

Σχολή Θετικών Επιστημών και Τεχνολογίας

Τμήμα Πληροφορικής

Πτυχιακή Εργασία

Σύστημα απομακρυσμένου ελέγχου περιστροφής θόλου αστεροσκοπείου

Κωνσταντίνος Μιχαηλίδης

Επιβλέπων καθηγητής: Αθανάσιος Κακαρούντας



Πάτρα, Ιούλιος 2026

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή (Κωνσταντίνου Μιχαηλίδη) που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.



Σύστημα απομακρυσμένου ελέγχου περιστροφής θόλου αστεροσκοπείου

Κωνσταντίνος Μιχαηλίδης

Επιτροπή Επίβλεψης Πτυχιακής Εργασίας

Επιβλέπων Καθηγητής:

Δρ. Αθανάσιος Κακαρούντας

Συνεργαζόμενο Εκπαιδευτικό Προσωπικό (ΣΕΠ)

Ελληνικό Ανοικτό Πανεπιστήμιο (ΕΑΠ)

Επιτροπή κρίσης:

Δρ. Ευάγγελος Τοπάλης

Συνεργαζόμενο Εκπαιδευτικό Προσωπικό
(ΣΕΠ)

Ελληνικό Ανοικτό Πανεπιστήμιο (ΕΑΠ)

Δρ. Ιωάννης Κουρέτας

Συνεργαζόμενο Εκπαιδευτικό Προσωπικό
(ΣΕΠ)

Ελληνικό Ανοικτό Πανεπιστήμιο (ΕΑΠ)

Πάτρα, Ιούλιος 2026

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Αθανάσιο Κακαρούντα που με εμπιστεύτηκε, όπως και για την καθοδήγηση που μου παρείχε για την εκπόνηση της εν λόγω πτυχιακής εργασίας.

Θα ήθελα να ευχαριστήσω επίσης το προσωπικό του αστεροσκοπείου της Υπάτης για τη συνεργασία του.

Περίληψη

Η παρούσα πτυχιακή εργασία αφορά το σχεδιασμό και την ανάπτυξη ενός συστήματος ελέγχου εξ αποστάσεως κίνησης θόλου αστεροσκοπείου. Περιγράφει μια λύση χαμηλού κόστους η οποία περιλαμβάνει μελέτη και ανάπτυξη λογισμικού ενός συστήματος απομακρυσμένου ελέγχου χρησιμοποιώντας την πλατφόρμα Arduino και βασίζεται στα χαρακτηριστικά του αστεροσκοπείου της Υπάτης. Ένα από τα πλεονεκτήματα του συστήματος είναι η εξοικονόμηση χρόνου καθώς ο χρήστης δεν χρειάζεται να μεταβεί στο αστεροσκοπείο προκειμένου να κάνει τις ουράνιες παρατηρήσεις του.

Πιο αναλυτικά, η πτυχιακή εργασία ασχολείται με την επιλογή κατάλληλων αισθητήρων θέσης που καταγράφουν τη γωνιακή θέση του θόλου και τη θέση του τηλεσκοπίου, αισθητήρα για την παρακολούθηση του καιρού, ενεργοποιητών (actuators) για την κίνηση του θόλου και το κλείσιμο/άνοιγμα της θυρίδας καθώς και με την επικοινωνία μέσω διαδικτύου για την απομακρυσμένη μετάδοση εντολών και ανατροφοδότησης των αισθητήρων θέσης. Η λειτουργία των αισθητήρων και των ενεργοποιητών θα ελέγχεται από λογισμικό, τμήμα του οποίου θα τρέχει τοπικά και ένα άλλο απομακρυσμένα. Το απομακρυσμένο λογισμικό αποτελείται από το τμήμα που θα τρέχει στην πλατφόρμα Arduino cloud και από το λογισμικό που θα τρέχει στο τερματικό του χρήστη. Ο χειρισμός εκτός από απομακρυσμένα θα γίνεται και τοπικά.

Το τελικό σύστημα θα παρέχει εργαλεία για τον τοπικό και απομακρυσμένο έλεγχο του θόλου και της θυρίδας του αστεροσκοπείου Υπάτης και το συγχρονισμό της θέσης του θόλου με τη θέση του τηλεσκοπίου. Επιπλέον παρέχονται δικλείδες ασφαλείας τόσο για την απομακρυσμένη πρόσβαση όσο και για τη λειτουργία του συστήματος και την προστασία του εξοπλισμού. Η λύση που προτείνεται είναι μια οικονομική εναλλακτική για την αναβάθμιση του αστεροσκοπείου της Υπάτης σε σχέση με ανάλογες του εμπορίου.

Λέξεις – Κλειδιά

Αστεροσκοπείο, Τηλεσκόπιο, Θόλος, Υπάτη, Arduino, IoT

Abstract

This thesis focuses on the design and development of a remote-control system for the movement of an observatory dome. It describes a low-cost solution that involves the study and development of software for a remote-control system using the Arduino platform, based on the specific characteristics of the Ypati Observatory. One of the system's main advantages is that it saves time, as the user does not have to travel to make his celestial observations.

More specifically, the thesis deals with the selection of suitable position sensors that track the dome's angular position and the telescope's position, a weather sensor, actuators (for dome rotation and for opening/closing its shutter), as well as the internet-based communication for transmitting remote commands and receiving feedback from the position sensors. The operation of the sensors and actuators will be controlled by software, part of which will run locally and another part remotely. The remote software includes the portion that runs on the Arduino cloud platform and the software that runs on the user's terminal. Operation can be performed both locally and remotely.

The final system will provide tools for both local and remote control of the Ypati Observatory's dome and its shutter, as well as for synchronizing the dome's position with the telescope's position. In addition, safety measures are included to ensure secure remote access, proper system operation, and equipment protection. The proposed solution is a cost-effective alternative for upgrading the Ypati Observatory compared to similar commercial solutions.

Keywords

Observatory, Telescope, Dome, Ypati, Arduino, IoT

Περιεχόμενα

Περίληψη.....	v
Abstract	vi
Περιεχόμενα	vii
Κατάλογος Εικόνων / Σχημάτων	ix
Κατάλογος Πινάκων	x
Συντομογραφίες & Ακρωνύμια.....	xi
1. Εισαγωγή.....	1
1.1 Υπάρχοντα Συστήματα – Εξοπλισμός Αστεροσκοπείου	1
1.2 Σύστημα προς υλοποίηση	3
1.3 Οργάνωση πτυχιακής	5
2. Αρχές λειτουργίας	6
2.1 Αστεροσκοπείο.....	6
2.1.1 Αστεροσκοπείο Υπάτης.....	6
2.2 Τηλεσκόπιο.....	7
2.3 Θόλος.....	8
2.3.1 Σύστημα περιστροφής του θόλου.....	9
2.3.2 Παρακολούθηση της θέσης του θόλου.....	10
2.3.3 Έλεγχος της θυρίδας.....	11
2.3.4 Θόλος Αστεροσκοπείου Υπάτης	11
2.4 Ψηφιακά συστήματα σύνδεσης, συλλογής, αποθήκευσης και μετάδοσης μετρήσεων 12	
2.5 Αυτοματισμοί ελεγχόμενων από απόσταση (τηλεμετρία) για τηλεσκόπια.....	13
2.5.1 Πλεονεκτήματα των αυτοματισμών	14
3. Απαιτήσεις Συστήματος, Προδιαγραφές, Πλαίσιο Λειτουργίας.....	15
3.1 Μη Λειτουργικές Απαιτήσεις.....	16
3.1.1 Αξιοπιστία	16
3.1.2 Απόκριση και Ταχύτητα εκτέλεσης	16
3.1.3 Ασφάλεια.....	16
3.1.4 Επεκτασιμότητα / Upgradability	16
3.1.5 Ευχρηστία.....	16
3.1.6 Συντήρηση.....	16
3.2 Λειτουργικές Απαιτήσεις	17
3.2.1 Βασικός Έλεγχος Θόλου	17
3.2.2 Αυτόματος συγχρονισμός θέσης θόλου με το τηλεσκόπιο	17
3.2.3 Έλεγχος Θυρίδας.....	17
3.2.4 Επιστροφή του θόλου στην αρχική του θέση.....	18
3.2.5 Λειτουργίες ασφαλείας	18
3.3 Πλαίσιο Λειτουργίας	19
4. Σχεδιασμός Συστήματος	20
4.1 Τοπολογία προτεινόμενου Συστήματος	20
4.2 Λειτουργία των στοιχείων του συστήματος.....	21
4.3 Επικοινωνία των στοιχείων με τον κεντρικό ελεγκτή.....	22
4.4 Μορφή ενημέρωσης της κατάστασης του συστήματος	23
4.5 Περιπτώσεις Χρήσης (use cases)	23
5. Υλοποίηση Συστήματος.....	31
5.1 Εισαγωγή.....	31

5.2	Συστατικά στοιχεία του μοντέλου.....	31
5.2.1.	Κάμερα εσωτερικού χώρου TAPO C210 για την προσομοίωση του θόλου. ...	31
5.2.2	Arduino UNO R4 WiFi.....	33
5.2.3	Μονοπολικός βηματικός κινητήρας και πλακέτα οδήγησης	34
5.2.4	Hall Sensor Board	36
5.2.5	Modulino Knob (rotary encoder)	39
5.2.6	Modulino Buttons.....	40
5.2.7	Rain Sensor Module.....	40
5.3	Συναρμολογημένο σύστημα.....	41
5.4	Περιγραφή του κυκλώματος	42
6.	Απομακρυσμένος Έλεγχος.....	44
7.	Λογισμικό εφαρμογής.....	49
7.1	Εισαγωγή στη μη-αποκλειστική εκτέλεση (Non-Blocking Architecture)	49
7.2	Ο Κύκλος Χρονισμού (timingState)	49
7.3	Υπο-Μηχανές Καταστάσεων (Sub-State Machines) για την Κίνηση.....	50
7.4	Αλγόριθμος Αυτόματου Συγχρονισμού (Auto Sync) και Συντομότερης Διαδρομής	51
7.5	Ανάγνωση Ψηφιακών Κωδικοποιητών (encoders) και Χωρική Αντιστοίχιση.....	52
8.	Τρόπος Λειτουργίας.....	54
8.1	Εκκίνηση Συστήματος	54
8.2	Έλεγχος θυρίδας.....	54
8.3	Περιστροφή του θόλου.....	55
8.4	Απομακρυσμένος έλεγχος.....	56
9.	Ενημέρωση της Κατάστασης του Συστήματος (visualization).....	57
10.	Αξιολόγηση του έργου	59
10.1.	Αξιολόγηση Μη Λειτουργικών Απαιτήσεων.....	59
10.2	Αξιολόγηση λειτουργικών απαιτήσεων	60
10.3	Λειτουργική Αξιολόγηση υλικού και τεχνολογιών.....	61
10.3.1	Κεντρική μονάδα ελέγχου:.....	62
10.3.2	Οικοσύστημα Modulino (I ² C):.....	63
10.3.3	Γιατί Rotary Encoder και όχι Γυροσκόπιο:.....	64
10.3.4	Αισθητήρες Hall:.....	65
10.3.5	Βηματικοί Κινητήρες (Stepper Motors):.....	66
10.3.6	Αισθητήρας Βροχής	66
10.3.7	Αντιμετώπιση Προβλημάτων και Επανασχεδιασμός.....	67
10.4	Ποιοτική και Ποσοτική Αξιολόγηση Λογισμικού	68
10.4.1	Ποιοτική Αξιολόγηση Λογισμικού	69
10.4.2	Ποσοτική Αξιολόγηση Λογισμικού	69
10.4.3	Επικοινωνία.....	70
10.5	Αξιολόγηση Ενημέρωσης Κατάστασης Συστήματος (Visualization)	71
10.6	Επίτευξη Στόχων	73
10.7	Μελλοντικές Προεκτάσεις	74
11.	Προσαρμογή του συστήματος στο αστεροσκοπείο της Υπάτης.....	76
12.	Συμπεράσματα - Προτάσεις	77
	Βιβλιογραφία – Ηλεκτρονικές Πηγές	79
	Παράρτημα Α: ΚΩΔΙΚΑΣ	81

