MIDDLE EAST TECHNICAL UNIVERSTIY

DEPARTMENT OF COMPUTER ENGINEERING

SUMMER PRACTICE REPORT

CENG 300

STUDENT NAME: Ali Fırat ÇELİK

ORGANIZATION NAME: METU Applied Intelligence Research Laboratory - Informatics Institute

ADDRESS: METU Informatics Institute, Üniversiteler Mahallesi, Dumlupınar Bulvarı, No:1, 06800, Ankara, Turkey

START DATE: 25.08.2025

END DATE: 20.09.2025

TOTAL WORKING DAYS: 20

STUDENT'S SIGNATURE	**ORGANIZATION APPROVAL**

TABLE OF CONTENTS

1. INTRODUCTION

This report summarizes my four-week summer internship at the Informatics Institute, AIRLab research group, from August 25 to September 20, 2025. I worked under the supervision of Assoc. Prof. Dr. Erdem Akagündüz on an augmented shadow system project that integrates real-time computer vision techniques with a camera–projector setup.

At the beginning of my internship, I did not have extensive knowledge about deep learning or computer vision. To build a strong foundation, I started by watching the CS231n Winter 2016 lecture videos on YouTube. These lectures taught me the main concepts of convolutional neural networks (CNNs) and how they are used in image understanding, including topics such as convolutional layers, pooling operations, backpropagation, and the role of large-scale datasets. This theoretical background helped me understand the practical work I would be doing during the internship.

The Organization

METU Applied Intelligence Research Laboratory (AIRLab) is a research group within the Informatics Institute at Middle East Technical University. The laboratory focuses on several research areas including computer vision, geoinformatics, earthquake engineering, and decision support systems. The lab brings together researchers, graduate students, and undergraduate interns to work on cutting-edge projects combining artificial intelligence with real-world applications.

Project Overview

My primary responsibility during the internship was working on human pose estimation for the Augmented Shadow System project. The project aims to create an interactive installation where a camera captures human movement in real time, processes the video stream using pose estimation algorithms, and projects corresponding visual effects through a projector.

My specific role had two main focuses: implementing and evaluating human pose estimation models (MoveNet and PoseNet), and setting up and configuring the NVIDIA Jetson Nano embedded system to run these models efficiently for real-time performance. Throughout the internship, I gained hands-on experience in embedded systems configuration, model optimization for edge devices, computer vision algorithms, and the integration of hardware and software components into a cohesive interactive system.

2. INFORMATION ABOUT PROJECT

The project I worked on during my internship at AIRLab was called the Augmented Shadow System. The main goal of this project was to create an interactive system that uses a camera and a projector together to show visual effects that move with a person's body in real time. When a person stands in front of the camera, the system detects their body movements and projects corresponding effects—like shadows or colors—onto a surface instantly.

The project is expected to evolve into an industry-partnered project in the long term. During my internship, I contributed to an early research and development phase, working toward a prototype that initially runs on laptop software. The system architecture includes multiple submodules that work together:

- Main Camera/Projector Module for real-time frame acquisition and projection
- Projector Calibration Module for accurate spatial alignment
- Pose Estimation Module utilizing deep learning models (PoseNet and MoveNet)
- 2D Visual Effects Module for interactive visual feedback

My main responsibility in the project was human pose estimation, which means detecting and tracking human body joints from camera images. I worked with two popular models: MoveNet and PoseNet. I tested these models under different lighting and movement conditions to find out which one worked faster and more accurately for real-time use on embedded hardware.

Besides my own part, I also learned how the other components of the system worked, including camera calibration, projector alignment, and visual rendering. By understanding how each part fits together, I was able to see how computer vision models can be integrated into real-world interactive systems.

2.1. ANALYSIS PHASE

During the analysis phase, we identified the key requirements and constraints for the augmented shadow system. The primary functional requirements included:

- Real-time detection of human body keypoints
- Reliable performance under varying lighting conditions
- Low-latency processing to minimize delay between movement and visual feedback

The technical constraints we identified were:

- Limited computational resources on the Jetson Nano (128-core NVIDIA Maxwell-based GPU and quad-core ARM A57 CPU with 2GB RAM)
- Need for optimized model formats (TensorFlow Lite) to achieve real-time performance

Problems Encountered: A key challenge during the analysis phase was determining which pose estimation model would be most suitable for our hardware constraints. Many state-of-the-art models are designed for high-performance GPUs and would not run at acceptable speeds on the Jetson Nano. We needed to find a balance between accuracy and computational efficiency.

Solution: After researching available models, we decided to evaluate two models specifically designed for edge devices: Google's MoveNet and PoseNet. Both have optimized versions (TensorFlow Lite) that can run on mobile and embedded hardware. We planned to conduct comparative testing to determine which model best met our requirements.

