual fitness coach empowered by wearable mobile devices. IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, INFOCOM 2017 - IEEE Conference on Computer Communications, IEEE. May 2017:1-9.
doi:10.1109/INFOCOM.2017.8057208 [Accessed: Apr. 11, 2025]