Κατάλογος Εικόνων / Σχημάτων

Εικόνα 1 - Πίνακας Λειτουργίας του Θόλου της Υπάτης.....	3
Εικόνα 2 - Κινητήρες θόλου και θυρίδας.....	3
Εικόνα 3 - Ταρο C210	5
Εικόνα 4 - Αστεροσκοπείο της Υπάτης	6
Εικόνα 5 - Τηλεσκόπιο Υπάτης	7
Εικόνα 6 - Θόλος Αστεροσκοπείου	8
Εικόνα 7 - Θόλος τοποθετημένος σε πύργο.....	9
Εικόνα 8 - Διακόπτης ορίων κίνησης (Limit Switch).....	10
Εικόνα 9 - Αισθητήρας θέσης (Home Sensor).....	10
Εικόνα 10 - Κωδικοποιητής (Encoder)	10
Εικόνα 11 - Θυρίδα	11
Εικόνα 12 - Θόλος Αστεροσκοπείου της Υπάτης.....	11
Εικόνα 13 - Πλακέτα ελέγχου του εμπορίου	13
Εικόνα 14 - Τοπολογία του Συστήματος.....	20
Εικόνα 15 - Διάγραμμα καταστάσεων	24
Εικόνα 16 - Κάμερα ΤΑΡΟ C210	32
Εικόνα 17 - Αποσυναρμολόγηση βάσης και τοποθέτηση κινητήρα.....	32
Εικόνα 18 - Επάνω τμήμα μετά την αφαίρεση της βάσης	33
Εικόνα 19 - Arduino UNO R4 WiFi	33
Εικόνα 20 - Βηματικός κινητήρας.....	34
Εικόνα 21 - Step motor Driver	34
Εικόνα 22 - Διάγραμμα σύνδεσης βηματικού κινητήρα	35
Εικόνα 23 - Παράδειγμα σύνδεσης βηματικού κινητήρα	35
Εικόνα 24 - Συναρμολογημένος driver και κινητήρας.....	35
Εικόνα 25 - Πλακέτα αισθητήρα Hall.....	37
Εικόνα 26 - Συναρμολόγηση του αισθητήρα Hall	37
Εικόνα 27 - Διάγραμμα σύνδεσης αισθητήρα Hall.....	38
Εικόνα 28 - Παράδειγμα σύνδεσης αισθητήρα Hall	38
Εικόνα 29 - Knob για την εξομοίωση περιστροφής τηλεσκοπίου	39
Εικόνα 30 - Knob τοποθετημένο στον περιστρεφόμενο θόλο	39
Εικόνα 31 - Αισθητήρας βροχής	40
Εικόνα 32 - Συναρμολογημένο σύστημα	41
Εικόνα 33 - Διάγραμμα συνδεσμολογίας.....	41
Εικόνα 34 - Screenshot του ολοκληρωμένου dashboard	45

Κατάλογος Πινάκων

Πίνακας 1 – Παράδειγμα σύνδεσης Step motor driver	35
Πίνακας 2 – Παράδειγμα σύνδεσης πλακέτας αισθητήρα hall	37
Πίνακας 3 – Αισθητήρες	43
Πίνακας 4 – Cloud Εντολές Ελέγχου	47
Πίνακας 5 - Ενδείκτες κατάστασης του συστήματος.....	48
Πίνακας 6 - Ποσοτική αξιολόγηση εξοικονόμησης κινήσεων μέσω του αλγορίθμου Shortest Path.....	70

Συντομογραφίες & Ακρωνύμια

ASCOM	AStronomy Common Object Model
HTTP	Hypertext Transfer Protocol
INDI	Instrument Neutral Distributed Interface
IOT	Internet Of Things
MQTT	Message Queuing Telemetry Transport
PWM	Pulse Width Modulation
SPI	Serial Protocol Interface
USB	Universal Serial Bus
I ² C	Inter-Integrated Circuit (I2C) Protocol
FSM	Finite State Machine - Μηχανή Πεπερασμένων Καταστάσεων

1. Εισαγωγή

1.1 Υπάρχοντα Συστήματα – Εξοπλισμός Αστεροσκοπείου

Τα *αστεροσκοπεία* είναι ειδικά διαμορφωμένοι χώροι σχεδιασμένοι για την παρατήρηση και τη μελέτη ουράνιων σωμάτων και φαινομένων. Η βασική τους λειτουργία εξαρτάται από τη συνεργασία διαφορετικών συστημάτων και τεχνολογιών, τα οποία εξασφαλίζουν την ακρίβεια των παρατηρήσεων και τη σταθερότητα του εξοπλισμού. Η θέση και η κατασκευή των αστεροσκοπειών επιλέγεται με γνώμονα την ελάχιστη επιρροή από παρεμβολές και συνήθως κατασκευάζονται σε απομακρυσμένες περιοχές καθώς και σε περιοχές με κάποιο υψόμετρο. Στην Ελλάδα λειτουργούν αρκετά αστεροσκοπεία μεταξύ αυτών το Αστεροσκοπείο Αθηνών με τους αστρονομικούς σταθμούς Πεντέλης και Κρυονερίου Κορινθίας, τα πανεπιστημιακά αστεροσκοπεία Αθηνών, Θεσσαλονίκης, Πατρών, Ιωαννίνων, Υπάτης, του Σκίνακα στην Κρήτη και άλλα.

Ο *θόλος* του αστεροσκοπείου είναι μια κατασκευή που προστατεύει τα τηλεσκόπια από τις περιβαλλοντικές συνθήκες (όπως βροχή, άνεμο, υγρασία και σκόνη) ενώ ταυτόχρονα παρέχει δυνατότητα παρατήρησης μέσω μιας ειδικά σχεδιασμένης θυρίδας. Η δυνατότητα περιστροφής του θόλου σε συγχρονισμό με το τηλεσκόπιο και μαζί με το άνοιγμα της θυρίδας είναι απαραίτητα για την ευθυγράμμιση του τηλεσκοπίου με τα ουράνια αντικείμενα.

Το *τηλεσκόπιο* είναι η «καρδιά» του αστεροσκοπείου. Ένα εξελιγμένο τηλεσκόπιο διαθέτει ακρίβεια προσανατολισμού και είναι σε θέση να ακολουθεί την κίνηση των ουράνιων σωμάτων. Με τη χρήση κατόπτρων ή φακών, συγκεντρώνει το φως από τα ουράνια σώματα, κάνοντας τα αντικείμενα πιο φωτεινά και ευδιάκριτα.

Ένα σύγχρονο αστεροσκοπείο διαθέτει επίσης *συστήματα αυτοματισμού* όπως:

- *Αυτοματισμό περιστροφής θόλου* ο οποίος συγχρονίζει το θόλο με το τηλεσκόπιο. Ο θόλος διαθέτει σύστημα παρακολούθησης και συγχρονισμού με το τηλεσκόπιο ώστε κάθε φορά που το τηλεσκόπιο στοχεύει σε μια περιοχή του ουρανού, ο θόλος περιστρέφεται ώστε η θυρίδα του θόλου να ευθυγραμμιστεί με τη σκόπευση του τηλεσκοπίου. Παραδείγματα:
 - Μέσω αισθητήρων που παρακολουθούν τη γωνιακή θέση του θόλου και του τηλεσκοπίου. Οι αισθητήρες θέσης είναι μαγνητικοί ή οπτικοί και

- συνδέονται σε έναν μικροελεγκτή προκειμένου με τη βοήθεια του λογισμικού να γίνει ο συγχρονισμός του θόλου με το τηλεσκόπιο.
- Μέσω κωδικοποιητή (encoder) τοποθετημένου στο μηχανισμό περιστροφής του θόλου ο οποίος με τη βοήθεια μικροελεγκτή στον οποίο συνδέεται επίσης και ο κωδικοποιητής του τηλεσκοπίου, υπολογίζει τη θέση τους σε μοίρες και στη συνέχεια το λογισμικό του μικροελεγκτή συγχρονίζει τη γωνία του θόλου με αυτή του τηλεσκοπίου.
 - Οπτικά, μέσω της χρήσης καμερών και οπτικών αισθητήρων παρακολούθησης της θέσης του τηλεσκοπίου και με τη βοήθεια τεχνητής νοημοσύνης συγχρονίζεται η θέση του θόλου με του τηλεσκοπίου.
 - *Αυτόματο έλεγχο της θυρίδας* και χρήση αισθητήρων παρακολούθησης του καιρού ώστε σε περίπτωση άσχημων καιρικών συνθηκών να γίνεται το αυτόματο κλείσιμο της θυρίδας για την προστασία του εξοπλισμού του αστεροσκοπείου.
 - *Απομακρυσμένο έλεγχο* χωρίς την ανάγκη φυσικής παρουσίας. Επιτρέπει σε επιστήμονες οπουδήποτε στον κόσμο να χρησιμοποιούν τα τηλεσκόπια και τα συστήματα ενός αστεροσκοπείου ακόμη και αν αυτό βρίσκεται σε απομακρυσμένες ή δυσπρόσιτες τοποθεσίες. Εκτός από το απαιτούμενο υλικό περιλαμβάνει επίσης ειδικό λογισμικό για την περιστροφή του θόλου, την ευθυγράμμιση του τηλεσκοπίου και τη συλλογή δεδομένων σε πραγματικό χρόνο. Συγκεκριμένα, γίνεται μετάδοση της εικόνας του τηλεσκοπίου όπως επίσης και μετάδοση μέσω διαδικτύου των εντολών ελέγχου της περιστροφής του θόλου με ταυτόχρονη λήψη ανατροφοδότησης της θέσης και πιθανής διόρθωσης αυτής. Οι εντολές ελέγχου κίνησης λαμβάνονται από μια τοπική μονάδα στο αστεροσκοπείο η οποία ελέγχει τους ενεργοποιητές (actuators).

Συμπεράσματα: Σήμερα, μπορεί κανείς με μια αναζήτηση στο διαδίκτυο να βρει έτοιμες και αξιόπιστες λύσεις αυτοματισμού του θόλου ενός αστεροσκοπείου [14], [1], [3], όμως είναι πολύ υψηλού κόστους με αποτέλεσμα να μη μπορούν να τα προμηθευτούν τα αστεροσκοπεία εκπαιδευτικού σκοπού όπως του αστεροσκοπείου της *Υπάτης*. Είναι ανάγκη λοιπόν να κατασκευαστεί ένα οικονομικό σύστημα το οποίο θα είναι αξιόπιστο έτσι ώστε τα εκπαιδευτικά αστεροσκοπεία να μπορούν να αξιοποιούν τη σύγχρονη τεχνολογία.

1.2 Σύστημα προς υλοποίηση

Το αστεροσκοπείο της *Υπάτης* διαθέτει έναν απλό τρόπο περιστροφής του θόλου μέσω διακοπών ελέγχου του κινητήρα που βρίσκονται εντός του αστεροσκοπείου.



Εικόνα 1 - Πίνακας Λειτουργίας του Θόλου της Υπάτης



Εικόνα 2 - Κινητήρες θόλου και θυρίδας

Σκοπός της παρούσας πτυχιακής εργασίας, είναι η ανάπτυξη ενός οικονομικού εναλλακτικού αυτοματοποιημένου συστήματος ελέγχου θόλου αστεροσκοπείου, το οποίο θα υποστηρίζει απομακρυσμένη λειτουργία μέσω διαδικτύου. Το σύστημα θα σχεδιαστεί για να εφαρμοστεί στο Αστεροσκοπείο της Υπάτης, με σκοπό να επιτρέπει την αποτελεσματική και ασφαλή διαχείριση του θόλου χωρίς την ανάγκη φυσικής παρουσίας.

Το σύστημα θα περιλαμβάνει:

- **Αυτόματο έλεγχο του θόλου:** Εξασφάλιση συγχρονισμού με το τηλεσκόπιο, ώστε το άνοιγμα του θόλου να παραμένει ευθυγραμμισμένο με την οπτική γραμμή παρατήρησης.
- **Απομακρυσμένη διαχείριση:** Ανάπτυξη διεπαφής χρήστη για τον έλεγχο του θόλου μέσω υπολογιστή ή κινητής συσκευής.
- **Σύστημα ασφαλείας:** Μηχανισμούς αυτόματης αντίδρασης σε περίπτωση δυσμενών καιρικών συνθηκών ή τεχνικών προβλημάτων.

Η μεθοδολογία που θα χρησιμοποιηθεί περιλαμβάνει ανάλυση απαιτήσεων, σχεδιασμό, ανάπτυξη και υλοποίηση. Θα μελετηθεί η υπάρχουσα υποδομή του Αστεροσκοπείου της Υπάτης, θα επιλεγεί το υλικό, θα επιλεγεί η πλατφόρμα (Arduino, Raspberry, ESP32) με το αντίστοιχο λογισμικό, θα γίνει ανάπτυξη λογισμικού που θα υποστηρίζει την απομακρυσμένη διαχείριση, θα γίνει επιλογή των αισθητήρων, συναρμολόγηση πρωτοτύπου και έλεγχος καλής λειτουργίας.

Λόγω της φύσης και της δυσκολίας του έργου, θα αναπτυχθεί και κατασκευαστεί ένα πρωτότυπο με τη βοήθεια υλικών και έτοιμων συσκευών του εμπορίου ώστε να δοκιμαστεί στην πράξη η προτεινόμενη λύση. Συγκεκριμένα, για την εξομοίωση του θόλου θα χρησιμοποιηθεί η βάση της περιστρεφόμενης κάμερας Taro C210. Η βάση θα αφαιρεθεί και θα τοποθετηθεί ανάποδα επάνω σε έναν βηματικό κινητήρα ιδίου τύπου με αυτόν της κάμερας, σκοπός του οποίου θα είναι η περιστροφή της. Στο επάνω τμήμα της βάσης που πλέον θα προσομοιώνει τον θόλο, θα τοποθετηθεί ένας encoder. Το υπόλοιπο τμήμα της κάμερας C210 (το επάνω) το οποίο περιλαμβάνει και την ίδια την κάμερα, θα χρησιμοποιηθεί για να προσομοιώσει την θυρίδα του θόλου, δηλαδή η κίνηση της κάμερας C210 προς τα πάνω/κάτω θα αντιστοιχεί με την άνοιγμα/κλείσιμο της θυρίδας του θόλου. Ένας δεύτερος encoder θα χρησιμοποιηθεί για την εξομοίωση της περιστροφής του τηλεσκοπίου. Ο encoder αυτός θα περιστρέφεται χειροκίνητα και θα είναι ιδίου τύπου με αυτόν που θα τοποθετηθεί στο θόλο ώστε να μπορούν να συγχρονίζονται. Ο σκοπός είναι ο θόλος να παρακολουθεί σε πραγματικό χρόνο την περιστροφή και τη θέση του τηλεσκοπίου μέσω των encoders ώστε να συγχρονίζει τη θέση του με αυτό.