Methods Followed: We conducted a literature review of current pose estimation models, examined benchmark results on similar hardware platforms, and defined clear evaluation metrics (inference speed, accuracy, robustness to lighting changes, and resource utilization).

Key Learning Points: This phase taught me the importance of understanding hardware constraints before selecting algorithms. In research papers, models are often evaluated on powerful desktop GPUs, but real-world deployment requires considering the actual hardware environment. I learned to prioritize practical performance metrics (FPS, latency) alongside accuracy metrics.

2.2. DESIGN PHASE

In the design phase, we developed the system architecture and workflow for the augmented shadow system. The overall pipeline consists of the following stages:

1. **Video Capture:** The camera continuously captures frames.
2. **Preprocessing:** Each frame is resized and normalized to match the input requirements of the pose estimation model
3. **Pose Estimation:** The model processes the frame and outputs keypoint coordinates and confidence scores
4. **Postprocessing:** Keypoint coordinates are transformed from image space to projector space using calibration matrices
5. **Visual Rendering:** Visual effects are generated based on the detected pose
6. **Projection:** The rendered output is projected onto the display surface

My responsibility focused primarily on stages 2-3 (preprocessing and pose estimation), ensuring these components could run efficiently on embedded hardware.

Hardware Configuration Design: We designed the Jetson Nano setup to maximize performance:

- Power mode settings for maximum performance

Model Selection Design: We planned to implement and compare two models:

- **PoseNet:** A lightweight model that outputs heatmaps for each keypoint. Good for understanding basic pose estimation concepts and suitable for educational purposes.
- **MoveNet:** A newer model with better accuracy-speed tradeoff, available in Lightning (faster) and Thunder (more accurate) variants. Outputs direct keypoint coordinates with high precision.

Problems Encountered: During the design phase, we were not sure if the Jetson Nano would be fast enough for real-time use. The technical specifications said it should work, but we knew that actual performance is often different from what the numbers suggest.

Solution: We decided to test the speed and performance early in the project. This way, if the Jetson Nano was too slow, we could quickly change our plan by using smaller images or choosing a faster model.

Methods Followed: We drew diagrams to show how data moves through the system. We also set clear goals for how fast the system needed to be. These plans helped us make better decisions when we started building the actual system.

Key Learning Points: I learned that it is important to plan for testing from the beginning. By deciding early what we would measure and how we would measure it, we could make choices based on real results instead of guesses. I also learned that choosing a model is not just about accuracy—speed and hardware limitations are just as important for real-world applications.

2.3. IMPLEMENTATION PHASE

The implementation phase was the most intensive part of the internship, involving both software development and hardware configuration. My work focused on two main areas: setting up the Jetson Nano and implementing the pose estimation models.

2.3.1. Jetson Nano Setup and Configuration

Setting up the NVIDIA Jetson Nano proved to be one of the most challenging aspects of the project.

Operating System Installation: I had to flash the JetPack operating system onto the Jetson Nano 2GB Developer Kit. This process encountered significant difficulties:

- The standard SD card flashing method did not work reliably due to the Jetson Nano 2GB being end-of-life hardware
- NVIDIA no longer actively supports or maintains the SD card flashing tools for this device
- Multiple SD card flashing attempts failed with corruption errors and boot failures
- The recommended solution required using Force Recovery mode flashing from a Linux host PC

Force Recovery Mode Setup: The working solution required a more complex setup process:

- Setting up a Linux host machine (Ubuntu) for flashing
- Activating Force Recovery mode by placing a female-to-female jumper on header J12: connecting pin 10 (Force Recovery / FC REC) to pin 11(or 9) (GND)
- Connecting the Jetson Nano to the host PC via micro USB in Force Recovery mode
- Using NVIDIA SDK Manager or manual flashing commands from the host PC
- This method bypassed the unreliable SD card flashing approach
- The complete detailed process is documented on my website at `https://alifirat.xyz/jetson`

Initial Boot and System Configuration: After successful flashing, the first boot process took considerable time and required careful configuration:

- Setting up network connectivity by wifi dongle

- Configured the Jetson device for maximum performance by setting the power mode to `MAXN` and locking all CPU and GPU frequencies using `nvpmodel -m 0` and `jetson_clocks`
- Updating system packages and resolving dependency conflicts

Deep Learning Framework Installation: Installing TensorFlow and related libraries on the ARM-based Jetson Nano required specific procedures:

- Installing JetPack SDK which includes CUDA
- Installing TensorFlow and TensorFlow Lite runtime for optimized inference
- Setting up Python virtual environment to manage dependencies
- Installing OpenCV for image processing tasks
- Configuring GPU acceleration for TensorFlow Lite

