Design Chapter - Arduino-Based Automatic Glass Door System

# 1. Introduction

The design of the Arduino-based automatic glass door system aims to combine automation, safety, and modularity in a real-world access control application. This chapter presents the conceptualization, planning, and execution involved in building the system. It highlights key design choices, component integration strategies, and optimization methods to ensure reliable and smooth door operation. Both hardware and software were considered from a system-level perspective to ensure interoperability and robustness.

# 2. Design Considerations

- \*\*Cost Efficiency\*\*: Components were chosen based on affordability without compromising core functionality.  
- \*\*Modularity\*\*: The system can be modified or scaled easily.  
- \*\*Power Constraints\*\*: Operates on 24V DC SMPS, which is safe and manageable.  
- \*\*Mechanical Constraints\*\*: The glass door requires precise control to prevent damage.  
- \*\*Safety\*\*: Emergency stop logic and PIR sensor ensure human safety.

# 3. Design Calculations (Simplified)

- \*\*Encoder Pulses\*\*: Used to determine maxPosition and track door travel distance.  
- \*\*Motor Power Requirement\*\*: Motor selected based on torque required to move glass door.  
- \*\*Timing\*\*: Delay logic and motion duration calculated based on encoder feedback.

# 4. Iterative Design Process

The project went through multiple revisions. Initially, the system was button-controlled without motion detection. Later, PIR sensors were integrated for automatic operation. Encoder feedback and speed control were implemented after initial fixed-speed trials led to abrupt starts/stops. These changes improved user safety and system efficiency.

# 5. Future Enhancements

- Replace Arduino Uno with ESP32 for Wi-Fi capabilities.  
- Add mobile app interface for remote door control.  
- Integrate load cell sensors for additional safety.  
- Log data such as usage frequency or maintenance alerts.

# 6. Risk Analysis and Mitigation

|  |  |  |
| --- | --- | --- |
| Risk | Possible Cause | Mitigation Strategy |
| Door doesn’t stop | Encoder failure | Relay override, use homing sensor fallback |
| Door doesn’t auto-close | PIR stuck or failed | Fallback to timer-based closing |
| Motor overheating | Excessive runtime | PWM control + cooling intervals |
| Unexpected opening | False PIR trigger | Add logic to verify human presence |

# 7. Hardware-Software Integration Strategy

The system software was modularized into control blocks for sensors, actuators, and feedback systems. Encoder interrupts were used for precise position tracking, while control logic handled safe transitions and smooth acceleration. State machines managed system behavior like homing, opening, and auto-closing.

# 8. Real-World Inspiration

The system design drew inspiration from commercial automatic doors used in malls and hospitals. Such systems prioritize aesthetics, seamless operation, and safety—factors that were mirrored in the current design by implementing smooth speed control, PIR-based activation, and safe hardware configurations.