Εικόνα 3 - Tapo C210

1.3 Οργάνωση πτυχιακής

Στο δεύτερο κεφάλαιο γίνεται μια θεωρητική γνωριμία με το αντικείμενο της εργασίας, περιγράφοντας τα λειτουργικά μέρη ενός σύγχρονου αστεροσκοπείου (θόλος, τηλεσκόπιο) και των αρχών λειτουργίας τους, γίνεται μια εισαγωγή στις τεχνολογίες απομακρυσμένου ελέγχου, και παρουσιάζονται υπάρχοντα συστήματα απομακρυσμένου ελέγχου.

Στο τρίτο κεφάλαιο καταγράφονται οι λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος, ορίζονται οι προδιαγραφές και γίνεται μια αναφορά στο πλαίσιο λειτουργίας. Στο τέταρτο κεφάλαιο δομείται η αρχιτεκτονική και παρουσιάζονται οι περιπτώσεις χρήσης του συστήματος.

Στο πέμπτο κεφάλαιο παρουσιάζονται τα μέρη του συστήματος καθώς και ο τρόπος συναρμολόγησης και ενοποίησης.

Στο έκτο κεφάλαιο περιγράφεται ο τρόπος σύνδεσης του συστήματος στο cloud.

Στο έβδομο κεφάλαιο περιγράφεται το λογισμικό του συστήματος.

Στο όγδοο κεφάλαιο παρουσιάζεται ο τρόπος λειτουργίας του συστήματος.

Στο ένατο κεφάλαιο περιγράφεται ο τρόπος υλοποίησης της ενημέρωσης του συστήματος.

Στο δέκατο κεφάλαιο εξετάζεται ο βαθμός ικανοποίησης των απαιτήσεων.

Στο ενδέκατο κεφάλαιο δίνεται ο προτεινόμενος τρόπος εφαρμογής στο αστεροσκοπείο της Υπάτης.

Τέλος, στο δωδέκατο κεφάλαιο αναλύονται τα συμπεράσματα.

2. Αρχές λειτουργίας

2.1 Αστεροσκοπείο

Από αρχαιοτάτων χρόνων ο άνθρωπος παρατηρούσε τα ουράνια σώματα και φαινόμενα. Στην αρχή με γυμνό οφθαλμό και αργότερα χρησιμοποιώντας τεχνικά μέσα τα οποία δημιούργησε για αυτό το σκοπό.

Προκειμένου να παρατηρήσουμε και να μελετήσουμε τα ουράνια σώματα και φαινόμενα χρειαζόμαστε κατάλληλο εξοπλισμό και μέσα, εγκατεστημένα σε ένα κατάλληλο διαμορφωμένο χώρο σε μια κατάλληλη τοποθεσία καθώς και το κατάλληλο επιστημονικό προσωπικό. Ο ελάχιστος εξοπλισμός που χρειαζόμαστε είναι το τηλεσκόπιο και ο θόλος. Το επιστημονικό ίδρυμα που διαθέτει τα παραπάνω χαρακτηριστικά ονομάζεται αστεροσκοπείο [15].

Το πρώτο σύγχρονο αστεροσκοπείο στην Ευρώπη θεωρείται το αστεροσκοπείο της Νυρεμβέργης που ιδρύθηκε το 1472.

2.1.1 Αστεροσκοπείο Υπάτης

Το «Κακογιάννειο Αστεροσχολείο» στεγάζεται στο παλιό κτίριο του Δημοτικού Σχολείου Υπάτης που παραχωρήθηκε από τον πρώην Δήμο Υπάτης (νυν Δήμο Λαμιέων) του οποίου ο εξοπλισμός χρηματοδοτήθηκε με δωρεά του ζεύγους Υπαταίων, Σταύρου & Ειρήνης Κακογιάννη, το 2008, προς τον Σύλλογο Ερασιτεχνών Αστρονόμων Φθιώτιδας «Ο Ηρακλής».



Εικόνα 4 - Αστεροσκοπείο της Υπάτης

Αποτελεί κέντρο εκπαίδευσης, έρευνας και διάδοσης της επιστήμης της Αστρονομίας για τους μαθητές και κάθε ενδιαφερόμενο περί την επιστήμη της Αστρονομίας, είναι δε μια καινοτόμος εκπαιδευτική, ερευνητική και πολιτιστική δραστηριότητα, με την αμέριστη υποστήριξη του Δήμου Λαμιέων [17].

2.2 Τηλεσκόπιο



Εικόνα 5 - Τηλεσκόπιο Υάτης

Το τηλεσκόπιο είναι ένα όργανο σχεδιασμένο για την παρατήρηση μακρινών αντικειμένων [16] μέσω της συλλογής ηλεκτρομαγνητικής ακτινοβολίας. Τα πρώτα γνωστά σχεδόν λειτουργικά τηλεσκόπια ανακαλύφθηκαν στις Κάτω Χώρες στις αρχές του 17ου αιώνα. Ο όρος «τηλεσκόπια» μπορεί να αναφέρεται σε ένα ευρύ φάσμα οργάνων που λειτουργούν στις περισσότερες περιοχές του ηλεκτρομαγνητικού φάσματος.

Τα πρώτα τηλεσκόπια χρησιμοποιούσαν φακούς για να συλλέξουν το φως από ένα μακρινό αντικείμενο. Το φως κατόπιν εστιάζονταν για να σχηματίσει το είδωλο του αντικειμένου, ενώ ένας άλλος φακός μεγέθυνε το είδωλο για τον αστρονόμο. Το αποτέλεσμα ήταν τα αμυδρά μακρινά αντικείμενα να γίνονται μεγαλύτερα, λαμπρότερα και καθαρότερα.

Το τηλεσκόπιο που χρησιμοποιεί φακό για να συγκεντρώσει το φως ονομάζεται διοπτρικό ή διαθλαστικό. Το πρώτο κατοπτρικό τηλεσκόπιο σχεδιάστηκε το 1668 από τον Ισαάκ Νεύτωνα. Και οι δύο τύποι τηλεσκοπίου χρησιμοποιούνταν εξίσου μέχρι τις αρχές του 20ού αιώνα, όταν το ανακλαστικό τελικά επικράτησε. Στα τέλη του 20ού αιώνα, οι αστρονόμοι χρειάζονταν πιο ισχυρά τηλεσκόπια όσον αφορά στη συγκέντρωση φωτός, ισχύ που δεν τους πρόσφερε το ένα και μοναδικό κάτοπτρο. Έτσι, πολλά από τα πιο σύγχρονα τηλεσκόπια χρησιμοποιούν περισσότερα κάτοπτρα για να συγκεντρώσουν το φως. Τα δύο τηλεσκόπια Κεκ στη Χαβάη, για παράδειγμα, έχουν το

καθένα 36 κάτοπτρα ενωμένα μαζί που σχηματίζουν ένα μεγάλο κάτοπτρο με πλάτος 10 μέτρα. Σήμερα, τα περισσότερα τηλεσκόπια χρησιμοποιούν κάτοπτρα και ονομάζονται κατοπτρικά ή ανακλαστικά.

Η βάση του τηλεσκοπίου είναι μια μηχανική κατασκευή-διάταξη που στηρίζει κατά ένα ιδιαίτερο τρόπο το τηλεσκόπιο, έτσι ώστε να εξυπηρετεί τις λειτουργίες του τηλεσκοπίου. Κύρια λειτουργία της βάσης στήριξης, είναι η ακριβής στόχευση του κυρίου σώματος του τηλεσκοπίου.

2.3 Θόλος



Εικόνα 6 - Θόλος Αστεροσκοπείου

Ο θόλος του αστεροσκοπείου είναι μια κατασκευή σκοπός της οποίας είναι η προστασία του εξοπλισμού και ταυτόχρονα επιτρέπει την απρόσκοπτη παρατήρηση του ουρανού και τη διέλευση του φωτός μόνο από την θέση παρατήρησης, ενώ ταυτόχρονα προσφέρει φυσική προστασία από τις περιβαλλοντικές συνθήκες. Έχει ημισφαιρικό ή κυλινδρικό σχήμα το οποίο βοηθά στη βελτιστοποίηση της αεροδυναμικής και στη μείωση του επηρεασμού του από τον άνεμο, εξασφαλίζοντας σταθερότητα. Κατασκευάζεται από υλικά όπως αλουμίνιο, ανοξείδωτο χάλυβα ή σύνθετα υλικά τα οποία είναι ανθεκτικά στις καιρικές συνθήκες και ελαφριά, επιτρέποντας την ομαλή περιστροφή του. Οι σύγχρονοι θόλοι είναι εξοπλισμένοι με συστήματα αυτοματισμού που επιτρέπουν απομακρυσμένο έλεγχο περιστροφής, αυτόματο άνοιγμα και κλείσιμο της θυρίδας, καθώς και συγχρονισμό με την κίνηση του τηλεσκοπίου.

Διαθέτει μια θυρίδα στραμμένη προς τον ουρανό και η οποία είναι το μοναδικό άνοιγμα θέασης του τηλεσκοπίου ώστε το μοναδικό φως που εισέρχεται εντός του θόλου να προέρχεται από την περιοχή παρατήρησης. Προκειμένου να μπορεί να παρατηρηθεί οποιοδήποτε ουράνιο σώμα πρέπει το άνοιγμα του θόλου να μπορεί να τοποθετηθεί σε

οποιαδήποτε θέση κινούμενο προς τον οριζόντιό του άξονα. Για να επιτευχθεί αυτό πρέπει να είναι δυνατή η περιστροφή του θόλου (και μαζί του η θυρίδα) κατά 360 μοίρες, επιτρέποντας στο τηλεσκόπιο τη θέαση οποιασδήποτε περιοχής του ουρανού. Η κίνηση επιτυγχάνεται μέσω οδοντωτών γραναζιών και συνήθως υποβοηθείται από ηλεκτροκινητήρες. Ο θόλος τοποθετείται επάνω σε ένα πύργο κατάλληλων διαστάσεων ο οποίος διαθέτει και μια πόρτα εισόδου.



Εικόνα 7 - Θόλος τοποθετημένος σε πύργο

Υπάρχουν διαφόρων ειδών θόλοι αστεροσκοπείου. Όταν πρόκειται να επιλέξουμε έναν κατάλληλο θόλο το πρώτο χαρακτηριστικό που μας ενδιαφέρει είναι η διάμετρός του η οποία συνήθως είναι μεγαλύτερη από τα δυο μέτρα. Άλλα χαρακτηριστικά είναι ο τρόπος περιστροφής του θόλου, ο τρόπος ανοίγματος της θυρίδας, δυνατότητες απομακρυσμένου ελέγχου, κ.α.

2.3.1 Σύστημα περιστροφής του θόλου

Ο απλούστερος είναι ο χειροκίνητος όπου η περιστροφή του θόλου και το άνοιγμα της θυρίδας γίνεται χειροκίνητα.

Ένας καλύτερος τρόπος ελέγχου της περιστροφής και του ελέγχου της θυρίδας είναι μέσω ηλεκτροκινητήρων, ένας για την περιστροφή του θόλου και ένας για τη θυρίδα οι οποίοι τροφοδοτούνται από μια μονάδα ελέγχου. Στην περίπτωση αυτή ο θόλος διαθέτει και μια μονάδα ελέγχου των ηλεκτροκινητήρων μέσω διακοπών και ηλεκτρονόμων.

Μπορούμε να αυτοματοποιήσουμε τον τρόπο ελέγχου των κινητήρων με τη βοήθεια ηλεκτρονικών πλακετών επέκτασης.

Επίσης, μέσω της δυνατότητας που μας δίνει η τεχνολογία, μπορούμε να χειριστούμε τους κινητήρες απομακρυσμένα από οπουδήποτε στον κόσμο μέσω διασύνδεσης IoT.

2.3.2 Παρακολούθηση της θέσης του θόλου

Η παρακολούθηση της θέσης ενός περιστροφικού στοιχείου, όπως ο θόλος ενός αστεροσκοπείου, γίνεται μέσω αισθητήρων που ανιχνεύουν και καταγράφουν τη γωνιακή του θέση με ακρίβεια. Οι αισθητήρες αυτοί χωρίζονται σε διάφορες κατηγορίες, ανάλογα με την τεχνολογία που χρησιμοποιούν και την ακρίβεια που προσφέρουν. Συνήθως χρησιμοποιούνται διακόπτες ορίων και αισθητήρας αρχικής θέσης.



Εικόνα 8 - Διακόπτης ορίων κίνησης (Limit Switch)

Οι διακόπτες ορίων είναι διακόπτες τοποθετημένοι σε προκαθορισμένες θέσεις. Κάθε φορά που ενεργοποιείται ένας διακόπτης και δεδομένου της φοράς περιστροφής μπορούμε να κωδικοποιήσουμε τη θέση του θόλου με ακρίβεια όταν αυτή βρίσκεται σε σημείο επαφής.