Problems Encountered: The Jetson Nano setup was extremely problematic. The SD card flashing failed multiple times It stuck in boot screen only showing NVIDIA logo. We later discovered that the Jetson Nano 2GB Developer Kit has reached end-of-life, and the SD card flashing method does not work properly. NVIDIA does not appear to be actively working to fix this issue. The NVIDIA-recommended method was to flash from a ubuntu 18.04 host PC with the board in Force Recovery mode, which adds significant complexity to the setup process. Even after successful flashing, the first boot would sometimes freeze or fail to complete. The installation of deep learning frameworks was particularly difficult, with various dependency conflicts and version compatibility issues. Memory limitations during package installation also caused problems.

Solution: For the SD card issues, we eventually had to use the Force Recovery mode flashing method from a Linux host PC, following NVIDIA's recommended approach for the end-of-life hardware. This required additional hardware (a data transfer capable micro USB cable for the recovery mode connection) and access to a ubuntu 18.04 system even 20.04 did not work. The complete step-by-step process for setting up the Jetson Nano, including all the troubleshooting steps and solutions we discovered, is documented in detail on my website at `https://alifirat.xyz/jetson/setup/`. We tried multiple SD cards and eventually found that using Force Recovery method provided the most reliable results. For the boot freezing issues, we discovered that power supply stability is critical, and using a high-quality 5V/4A power adapter solved the problem.

For software installation, we followed NVIDIA's official documentation more carefully and used pre-built packages where available rather than building from source. We also created sufficient swap space on the SD card to handle memory-intensive operations during installation. Working closely with my supervisor Dr. Akagündüz helped resolve many of these technical challenges.

2.3.2. Pose Estimation Model Implementation

After successfully configuring the Jetson Nano, I implemented both PoseNet and MoveNet models:

PoseNet Implementation:

- Downloaded the pre-trained model from TensorFlow Hub
- Converted the model to TensorFlow Lite format for edge deployment
- Implemented preprocessing pipeline (resize to 353×353, normalize pixel values)
- Parsed model outputs to extract keypoint coordinates
- Implemented post-processing to filter low-confidence keypoints

MoveNet Implementation:

- Downloaded both Lightning and Thunder variants from TensorFlow Hub
- Converted models to TensorFlow Lite format
- Implemented preprocessing (resize to 192×192 for Lightning, 256×256 for Thunder)
- Extracted keypoint coordinates directly from model output (simpler than PoseNet)

2.4. TESTING PHASE

The testing phase involved comprehensive evaluation of both pose estimation models under various conditions:

2.4.1. Performance Benchmarking

I conducted systematic performance testing to measure inference speed and system resource usage:

Frame Rate Testing:

- Measured average FPS over extended periods under different conditions
- Recorded minimum and maximum FPS to assess consistency
- Tested both models with and without GPU acceleration
- Evaluated impact of input resolution on speed

Observations:

- MoveNet, especially the quantized Lightning variant, achieved near real-time performance on Jetson Nano
- PoseNet showed higher latency and struggled to maintain consistent frame rates
- GPU acceleration provided significant performance improvements for both models

2.4.2. Accuracy and Robustness Testing

To evaluate detection accuracy, we tested the models under various scenarios using sample images from the COCO validation dataset:

Test Scenarios:

- **Standard lighting conditions:** Both models performed well, detecting keypoints consistently
- **Low lighting (concert-like scenes):** MoveNet maintained better performance with more stable keypoint detection, while PoseNet produced more false positives and fragmented skeletons
- **Crowded scenes:** MoveNet was more robust, while PoseNet often incorrectly assigned keypoints to background individuals
- **Well-lit scenarios:** Both models produced reasonable results, but MoveNet predictions were cleaner and more consistent

Key Findings:

- **Accuracy:** MoveNet consistently provided more precise and stable localization of body keypoints across all scenarios
- **Robustness:** MoveNet was more reliable under challenging conditions such as low-light environments or partial occlusions
- **False Positives:** PoseNet tended to generate noisier predictions with more spurious keypoints in cluttered scenes

3. ORGANIZATION

3.1. ORGANIZATION AND STRUCTURE

METU Applied Intelligence Research Laboratory (AIRLab) operates within the Informatics Institute at Middle East Technical University. Based on my experience during the internship, the laboratory has the following characteristics:

Leadership and Supervision: The lab is directed by Assoc. Prof. Dr. Erdem Akagündüz, who oversees research projects and provides technical guidance to the research group. During my internship, Dr. Akagündüz maintained a very hands-on approach, being present on-site constantly and directly involved in technical discussions, problem-solving, and day-to-day supervision.