Εικόνα 9 - Αισθητήρας θέσης (Home Sensor)

Ο αισθητήρας αρχικής θέσης προορίζεται για τη δημιουργία μιας θέσης αναφοράς. Η θέση αυτή συνήθως είναι και η τρέχουσα θέση του τηλεσκοπίου ώστε στη συνέχεια να συγχρονιστεί η θέση της θυρίδας με αυτό.

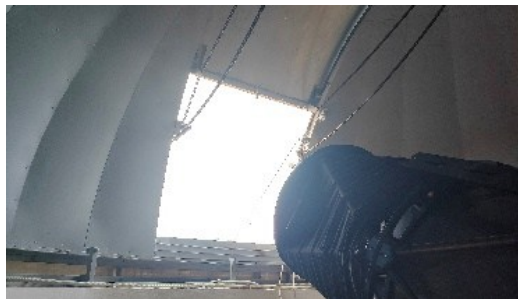


Εικόνα 10 - Κωδικοποιητής (Encoder)

Ένας περισσότερο ακριβής προσδιορισμός της θέσης είναι μέσω κωδικοποιητή στον κινητήρα ο οποίος παρακολουθεί την περιστροφή του κινητήρα και έτσι μπορούμε να υπολογίσουμε τη γωνία περιστροφής σε σχέση με την αρχική θέση.

2.3.3 Έλεγχος της θυρίδας

Ο έλεγχος του ανοίγματος και κλεισίματος της θυρίδας γίνεται μέσω ενεργοποιητών (actuators) οι οποίοι προκαλούν την ενεργοποίηση του κινητήρα της θυρίδας. Τοποθετούνται επίσης και τερματικοί διακόπτες ώστε το σύστημα να γνωρίζει για το πλήρες άνοιγμα/κλείσιμο της θυρίδας.



Εικόνα 11 - Θυρίδα

2.3.4 Θόλος Αστεροσκοπείου Υπάτης

Κατασκευάστηκε από την «Εργαλειοτεχνική Κρήτης» και έχει τη δυνατότητα να περιστρέφεται ηλεκτρικά 360 μοίρες με τη βοήθεια διακοπών οι οποίοι ελέγχουν τη φορά περιστροφής.

Ηλεκτρικά γίνεται επίσης και το άνοιγμα κλείσιμο της θυρίδας.



Εικόνα 12 - Θόλος Αστεροσκοπείου της Υπάτης

2.4 Ψηφιακά συστήματα σύνδεσης, συλλογής, αποθήκευσης και μετάδοσης μετρήσεων

Η λειτουργία του θόλου ενός αστεροσκοπείου απαιτεί υψηλό βαθμό ακρίβειας και συγχρονισμού με το τηλεσκόπιο. Τα συστήματα αυτοματισμού και ελέγχου είναι απαραίτητα για την επίτευξη αυτών των στόχων, ενώ διευκολύνουν και την απομακρυσμένη διαχείριση του εξοπλισμού.

Η σύνδεση αισθητήρων με ψηφιακά συστήματα για τη συλλογή, αποθήκευση και μετάδοση μετρήσεων είναι μια κρίσιμη διαδικασία στα απομακρυσμένα συστήματα ελέγχου, όπως αυτό για το θόλο ενός αστεροσκοπείου. Οι αισθητήρες, που παρακολουθούν παραμέτρους όπως η θέση ή η γωνιακή κίνηση του θόλου, μετατρέπουν τις φυσικές μετρήσεις σε ψηφιακά σήματα που μπορούν να διαβαστούν από μικροελεγκτές.

Αρχικά, οι αισθητήρες συνδέονται με μικροελεγκτές (π.χ., Arduino, Raspberry Pi, ESP32), οι οποίοι διαβάζουν τις μετρήσεις και τις επεξεργάζονται σε πραγματικό χρόνο. Οι μικροελεγκτές μπορούν να αποθηκεύουν αυτά τα δεδομένα τοπικά ή να τα μεταδίδουν σε μια gateway, που λειτουργεί ως κεντρικός κόμβος επικοινωνίας. Η gateway συγκεντρώνει τις μετρήσεις και διασφαλίζει ότι οι πληροφορίες είναι άμεσα διαθέσιμες στο χρήστη.

Για τη μετάδοση των δεδομένων από τη gateway προς απομακρυσμένες συσκευές, χρησιμοποιούνται πρωτόκολλα επικοινωνίας, όπως το MQTT και το HTTP, τα οποία επιτρέπουν τη γρήγορη και ασφαλή αποστολή δεδομένων μέσω του διαδικτύου. Η χρήση αυτών των πρωτοκόλλων βοηθά στην αξιόπιστη μεταφορά των πληροφοριών σε πραγματικό χρόνο, επιτρέποντας την απομακρυσμένη ρύθμιση του θόλου του αστεροσκοπείου.

Τα ψηφιακά συστήματα τα οποία προορίζονται για τη σύνδεση αισθητήρων για τη συλλογή, επεξεργασία, αποθήκευση και μετάδοση των μετρήσεων, για λόγους ευελιξίας και επέκτασης συνήθως είναι συμβατά με το περιβάλλον Arduino, επικοινωνούν με πρωτόκολλα ανοιχτού κώδικα, και ελέγχονται μέσω θύρας Ethernet ή USB. Χωρίζονται σε δυο μονάδες, μια για τον έλεγχο της περιστροφής του θόλου και μια για τον έλεγχο της θύρας του θόλου.

Διαθέτουν λογικές εισόδους για σύνδεση των αισθητήρων, εξόδους με relay για τον έλεγχο των κινητήρων, κάποια ενσωματώνουν mini web server, και διαθέτουν προγράμματα

οδήγησης για Windows. Ανάλογα με το μοντέλο μπορούν επίσης να διαθέτουν PWM εξόδους, touch screen, εισόδους SPI, OneWire, θερμόμετρο, Bluetooth και άλλα.

Για τον έλεγχο του θόλου χρησιμοποιείται συνήθως το πρότυπο ASCOM [5] το οποίο εξασφαλίζει τη συμβατότητα μεταξύ του λογισμικού του θόλου και άλλων συστημάτων, όπως το τηλεσκόπιο.



Εικόνα 13 - Πλακέτα ελέγχου του εμπορίου

2.5 Αυτοματισμοί ελεγχόμενων από απόσταση (τηλεμετρία) για τηλεσκόπια

Οι αυτοματισμοί με τηλεμετρία για τηλεσκόπια επιτρέπουν την απομακρυσμένη διαχείριση και παρακολούθηση της θέσης και των λειτουργιών τους. Μέσω τηλεμετρικών συστημάτων, οι χρήστες μπορούν να ελέγχουν το τηλεσκόπιο από απόσταση, ρυθμίζοντας τη θέση του, την κίνηση του θόλου και την εστίαση του φακού, χωρίς να βρίσκονται στο ίδιο μέρος με τον εξοπλισμό.

Τα συστήματα τηλεμετρίας περιλαμβάνουν αισθητήρες θέσης, ενεργοποιητές και μικροελεγκτές ή υπολογιστές για την ακριβή καταγραφή και διαχείριση των δεδομένων. Η επικοινωνία μεταξύ του τηλεσκοπίου και της απομακρυσμένης συσκευής ελέγχου γίνεται μέσω διαδικτυακών πρωτοκόλλων, όπως το MQTT ή το HTTP, τα οποία επιτρέπουν την ανταλλαγή δεδομένων σε πραγματικό χρόνο. Με αυτό τον τρόπο, είναι δυνατή η λήψη πληροφοριών για την τρέχουσα θέση του τηλεσκοπίου και η αποστολή εντολών για την προσαρμογή του σύμφωνα με τις ανάγκες του χρήστη. Όσον αφορά το λογισμικό ελέγχου του θόλου αυτό συνήθως χρησιμοποιεί τα πρότυπα ASCOM [5] ή INDI [6],[2].

Η απομακρυσμένη τηλεμετρία είναι πολύτιμη για επαγγελματικά και ερασιτεχνικά αστεροσκοπεία, καθώς μειώνει την ανάγκη φυσικής παρουσίας στον χώρο παρατήρησης, προσφέροντας παράλληλα ακριβή και άμεσο έλεγχο του εξοπλισμού. Επιτρέπει επίσης τη συνεργασία μεταξύ αστεροσκοπείων και επιστημόνων σε διαφορετικά μέρη, ενισχύοντας την αποδοτικότητα των παρατηρήσεων και των ερευνών.

2.5.1 Πλεονεκτήματα των αυτοματισμών

1. **Ασφάλεια:** Άμεση αντίδραση σε περιβαλλοντικές αλλαγές, μειώνοντας τον κίνδυνο φθοράς του εξοπλισμού.
2. **Ακρίβεια:** Εξασφαλίζεται η απόλυτη ευθυγράμμιση του θόλου με το τηλεσκόπιο σε οποιαδήποτε φάση της παρατήρησης.
3. **Αποδοτικότητα:** Μειώνονται οι ανάγκες για χειροκίνητη παρέμβαση, εξοικονομώντας χρόνο και κόπο.
4. **Υποστήριξη Απομακρυσμένης Παρατήρησης:** Διευκολύνεται η λειτουργία του αστεροσκοπείου από απομακρυσμένες τοποθεσίες.

3. Απαιτήσεις Συστήματος, Προδιαγραφές, Πλαίσιο Λειτουργίας

Το πρώτο βήμα στο σχεδιασμό ενός συστήματος είναι η καταγραφή των απαιτήσεων, δηλαδή των προδιαγραφών του συστήματος. Κατά την καταγραφή πρέπει να δοθεί πολύ μεγάλη προσοχή γιατί εδώ ορίζεται με σαφήνεια το τι θα κάνει το σύστημα. Στην περίπτωση που έχει ξεκινήσει ο σχεδιασμός και διαπιστωθεί παράλειψη ή λάθος στις απαιτήσεις, είναι πολύ δύσκολη και υψηλού κόστους η τροποποίησή τους. Για αυτό κατά τη διάρκεια της καταγραφής των απαιτήσεων πρέπει να συμβάλουν όλα τα εμπλεκόμενα μέρη ώστε να είναι όσο το δυνατόν εξασφαλισμένη η πληρότητά τους.

Οι *μη λειτουργικές απαιτήσεις* περιγράφουν τις ιδιότητες ή τα χαρακτηριστικά που πρέπει να έχει ένα σύστημα, αλλά δεν σχετίζονται άμεσα με το τι κάνει το σύστημα (τη λειτουργικότητά του), αλλά με το πώς το κάνει. Εστιάζουν στην ποιότητα της λειτουργίας του συστήματος και διασφαλίζουν ότι το σύστημα πληροί συγκεκριμένα πρότυπα. Για παράδειγμα ορίζουν το χρόνο απόκρισης του συστήματος, την αξιοπιστία του, θέματα ασφαλείας, και άλλα. Είναι σημαντικές γιατί καθορίζουν το επίπεδο ποιότητας του συστήματος. Ακόμη κι αν ένα σύστημα μπορεί να εκτελέσει τις βασικές λειτουργίες του (π.χ., να περιστρέφει έναν θόλο), αν αυτές οι λειτουργίες δεν είναι γρήγορες, αξιόπιστες και ασφαλείς, το σύστημα δεν θα θεωρηθεί επιτυχημένο. Με απλά λόγια, οι *μη λειτουργικές απαιτήσεις* καθορίζουν το πόσο καλά δουλεύει ένα σύστημα, όχι απλά τι κάνει.

Οι *λειτουργικές απαιτήσεις* περιγράφουν το τι πρέπει να κάνει ένα σύστημα. Εστιάζουν στις βασικές λειτουργίες και δυνατότητες που πρέπει να έχει το σύστημα ώστε να εκπληρώσει τον σκοπό του. Είναι μια λίστα με τις ενέργειες που πρέπει να μπορεί να εκτελέσει το σύστημα. Είναι σημαντικές γιατί ορίζουν τι ακριβώς πρέπει να κάνει το σύστημα ώστε να εκπληρώσει τον στόχο του. Με απλά λόγια, οι *λειτουργικές απαιτήσεις* απαντούν στην ερώτηση «τι κάνει το σύστημα;»

3.1 Μη Λειτουργικές Απαιτήσεις

3.1.1 Αξιοπιστία

- Το σύστημα πρέπει να μπορεί να λειτουργεί απρόσκοπτα και αδιαλείπτως.

3.1.2 Απόκριση και Ταχύτητα εκτέλεσης

- Ο χρόνος απόκρισης για το άνοιγμα/κλείσιμο της θυρίδας δεν πρέπει να ξεπερνά τα 30 δευτερόλεπτα.
- Η απομακρυσμένη εντολή για την περιστροφή του θόλου πρέπει να αρχίζει να εκτελείται μέσα σε διάστημα 5 δευτερολέπτων.

3.1.3 Ασφάλεια

- Το σύστημα πρέπει να διαθέτει μηχανισμούς ανίχνευσης και διόρθωσης λαθών για την αποφυγή παρεμβολών και λανθασμένων εντολών.
- Θα πρέπει να είναι εξασφαλισμένη η λειτουργία μόνο από εξουσιοδοτημένα άτομα.

3.1.4 Επεκτασιμότητα / Upgradability

- Δυνατότητα μελλοντικής επέκτασης του συστήματος.
- Πρέπει να είναι δυνατή η αναβάθμιση λογισμικού με ή χωρίς διακοπή της λειτουργίας.

3.1.5 Ευχρηστία

- Το περιβάλλον χειρισμού πρέπει να είναι φιλικό προς τον χρήστη.
- Δυνατότητα απομακρυσμένου ελέγχου.

3.1.6 Συντήρηση

1. Το σύστημα πρέπει να απαιτεί ελάχιστη συντήρηση.

3.2 Λειτουργικές Απαιτήσεις

3.2.1 Βασικός Έλεγχος Θόλου

- Ο θόλος θα πρέπει να μπορεί να περιστρέφεται και προς τις δυο κατευθύνσεις.
- Θα πρέπει να υπάρχει η δυνατότητα ελέγχου της περιστροφής του θόλου τόσο τοπικά όσο και απομακρυσμένα.
- Ο τοπικός χειρισμός της περιστροφής θα πραγματοποιείται μέσω κατάλληλων πλήκτρων, ενώ ο απομακρυσμένος θα πραγματοποιείται μέσω εικονικών πλήκτρων διεπαφής, ένα για κάθε κατεύθυνση. Με το πάτημα του πλήκτρου θα ξεκινάει η κίνηση και με την απελευθέρωσή του θα σταματάει. Θα σταματάει επίσης με αυτόματο τρόπο στην περίπτωση που κατά την περιστροφή η θέση του θόλου βρεθεί στην αρχική του ή ταυτιστεί με αυτήν του τηλεσκοπίου.
- Στον απομακρυσμένο χειρισμό, η διεπαφή θα πρέπει να περιλαμβάνει την τρέχουσα θέση του θόλου.

3.2.2 Αυτόματος συγχρονισμός θέσης θόλου με το τηλεσκόπιο

Θα πρέπει να μπορεί ο χρήστης να ενεργοποιεί αυτόματη λειτουργία κατά την οποία ο θόλος θα ακολουθεί και θα ευθυγραμμίζεται με το τηλεσκόπιο.

3.2.3 Έλεγχος Θυρίδας

- Το σύστημα πρέπει να επιτρέπει το άνοιγμα και το κλείσιμο του θυρίδας τόσο τοπικά μέσω κατάλληλων πλήκτρων όσο και απομακρυσμένα.
- Η θυρίδα θα πρέπει να διαθέτει 2 τερματικούς διακόπτες με τους οποίους θα διακρίνονται οι καταστάσεις κλειστή, ανοιχτή, τέρμα ανοιχτή. Η διαφορά της ανοιχτής από την τέρμα ανοιχτή είναι ότι ανοιχτή θυρίδα σημαίνει όχι κλειστή, ενώ τέρμα ανοιχτή θυρίδα σημαίνει ότι είναι στην τερματική θέση ανοίγματος.
- Ο έλεγχος της θυρίδας θα πρέπει να μπορεί να την τοποθετεί σε οποιαδήποτε θέση ανοίγματος εντός των ορίων κλειστής-τέρμα ανοιχτής.
- Στον απομακρυσμένο χειρισμό, η διεπαφή θα πρέπει να περιλαμβάνει την τρέχουσα θέση της θυρίδας (κλειστή, ανοιχτή, τέρμα ανοιχτή).

3.2.4 Επιστροφή του θόλου στην αρχική του θέση

- Το σημείο ηρεμίας (αρχική θέση) του θόλου θα πρέπει να είναι ευθυγραμμισμένο με την αρχική θέση του τηλεσκοπίου.
- Με το πέρας της παρατήρησης, θα κλείνει η θυρίδα και ο θόλος θα επιστρέφει στην αρχική του θέση.

3.2.5 Λειτουργίες ασφαλείας

- Σε περίπτωση βροχής, η θυρίδα θα κλείνει με αυτόματο τρόπο και θα ενημερώνεται ο χρήστης μέσω του συστήματος.
- Σε περίπτωση διακοπής της επικοινωνίας ή αδράνειας για κάποιο ορισμένο χρόνο, θα πρέπει να επιστρέφει ο θόλος στην αρχική του θέση και η θυρίδα να κλείνει.
- Σε περίπτωση που ο χρόνος περιστροφής υπερβεί ένα μέγιστο ή ολοκληρωθεί ένας πλήρης κύκλος, θα πρέπει να γίνεται αυτόματη διακοπή της περιστροφής και ενημέρωση του χρήστη.
- Σε περίπτωση που κατά την περιστροφή δεν μεταβάλλεται η ένδειξη του encoder, θα πρέπει να γίνεται αυτόματη διακοπή της περιστροφής και ενημέρωση του χρήστη.
- Θα πρέπει να υπάρχει ένας μέγιστος χρόνος αναμονής του σήματος θυρίδα τέρμα ανοιχτή/θυρίδα κλειστή κατά την εκτέλεση της αντίστοιχης εντολής ώστε αν η θυρίδα κινείται για μεγαλύτερο χρόνο, τότε η εντολή θα ακυρώνεται.

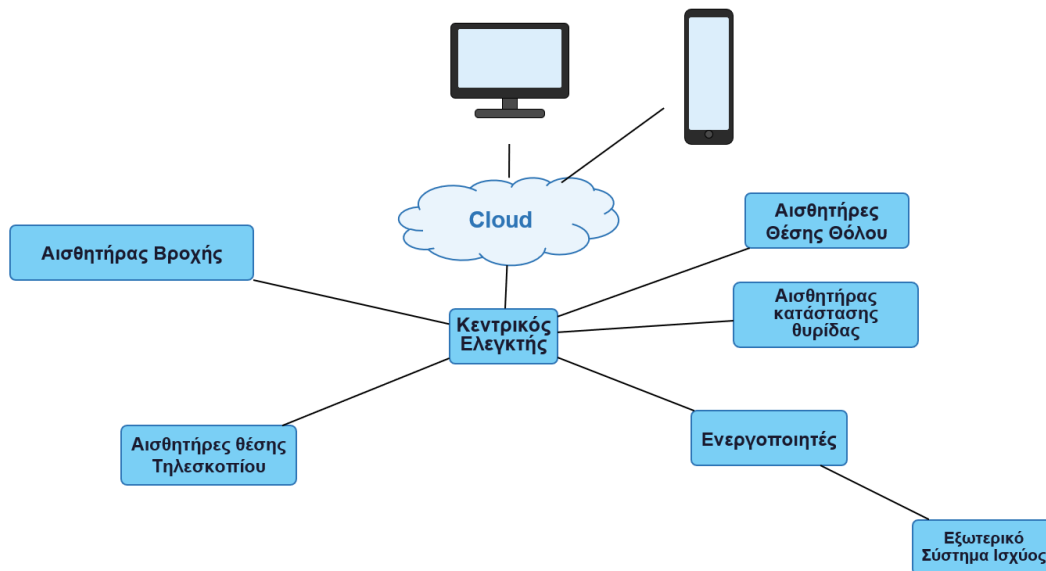
3.3 Πλαίσιο Λειτουργίας

- **Νομοθετικές ρυθμίσεις:** Δεδομένου της υφιστάμενης κατασκευής και λειτουργίας του αστεροσκοπείου της Υπάτης, θεωρείται δεδομένη η συμμόρφωση με το νομικό πλαίσιο λειτουργίας όπως καθορίζεται στις πολεοδομικές και κατασκευαστικές διατάξεις που περιλαμβάνουν και τις ρυθμίσεις για την προστασία του περιβάλλοντος. Δεδομένη επίσης θεωρείται η τήρηση της εργασιακής νομοθεσίας για την ασφάλεια των εργαζομένων καθώς επίσης και με τις ειδικές ρυθμίσεις για τα ερευνητικά κέντρα. Είναι σημαντικό να σημειωθεί ότι πέρα από τις γενικές διατάξεις, ενδέχεται να υπάρχουν ειδικές ρυθμίσεις ανάλογα με τη φύση των δραστηριοτήτων του αστεροσκοπείου, την τοποθεσία του και τις τεχνολογίες που χρησιμοποιεί. Για την πλήρη συμμόρφωση με το νομοθετικό πλαίσιο, συνιστάται η συνεργασία με νομικούς συμβούλους και τις αρμόδιες αρχές, ώστε να διασφαλιστεί η ορθή λειτουργία του θόλου του αστεροσκοπείου σύμφωνα με την ισχύουσα νομοθεσία.
- **Διεθνή Πρότυπα:**
 - **IEC 61508** Functional Safety of Electrical/Electronic Systems: Καθορίζει τις απαιτήσεις για την ασφάλεια και τη λειτουργικότητα των ηλεκτρονικών και αυτόματων συστημάτων.
 - **ISO 20474** Earth-moving machinery — Safety: Το συγκεκριμένο πρότυπο βρίσκει εφαρμογή στην ασφάλεια των κινητών μηχανισμών του θόλου, όπως είναι τα συστήματα περιστροφής.
 - **ISO 50001** Energy Management Systems: Παρέχει οδηγίες για τη βέλτιστη χρήση ενέργειας και τη διαχείριση του συστήματος ισχύος του θόλου.
 - **ISO 31000** Risk Management: Ασχολείται με τη διαχείριση κινδύνων, όπως οι επιπτώσεις ακραίων καιρικών φαινομένων στον θόλο.

4. Σχεδιασμός Συστήματος

4.1 Τοπολογία προτεινόμενου Συστήματος

Η προτεινόμενη λύση στοχεύει στη δημιουργία ενός ολοκληρωμένου συστήματος απομακρυσμένου ελέγχου και αυτοματισμού για τον θόλο του αστεροσκοπείου Υπάτης. Το σύστημα συνδυάζει σύγχρονες τεχνολογίες IoT (Internet of Things), μηχανισμούς αυτοματισμού και συστήματα ασφαλούς επικοινωνίας για να εξασφαλίσει την εύρυθμη λειτουργία και την αποδοτικότητα.



Εικόνα 14 - Τοπολογία του Συστήματος

Η τοπολογία του συστήματος είναι σχεδιασμένη έτσι ώστε να επιτρέπει την αποδοτική και αξιόπιστη επικοινωνία μεταξύ των επιμέρους στοιχείων. Περιλαμβάνει:

- **Κεντρικό Ελεγκτή:** Βρίσκεται στο κέντρο της τοπολογίας και διαχειρίζεται τη συνολική λειτουργία του θόλου.
- **Αισθητήρες Θέσης Θόλου:** Ελέγχουν την θέση και την ευθυγράμμιση του θόλου με το τηλεσκόπιο.
- **Αισθητήρες Θέσης Τηλεσκοπίου:** Ελέγχουν τη θέση του τηλεσκοπίου.
- **Αισθητήρα Βροχής:** Συλλέγει δεδομένα καιρικών συνθηκών για βροχή.
- **Αισθητήρες κατάστασης θυρίδας:** Ελέγχουν την κατάσταση της θυρίδας (κλειστή, ανοιχτή, τέρμα ανοιχτή).

- **Ενεργοποιητές:** Χρησιμοποιούνται για την περιστροφή του θόλου και το άνοιγμα/κλείσιμο της θυρίδας.
- **Δίκτυο επικοινωνίας:** Χρησιμοποιεί το Internet of Things για την απομακρυσμένη επικοινωνία.
- **Απομακρυσμένοι κόμβοι ελέγχου:** Εξασφαλίζουν την αλληλεπίδραση του συστήματος με τον χρήστη μέσω web interface H/Y ή κινητών συσκευών.
- **Εξωτερικό Σύστημα Ισχύος:** Το εγκατεστημένο σύστημα ισχύος στην Υπάτη το οποίο χειρίζεται το θόλο και την θυρίδα του.

4.2 Λειτουργία των στοιχείων του συστήματος

- Ο *αισθητήρας βροχής* παρακολουθεί συνεχώς σε πραγματικό χρόνο την κατάσταση του καιρού για βροχή και ενημερώνει τον ελεγκτή.
- Οι *αισθητήρες θέσης τηλεσκοπίου* παρακολουθούν και ενημερώνουν τον ελεγκτή για την τρέχουσα θέση του τηλεσκοπίου η οποία μπορεί να είναι είτε στην αρχική του θέση, είτε σε κάποια γωνία σε σχέση με την αρχική του θέση.
- Οι *αισθητήρες θέσης θόλου* παρακολουθούν και ενημερώνουν τον ελεγκτή για την τρέχουσα θέση του θόλου η οποία μπορεί να είναι είτε στην αρχική του θέση, είτε σε κάποια γωνία σε σχέση με την αρχική του θέση.
- Οι *αισθητήρες κατάστασης θυρίδας* ελέγχουν την κατάσταση της θυρίδας (κλειστή, ανοιχτή, τέρμα ανοιχτή).
- Οι *ενεργοποιητές* δέχονται τις εντολές από τον κεντρικό ελεγκτή και στέλνουν τα κατάλληλα σήματα στη μονάδα ισχύος του θόλου.
- Η *μονάδα ισχύος* του θόλου μετατρέπει τα σήματα ελέγχου σε κίνηση του θόλου και της θυρίδας. Η συγκεκριμένη μονάδα είναι εξωτερική του συστήματος που αναπτύσσεται.
- Στο *δίκτυο* περιλαμβάνεται όλος ο εξοπλισμός ο οποίος είναι απαραίτητος για τη διασύνδεση του συστήματος με τον έξω κόσμο.
- Κεντρικό ρόλο στη λειτουργία του συστήματος, παίζει ο *κεντρικός ελεγκτής* ο οποίος ελέγχει και συγχρονίζει όλες τις λειτουργίες του θόλου. Συγκεκριμένα, αφού αξιολογήσει τις καιρικές συνθήκες που διαβάζει από τον αισθητήρα βροχής ως κατάλληλες, επιτρέπει τη λειτουργία του θόλου μέσω των εντολών του χρήστη.