Team Composition: During my internship period (August-September 2025), the team working on the augmented shadow project consisted of:

- Assoc. Prof. Dr. Erdem Akagündüz (Project Supervisor)
- A group of 10 interns, including 4 summer interns from METU Computer Engineering Department (myself, Arda, Yusuf, and Ethem)

The lab environment was collaborative and supportive. Despite being undergraduate interns, we were treated as contributing team members and our ideas were valued in project discussions.

Research Focus Areas: According to the AIRLab website and my observations, the laboratory conducts research across multiple domains:

- Computer Vision
- Geoinformatics
- Earthquake Engineering
- Decision Systems
- Artificial Intelligence Applications

The interdisciplinary nature of the lab's work creates opportunities for learning across different application domains of artificial intelligence and computer vision.

Physical Infrastructure: The lab is equipped with necessary computational resources including workstations, development boards like NVIDIA Jetson devices, cameras, projectors, and other equipment needed for computer vision research. The workspace provided a conducive environment for both independent work and collaborative discussions.

3.2. METHODOLOGIES AND STRATEGIES USED IN THE ORGANIZATION

Based on my four-week experience at AIRLab, I observed the following methodologies and working strategies:

Work Environment and Schedule:

- The work environment was primarily on-site, with team members present at the lab throughout the working day
- Dr. Akagündüz was consistently available on-site, providing immediate support and guidance when technical problems arose
- This constant availability of supervision was particularly valuable during challenging phases

Project Management Approach:

- Regular meetings and discussions about progress, challenges, and next steps
- Clear communication of project goals and individual responsibilities
- Flexibility to adjust plans based on technical discoveries and challenges encountered
- Emphasis on both learning and practical output

Technical Methodology:

- Iterative development approach: implement, test, analyze, optimize, repeat
- Emphasis on systematic evaluation and benchmarking to support technical decisions with data
- Use of standard datasets (COCO) and established models for reproducibility
- Documentation of results and findings for future reference
- Collaborative problem-solving when challenges arose

Learning and Development:

- Encouragement of self-study (CS231n lectures) alongside practical work
- Hands-on learning through real implementation tasks rather than just theoretical study
- Opportunity to understand the full system architecture, not just isolated components
- Freedom to explore solutions and learn from failures

Collaboration Style:

- Open communication between team members
- Knowledge sharing about different aspects of the project
- Collective problem-solving sessions when facing technical difficulties
- Peer learning between interns working on related components

Research Approach: The lab follows a research-oriented approach that balances academic rigor with practical application. The augmented shadow project, while targeting eventual real-world deployment, was approached methodically with proper analysis, design, implementation, and testing phases. This structured approach helped ensure that technical decisions were well-justified and that the project could serve as a foundation for future development.

Observations and Reflections:

Having Dr. Akagündüz present at the lab all the time was extremely helpful. Whenever we had problems or questions, we could ask him right away and get immediate help. This made learning much faster and helped us solve problems quickly. This hands-on approach created a very good learning environment for us as undergraduate students.

The friendly and open atmosphere at the lab made it easy to ask questions without feeling nervous. We were encouraged to understand how the whole system works, not just our own small part. This helped us see how different parts of a project connect together in real applications.

We were also given enough time to properly solve difficult problems instead of being rushed to move on. This showed us that in real research work, unexpected problems are normal and should be fixed properly, not just worked around quickly.

4. CONCLUSION

During this four-week internship at AirLab, I learned a lot about computer vision and embedded systems. I went from knowing very little about deep learning to successfully running pose estimation models on a Jetson Nano.

The project taught me to be patient and think of different solutions when the standard approach fails.

After testing both models, I found that MoveNet is clearly better than PoseNet. It's faster, more accurate, and works better in difficult situations like dark rooms or crowded scenes. PoseNet is simpler to start with, but it makes more mistakes and runs slower. This showed me that for real-time applications, you can't just pick the easiest option—you need to consider both speed and accuracy.

I also learned how to work in a research environment. Breaking problems into smaller steps, documenting what I did, and working with my teammates were all important skills I developed. Having Dr. Akagündüz available to help whenever we had problems made the learning process much faster.

The CS231n lectures gave me good theoretical knowledge, but actually deploying models on real hardware taught me much more.

This internship helped me discover that I enjoy making programs work on limited hardware and building real systems, not just studying theory. The experience gave me confidence to handle technical challenges and showed me what research work is actually like.

I want to thank Dr. Erdem Akagündüz for his help, my fellow interns Arda, Yusuf, and Ethem for their support, and the Informatics Institute for this opportunity.