Επειδή ο κεντρικός ελεγκτής συνδέεται με το διαδίκτυο είναι απαραίτητο να διαθέτει έλεγχο πρόσβασης.

Αφού δοθεί η πρόσβαση σε εξουσιοδοτημένο χρήστη και το επιτρέπουν οι καιρικές συνθήκες, εισέρχεται σε κατάσταση αναμονής εντολής. Οι διαθέσιμες απομακρυσμένες εντολές είναι *επιστροφή στην αρχική θέση, περιστροφή δεξιά, περιστροφή αριστερά, συγχρονισμός με τη θέση του τηλεσκόπιου, άνοιγμα θυρίδας, κλείσιμο θυρίδας*. Η κάθε εντολή ενεργεί στον κατάλληλο ενεργοποιητή. Κατά την περιστροφή παρακολουθείται διαρκώς η θέση του θόλου σε σχέση με την αρχική του θέση και εμφανίζεται στην οθόνη του χρήστη. Αν κατά την περιστροφή βρεθεί στην αρχική του θέση, η περιστροφή θα διακόπτεται και θα απαιτείται νέα εντολή. Σε περίπτωση που ενεργοποιηθεί ο αισθητήρας βροχής θα κλείνει αυτόματα η θυρίδα.

Οι λειτουργίες ασφαλείας που περιλαμβάνει έχουν να κάνουν με το πως θα ενεργήσει σε περιπτώσεις αστοχιών υλικού. Συγκεκριμένα η περιστροφή του θόλου ελέγχεται ως προς τον μέγιστο χρόνο περιστροφής και στην περίπτωση υπέρβασης διακόπτει την περιστροφή. Ο χρόνος ανοίγματος/κλεισίματος θυρίδας αν ξεπεράσει ένα μέγιστο όριο θα διακόπτει τη λειτουργία της.

4.3 Επικοινωνία των στοιχείων με τον κεντρικό ελεγκτή

Για την εξυπηρέτηση των σκοπών της πτυχιακής, η επικοινωνία των συστατικών στοιχείων με την κεντρική μονάδα στο μοντέλο θα είναι ενσύρματη και θα γίνεται είτε ψηφιακά είτε με τη χρήση ψηφιακών πρωτοκόλλων χαμηλής ταχύτητας. Όμως, όταν θα εφαρμοστεί η προτεινόμενη λύση στο θόλο του αστεροσκοπείου της Υπάτης, λόγω των αποστάσεων ενδέχεται να είναι απαραίτητη η χρήση ασύρματων αισθητήρων χρησιμοποιώντας κάποιο από τα διαθέσιμα πρωτόκολλα, καθώς επίσης και για λόγους αξιοπιστίας ίσως χρειαστεί και δεύτερη μονάδα ελέγχου.

4.4 Μορφή ενημέρωσης της κατάστασης του συστήματος

Η ενημέρωση για την κατάσταση του συστήματος πρέπει να είναι σχεδιασμένη ώστε να είναι κατανοητή, άμεση, και εύκολα προσβάσιμη από τον χρήστη. Η μορφή της ενημέρωσης θα πρέπει να περιλαμβάνει τα εξής στοιχεία (δεν υποστηρίζονται όλα από την παρούσα πτυχιακή, περιλαμβάνονται όμως εδώ για λόγους πληρότητας):

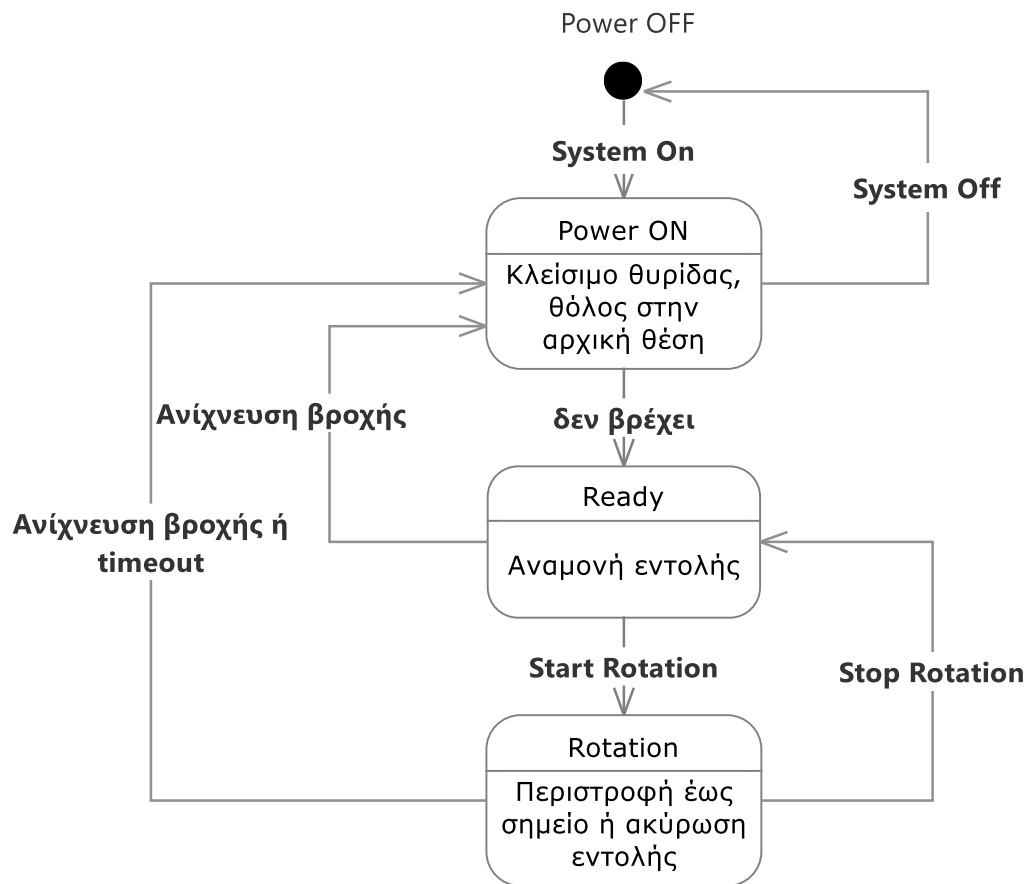
- *Διαδραστικός Πίνακας Ελέγχου (Dashboard)*: Προσφέρει μια συνολική εικόνα της κατάστασης του θόλου και των επιμέρους συστημάτων. Περιλαμβάνει γραφικές απεικονίσεις (π.χ. διαγράμματα, εικονίδια) που δείχνουν:
 1. Την κατάσταση του θόλου (αν κινείται ή όχι).
 2. Τη θέση του θόλου.
 3. Τη θέση του τηλεσκοπίου.
 4. Την κατάσταση της θυρίδας.
 5. Την κατάσταση των αισθητήρων.
 6. Καιρικές συνθήκες.
- Ο χρήστης μπορεί να λαμβάνει *πληροφορίες* σε πραγματικό χρόνο.
- Παρέχονται *ειδοποιήσεις* σε περιπτώσεις βλάβης ή κρίσιμων γεγονότων:

Σε περίπτωση αλλαγών ή προβλημάτων, ο χρήστης λαμβάνει άμεσες ειδοποιήσεις στην κινητή συσκευή του. Παράδειγμα ειδοποίησης: "Η θυρίδα έκλεισε λόγω βροχής."
- Καταγραφή ιστορικού: Δεδομένα από αισθητήρες, συνοπτική αναφορά γεγονότων (π.χ. ενεργοποίηση θόλου, καιρικές αλλαγές).

4.5 Περιπτώσεις Χρήσης (use cases)

Οι **περιπτώσεις χρήσης** είναι μια τυπική μέθοδος που χρησιμοποιείται στην ανάλυση και σχεδίαση συστημάτων για να περιγράψουν πώς οι χρήστες αλληλοεπιδρούν με το λογισμικό ή το σύστημα. Η κάθε μια είναι ένα σενάριο αλληλεπίδρασης ανάμεσα σε έναν χρήστη (ή άλλη οντότητα, που ονομάζεται «Actor») και το σύστημα, με σκοπό την επίτευξη ενός συγκεκριμένου στόχου ή επιθυμητού αποτελέσματος (π.χ. «ο χρήστης θέλει να ανοίξει τον θόλο απομακρυσμένα»).

Παρακάτω δίνεται ένα συνοπτικό διάγραμμα καταστάσεων στο οποίο γίνεται αναφορά στην περιγραφή των περιπτώσεων χρήσης που ακολουθεί.



Εικόνα 15 - Διάγραμμα καταστάσεων

i. Σενάρια Καλής Λειτουργίας

1. Αρχική κατάσταση έναρξης λειτουργίας

Περιγραφή: Το σύστημα περιστρέφει τον θόλο στην αρχική του θέση, και ελέγχεται η κατάσταση του καιρού για βροχή.

Actor: τοπικός χρήστης.

Διαδικασία:

1. Παροχή τροφοδοσίας στο σύστημα.

2. Η αρχική κατάσταση του θόλου είναι Power ON.
3. Κλείνει η θυρίδα αν είναι ανοικτή.
4. Ο θόλος μετακινείται στην αρχική του θέση.
5. Ελέγχεται ο αισθητήρας βροχής.
6. Αν δεν βρέχει, ο θόλος μπαίνει σε κατάσταση Ready, έτοιμος προς χρήση.
7. Αν βρέχει, ο θόλος παραμένει στην κατάσταση Power ON. Εάν στη συνέχεια σταματήσει η βροχή, η κατάσταση του θόλου αλλάζει σε Ready.

Αναμενόμενο αποτέλεσμα: Ο θόλος βρίσκεται στην αρχική του θέση, η θυρίδα είναι κλειστή, η κατάσταση του θόλου είναι Ready ή Power ON.

2. Σύνδεση Χρήστη

Περιγραφή: Ο εγγεγραμμένος χρήστης εισάγει διαπιστευτήρια για να αποκτήσει πρόσβαση στο σύστημα.

Actor: Χρήστης

Διαδικασία:

1. Ο χρήστης εισάγει όνομα χρήστη και κωδικό.
2. Το σύστημα επαληθεύει τα διαπιστευτήρια.
3. Το σύστημα εμφανίζει στον χρήστη την αρχική οθόνη ελέγχου του θόλου (dashboard).

Αναμενόμενο αποτέλεσμα: Ο χρήστης μπορεί να εκτελέσει λειτουργίες ελέγχου και παρακολούθησης του συστήματος.

3. Αποσύνδεση Χρήστη

Περιγραφή: Ο χρήστης αποσυνδέεται από το σύστημα και κλείνει τη συνεδρία του.

Actor: Χρήστης

Διαδικασία:

1. Ο χρήστης επιλέγει “Αποσύνδεση” στην πλατφόρμα ελέγχου.
2. Το σύστημα τερματίζει τη συνεδρία (session).
3. Η οθόνη επιστρέφει στη σελίδα εισόδου (login).

Αναμενόμενο αποτέλεσμα: Ο χρήστης δεν έχει πλέον πρόσβαση στο σύστημα.

4. Κλείσιμο θυρίδας λόγω καιρού

Περιγραφή: Η θυρίδα κλείνει αυτόματα όταν ανιχνευθεί βροχή.

Actor: Αισθητήρας Βροχής

Διαδικασία:

1. Ο αισθητήρας βροχής ανιχνεύει βροχή.
2. Ο κεντρικός ελεγκτής λαμβάνει την πληροφορία από τον αισθητήρα.
3. Ο κεντρικός ελεγκτής στέλνει εντολή κλεισίματος στον ενεργοποιητή της θυρίδας.
4. Ο ενεργοποιητής ενεργοποιεί την αντίστοιχη είσοδο της εξωτερικής μονάδα ισχύος και η θυρίδα αρχίζει να κλείνει.
5. Αν η κατάσταση του θόλου είναι Rotation , ο κεντρικός ελεγκτής ακυρώνει την τυχόν περιστροφή του θόλου.
6. Ο κεντρικός ελεγκτής αλλάζει την κατάσταση του θόλου σε Power ON.

Αναμενόμενο αποτέλεσμα: Η θυρίδα κλείνει για να προστατευτεί ο εξοπλισμός.

5. Αυτόματη περιστροφή θόλου

Περιγραφή: Το σύστημα περιστρέφει τον θόλο ώστε να ευθυγραμμιστεί με την κίνηση του τηλεσκοπίου.

Actor: Χρήστης.

Διαδικασία:

1. Ο χρήστης έχει ενεργοποιήσει την αυτόματη περιστροφή.
2. Ο encoder του τηλεσκοπίου παρακολουθεί τη θέση του τηλεσκοπίου.
3. Ο encoder του θόλου ανιχνεύει τη θέση του θόλου.
4. Ο Κεντρικός ελεγκτής συγκρίνει την θέση του θόλου με αυτήν του τηλεσκοπίου και σε περίπτωση απόκλισης στέλνει εντολή στους ενεργοποιητές περιστροφής του θόλου.
5. Ο Κεντρικός ελεγκτής αλλάζει την κατάσταση του θόλου σε Rotation.

6. Οι ενεργοποιητές περιστρέφουν το θόλο στην κατάλληλη θέση.

Αναμενόμενο αποτέλεσμα: Ο θόλος ευθυγραμίζεται με το τηλεσκόπιο.

6. Χειροκίνητη περιστροφή θόλου

Περιγραφή: Το σύστημα περιστρέφει τον θόλο κατόπιν εντολής του χρήστη.

Actor: Χρήστης.

Διαδικασία:

1. Ο χρήστης έχει δώσει εντολή περιστροφής.
2. Ο Κεντρικός ελεγκτής στέλνει εντολή στους ενεργοποιητές περιστροφής του θόλου στην επιθυμητή κατεύθυνση περιστροφής.
3. Ο Κεντρικός ελεγκτής αλλάζει την κατάσταση του θόλου σε Rotation.
4. Οι ενεργοποιητές περιστρέφουν το θόλο.
5. Ο χρήστης πληροφορείται από την οθόνη του για τη θέση του τηλεσκοπίου και του θόλου σε σχέση με την αρχική τους θέση.

Αναμενόμενο αποτέλεσμα: Ο θόλος περιστρέφεται για όσο χρόνο διαρκεί η εντολή του χρήστη.

7. Άνοιγμα θυρίδας

Περιγραφή: Το σύστημα ανοίγει τη θυρίδα μετά από εντολή του χρήστη.

Actor: Χρήστης.

Διαδικασία:

1. Ο χρήστης έχει δώσει εντολή ανοίγματος θυρίδας.
2. Ο Κεντρικός ελεγκτής ελέγχει την κατάσταση της θυρίδας.
3. Αν είναι ήδη ανοιχτή, επιστρέφει.
4. Διαφορετικά, στέλνει εντολή στον ενεργοποιητή ανοίγματος της θυρίδας.
5. Ο ενεργοποιητής ανοίγει τη θυρίδα.

Αναμενόμενο αποτέλεσμα: Ανοίγει η θυρίδα.

8. Κλείσιμο θυρίδας

Περιγραφή: Το σύστημα κλείνει τη θυρίδα μετά από εντολή του χρήστη.

Actor: Χρήστης.

Διαδικασία:

1. Ο χρήστης έχει δώσει εντολή κλεισίματος θυρίδας.
2. Ο Κεντρικός ελεγκτής ελέγχει την κατάσταση της θυρίδας.
3. Αν είναι ήδη κλειστή, επιστρέφει.
4. Διαφορετικά, στέλνει εντολή στον ενεργοποιητή κλεισίματος της θυρίδας.
5. Ο ενεργοποιητής κλείνει τη θυρίδα.

Αναμενόμενο αποτέλεσμα: Κλείνει η θυρίδα.

ii. Σενάρια Επικοινωνίας

1. Απομακρυσμένος Έλεγχος μέσω Εφαρμογής

Περιγραφή: Ο χρήστης ελέγχει τον θόλο και τη θυρίδα μέσω εφαρμογής από απομακρυσμένη τοποθεσία. Περιλαμβάνονται οι λειτουργίες «Αυτόματη και Χειροκίνητη περιστροφή θόλου» και «Χειροκίνητο άνοιγμα και κλείσιμο θυρίδας» των σεναρίων καλής λειτουργίας.

Actor: Χρήστης.

Διαδικασία:

1. Ο χρήστης εισέρχεται στην εφαρμογή.
2. Στέλνει την εντολή.
3. Το δίκτυο μεταφέρει την εντολή στην κεντρική μονάδα επεξεργασίας.
4. Η συνέχεια είναι ίδια ανά περίπτωση με τα αντίστοιχα σενάρια καλής λειτουργίας.

Αναμενόμενο αποτέλεσμα: Ίδια με τα αντίστοιχα σενάρια καλής λειτουργίας.

iii. Καταστάσεις Ειδοποιήσεων (warnings)

1. Αν ο αισθητήρας βροχής ανιχνεύσει βροχή, θα κλείσει τη θυρίδα, και θα εμφανίσει μια προειδοποίηση.
2. Αν χαθεί η επικοινωνία με τον αισθητήρα θέσης του τηλεσκοπίου, θα εμφανίσει μια προειδοποίηση και δεν θα δέχεται την εντολή συγχρονισμού με αυτό.
3. Αν χαθεί η επικοινωνία με τον αισθητήρα θέσης του θόλου, θα εμφανίσει μια προειδοποίηση και θα δέχεται μόνο χειροκίνητη εντολή περιστροφής.

iv. Καταστάσεις που προκαλούν σήμα κινδύνου (Alarm)

1. Αν χαθεί η επικοινωνία με τον αισθητήρα βροχής θα σταλεί στην διεπαφή ένα σήμα κινδύνου, θα κλείσει τη θυρίδα και δεν θα δέχεται άλλες εντολές.
2. Αν παρουσιαστεί timeout κατά την περιστροφή αυτό είναι σοβαρή ένδειξη μη λειτουργίας η οποία μπορεί να οφείλεται είτε στον κινητήρα του θόλου, είτε στην επικοινωνία με τους αισθητήρες είτε σε κάποια άλλη εμπλοκή του μηχανισμού. Σε αυτή την περίπτωση εμφανίζει alarm και δεν δέχεται άλλες εντολές πλην του κλεισίματος της θυρίδας.
3. Αν παρουσιαστεί timeout κατά το άνοιγμα της θυρίδας υπάρχουν δυο ενδεχόμενα. Αν η κατάσταση της θυρίδας είναι ανοιχτή (ενώ θα έπρεπε τέρμα ανοιχτή), τότε το πιθανότερο είναι βλάβη στον τερματικό ανοίγματος της θυρίδας. Αν η κατάσταση της θυρίδας παραμένει κλειστή, τότε έχει βλάβη ο τερματικός κλεισίματος ή ο κινητήρας της θυρίδας. Σε αυτή την περίπτωση εμφανίζει alarm και αν η κατάσταση της θυρίδας είναι ανοιχτή, κλείνει τη θυρίδα και δεν δέχεται άλλες εντολές.
4. Αν παρουσιαστεί timeout κατά το κλείσιμο της θυρίδας υπάρχουν δυο ενδεχόμενα. Αν η κατάσταση της θυρίδας είναι ανοιχτή, τότε το πιθανότερο είναι βλάβη στον τερματικό κλεισίματος της θυρίδας. Αν η κατάσταση της θυρίδας παραμένει τέρμα ανοιχτή, τότε έχει βλάβη ο τερματικός ανοίγματος ή/και κλεισίματος ή ο κινητήρας της θυρίδας. Σε αυτή την περίπτωση εμφανίζει alarm και δεν δέχεται άλλες εντολές.

5. Αν διακοπεί η απομακρυσμένη επικοινωνία για χρόνο μεγαλύτερο από κάποιο προκαθορισμένο και δεν βρίσκεται κάποιος στο χώρο του θόλου, τότε επιστρέφει στην κατάσταση Power ON.

5. Υλοποίηση Συστήματος

5.1 Εισαγωγή

Στο παρόν κεφάλαιο περιγράφεται η κατασκευή ενός μοντέλου θόλου αστεροσκοπείου που σκοπός του είναι να επιδείξει τη λειτουργία, τα συστατικά μέρη και τον τρόπο υλοποίησης σε ένα αστεροσκοπείο καθώς και στο αστεροσκοπείο της Υπάτης. Η κατασκευή αποτελείται από έτοιμες πλακέτες του εμπορίου συμβατές με την πλατφόρμα Arduino οι οποίες διασυνδέονται με την κεντρική μονάδα. Η υλοποίηση είναι η απλούστερη που θα μπορούσε να γίνει ώστε να γίνει κατανοητή στα πλαίσια της πτυχιακής εργασίας και να μπορεί εύκολα να αναπαραχθεί και να επεκταθεί.

Όπως αναφέρθηκε σε προηγούμενο σημείο της πτυχιακής για την προσομοίωση του θόλου χρησιμοποιήθηκε μια περιστρεφόμενη κάμερα του εμπορίου από την οποία αφαιρέθηκαν τα ηλεκτρονικά μέρη και κρατήθηκε το περίβλημα, ο κινητήρας περιστροφής και η κάμερα με τον κινητήρα ανοδικής/καθοδικής κίνησης. Το περίβλημα της Tapo αποτελείται από δυο μέρη, τη βάση και το περιστρεφόμενο μέρος που φιλοξενεί την κάμερα. Αυτά διαχωρίστηκαν και η μεν βάση μαζί με το μοτέρ περιστροφής το οποίο αφαιρέθηκε και τοποθετήθηκε ανάποδα στη βάση προσομοιώνουν τον θόλο του αστεροσκοπείου, το δε πάνω μέρος με την κάμερα η οποία μετακινείται πάνω/κάτω προσομοιώνουν τη θυρίδα του θόλου. Αυτά, μαζί με τα modules τοποθετούνται σε μία επίπεδη επιφάνεια όπως ένα πλέξι γκλας για της ανάγκες διασύνδεσης και επίδειξης του μοντέλου. Είναι προφανές ότι τα ηλεκτρονικά που χρησιμοποιούνται στο μοντέλο για την κίνηση του κινητήρα του μοντέλου δεν είναι κατάλληλα για χρήση στον θόλο του αστεροσκοπείου καθώς εκεί χρησιμοποιείται τριφασικός κινητήρας υψηλής ισχύος και θα χρειαστεί inverter για την τροφοδοσία του. Η δε επέκταση του μοντέλου ώστε να λειτουργήσει με ηλεκτρονικά ισχύος θα αναφερθεί σε επόμενο κεφάλαιο.

5.2 Συστατικά στοιχεία του μοντέλου

Σε αυτό το τμήμα παρουσιάζονται τα τμήματα που συνθέτουν το μοντέλο.

5.2.1. Κάμερα εσωτερικού χώρου TAP0 C210 για την προσομοίωση του θόλου.

Η εικονιζόμενη κάμερα μετά από κάποιες τροποποιήσεις που περιγράφονται στη συνέχεια χρησιμοποιείται για την προσομοίωση των λειτουργιών του θόλου.



Εικόνα 16 - Κάμερα TAP0 C210

Αποτελείται από δύο κύρια μέρη, την κάμερα, και τη βάση.

Αρχικά, διαχωρίζουμε τη βάση από το πάνω τμήμα και αφαιρούμε την πλακέτα με τον connector τροφοδοσίας. Αφαιρούμε το μοτέρ με το καλώδιό του ξεβιδώνοντάς το από το πάνω τμήμα και τοποθετούμε στη βάση ένα παρόμοιο βηματικό κινητήρα τύπου 28BYJ-48 ο οποίος έρχεται έτοιμος με καλώδιο και connector σύνδεσης. Η βάση μαζί με το μοτέρ θα προσομοιώνει την κίνηση του θόλου.

Το μοτέρ που κινεί την κάμερα πάνω κάτω θα χρησιμοποιηθεί για την προσομοίωση του ανοίγματος/κλεισίματος της θυρίδας. Επειδή ο connector του μοτέρ της κάμερας δεν ταιριάζει με τον connector του module στο οποίο θα συνδεθεί, τον κόβουμε και στη θέση του κολλάμε έναν κατάλληλο connector.



Εικόνα 17 - Αποσυναρμολόγηση βάσης και τοποθέτηση κινητήρα



Εικόνα 18 - Επάνω τμήμα μετά την αφαίρεση της βάσης

5.2.2 Arduino UNO R4 WiFi

Αποτελεί την κεντρική μονάδα του συστήματος [9]. Διασυνδέει τα περιφερειακά και τους αισθητήρες με την κεντρική μονάδα. Είναι εξοπλισμένη με Wi-Fi και μπορεί να συνδεθεί απομακρυσμένα με το Arduino Cloud.



Εικόνα 19 - Arduino UNO R4 WiFi

5.2.3 Μονοπολικός βηματικός κινητήρας και πλακέτα οδήγησης

Ο μονοπολικός βηματικός κινητήρας χρησιμοποιεί πέντε καλώδια και λειτουργεί με ένα σταθερό κοινό σημείο για κάθε τύλιγμα [10]. Η εναλλαγή του ρεύματος μεταξύ των τυλιγμάτων επιτρέπει την περιστροφή του κινητήρα βήμα-βήμα.

Για τη διασύνδεσή του χρησιμοποιείται το ULN2003 το οποίο συναντάται κολλημένο σε έτοιμη πλακέτα για την εύκολη χρήση του.

Αφού συνδέσουμε τον λευκό συνδετήρα στην αντίστοιχη υποδοχή της πλακέτας αυτό που μένει είναι να συνδέσουμε τα VDD, GND στην τροφοδοσία και τα In1, In2, In3, In4 σε ελεύθερες ψηφιακές εξόδους του Arduino.

Στην περίπτωση μας θα συνδέσουμε τον κινητήρα της κάμερας TAPO C210 ο οποίος είναι παρόμοιος με τον διαδεδομένο κινητήρα 28BYJ-48 χρησιμοποιεί όμως διαφορετικό connector. Για αυτό θα τον κόψουμε και θα κολλήσουμε τον κατάλληλο connector προκειμένου να τον συνδέσουμε με την πλακέτα του driver.

Στο μοντέλο χρησιμοποιούμε δυο βηματικούς κινητήρες, έναν για την περιστροφή του θόλου και έναν για το άνοιγμα/κλείσιμο της θυρίδας.



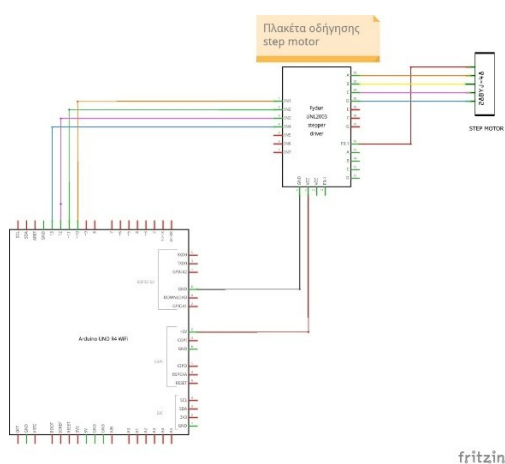
Εικόνα 20 – Βηματικός κινητήρας



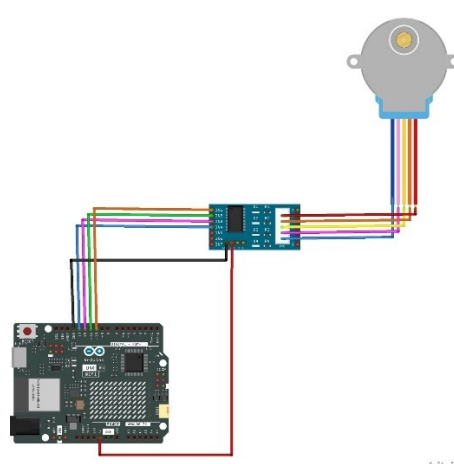
Εικόνα 21 - Step motor Driver

Πίνακας 1 – Παράδειγμα σύνδεσης Step motor driver

Πλακέτα οδήγησης βημ. κινητήρα	Arduino Uno R4 WiFi
IN1	D10
IN2	D11
IN3	D12
IN4	D13
GND	GND
VDD	5V



Εικόνα 22 - Διάγραμμα σύνδεσης βηματικού κινητήρα



Εικόνα 23 - Παράδειγμα σύνδεσης βηματικού κινητήρα



Εικόνα 24 - Συναρμολογημένος driver και κινητήρας

Δοκιμαστικό πρόγραμμα οδήγησης βηματικού κινητήρα

Το παρακάτω πρόγραμμα σκοπό έχει τη δοκιμή καλής λειτουργίας του βηματικού κινητήρα. Περιστρέφει εναλλάξ (δεξιόστροφα/αριστερόστροφα) τον κινητήρα κατά μια πλήρη στροφή κάθε φορά.

```
// ΔΟΚΙΜΑΣΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΟΔΗΓΗΣΗΣ
// ΒΗΜΑΤΙΚΟΥ ΚΙΝΗΤΗΡΑ

#include <Stepper.h>

const int steps = 2048; // Πλήθος βημάτων ενός πλήρους κύκλου
Stepper MyStepper = Stepper(steps, 13, 11, 10, 12);

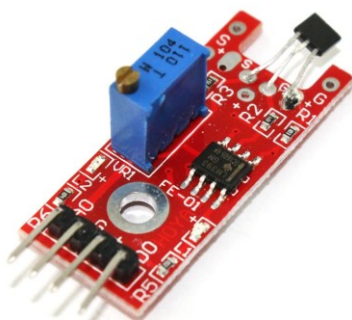
void setup() {
  // Ρύθμιση της ταχύτητας σε 10 στροφές ανά λεπτό (RPM)
  // Έως 15
  MyStepper.setSpeed(10);
}

void loop() {
  // Κάνει μια πλήρη περιστροφή δεξιόστροφα
  MyStepper.step(steps);
  delay(1000);

  // Κάνει μια πλήρη περιστροφή αριστερόστροφα
  MyStepper.step(-steps);
  delay(1000);
}
```

5.2.4 Hall Sensor Board

Ανιχνεύει την ύπαρξη μαγνητικού πεδίου. Χρησιμοποιούνται τρεις, ένας για τον εντοπισμό της αρχικής θέσης του θόλου, ένας για τον έλεγχο πλήρους κλεισίματος της θυρίδας και ένας για τον έλεγχο πλήρους ανοίγματος της θυρίδας.



Εικόνα 25 - Πλακέτα αισθητήρα Hall



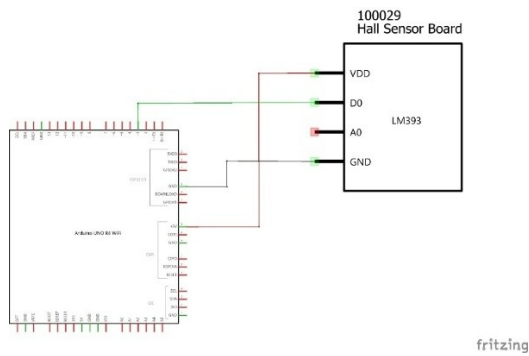
Εικόνα 26 - Συναρμολόγηση του αισθητήρα Hall

Ο αισθητήρας Hall είναι ένας αισθητήρας μαγνητικού πεδίου που χρησιμοποιεί το φαινόμενο Hall για να ανιχνεύσει την παρουσία, την ένταση ή την κατεύθυνση ενός μαγνητικού πεδίου. Το φαινόμενο Hall βασίζεται στη δημιουργία μιας τάσης όταν ένα ρεύμα περνάει μέσα από έναν αγωγό που βρίσκεται υπό την επίδραση ενός μαγνητικού πεδίου [8]. Παρέχει μια αναλογική και μια ψηφιακή έξοδο. Η αναλογική έξοδος μας δίνει μια τάση το μέγεθος της οποίας είναι ανάλογο με την ένταση του μαγνητικού πεδίου. Η ψηφιακή έξοδος είναι στην πραγματικότητα η έξοδος ενός συγκριτή ο οποίος μέσω του ρυθμιστικού ποτενσιόμετρου ρυθμίζουμε την ευαισθησία του ώστε όταν ξεπεραστεί να μας δίνει στην ψηφιακή έξοδο High και Low στην αντίθετη περίπτωση.

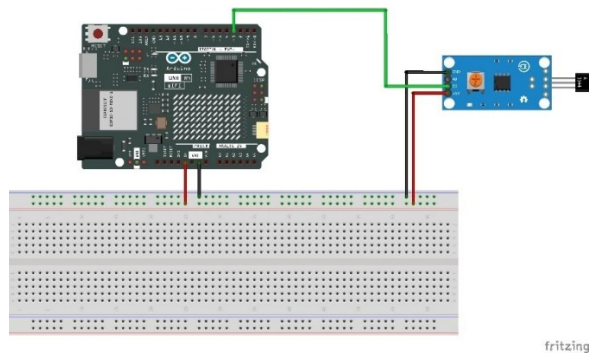
Εδώ χρησιμοποιείται προκειμένου να ελέγχεται εάν η θέση του θόλου βρίσκεται στην αρχική του θέση. Αυτό γίνεται τοποθετώντας έναν αισθητήρα hall σε μια σταθερή θέση και έναν μαγνήτη πάνω στον περιστρεφόμενο θόλο ο οποίος περιστρέφεται μαζί με το θόλο και όταν βρεθεί εντός του πεδίου ανίχνευσης του αισθητήρα παρέχει ένα ψηφιακό σήμα στην πλακέτα. Το σημείο αυτό θα ρυθμίζεται έτσι ώστε η αρχική θέση του θόλου να συμπίπτει με την αρχική θέση του τηλεσκοπίου και θα είναι οι 0 μοίρες γωνία περιστροφής.

Πίνακας 2 – Παράδειγμα σύνδεσης πλακέτας αισθητήρα hall

Πλακέτα οδήγησης βημ. κινητήρα	Arduino Uno R4 WiFi
D0	D3
G	GND
+	+5V



Εικόνα 27 - Διάγραμμα σύνδεσης αισθητήρα Hall



Εικόνα 28 - Παράδειγμα σύνδεσης αισθητήρα Hall

Δοκιμαστικό πρόγραμμα ελέγχου του αισθητήρα Hall

Το παρακάτω πρόγραμμα σκοπό έχει τη δοκιμή καλής λειτουργίας του αισθητήρα Hall. Κάθε φορά που ο αισθητήρας βρίσκεται εντός μαγνητικού πεδίου ανάβει το led.

```
// ΔΟΚΙΜΑΣΤΙΚΟ ΠΡΟΓΡΑΜΜΑ HALL SENSOR
// Όταν ο αισθητήρας βρεθεί εντός μαγνητικού πεδίου
// ανάβει το led του Arduino
```

```
int HallPin=3; //define hall sensor port

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(HallPin, INPUT);
}

void loop() {
  int hall;

  hall = digitalRead(HallPin);
  if (hall)
    digitalWrite(LED_BUILTIN, HIGH);
  else
    digitalWrite(LED_BUILTIN, LOW);
}
```

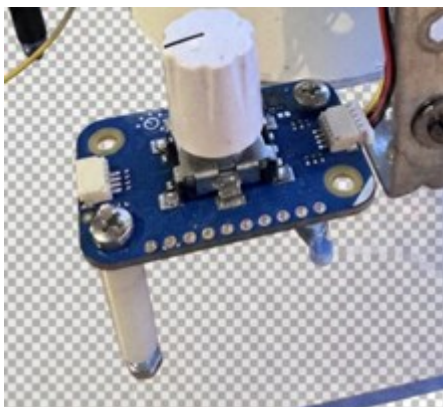
5.2.5 Modulino Knob (rotary encoder)



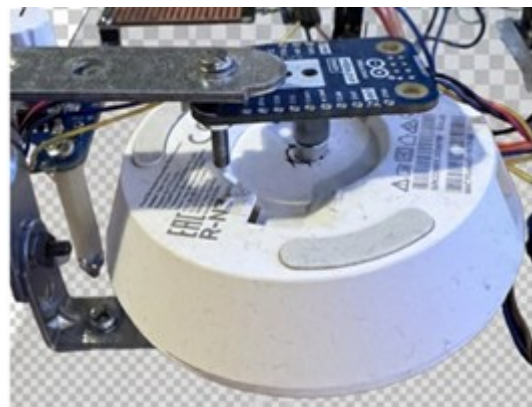
Πρόκειται για ένα περιστροφικό κωδικοποιητή με ενσωματωμένο διακόπτη ο οποίος συνδέεται με το Arduino Uno R4 WiFi μέσω της σειριακής θύρας Qwiic. Χρησιμοποιούνται δύο, ένας ως παράδειγμα χρήσης encoder για τον προσδιορισμό της θέσης του θόλου και ένας ως παράδειγμα χρήσης encoder για τον προσδιορισμό της θέσης του τηλεσκοπίου έτσι ώστε να είναι εφικτός ο συγχρονισμός της θέσης του θόλου με τη θέση του τηλεσκοπίου. Στο μοντέλο η εξομοίωση του τηλεσκοπίου επιτυγχάνεται με την χειροκίνητη περιστροφή του άξονα του knob. Το δεύτερο knob παίζει το ρόλο του αισθητήρα περιστροφής και προσαρμόζεται επάνω στη βάση της κάμερας που έχουμε προηγουμένως αποσπάσει και τοποθετήσει στη άλλη πλευρά τον κινητήρα για να την περιστρέφει.

Το συγκεκριμένο εξάρτημα έχει διακριτικότητα 30 βημάτων σε μια πλήρη περιστροφή η οποία αρκεί για τις ανάγκες του μοντέλου.

Το κάθε modulino knob έχει τη default διεύθυνση 0x3A για την επικοινωνία με τον δίαυλο I2C. Όμως επειδή χρησιμοποιούνται δύο, είναι απαραίτητο να αλλαχτεί η διεύθυνση του ενός. Για το σκοπό αυτό χρησιμοποιείται το example AddressChanger. Λεπτομέρειες υπάρχουν στην official τεκμηρίωση του Modulino [12].



Εικόνα 29 - Knob για την εξομοίωση περιστροφής τηλεσκοπίου



Εικόνα 30 - Knob τοποθετημένο στον περιστρεφόμενο θόλο

5.2.6 Modulino Buttons

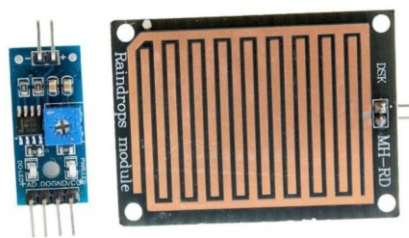


Πρόκειται για μια πλακέτα που φιλοξενεί τρία SPST push buttons και ένα ενδεικτικό led για το κάθε button. Χρησιμοποιεί για την είσοδο από το χρήστη εντολών ελέγχου, όπως περιστροφή, άνοιγμα/κλείσιμο θυρίδας, κλπ. Συνδέεται με το Arduino Uno R4 WiFi μέσω της σειριακής θύρας Qwiic.

Το κάθε modulino buttons έχει τη default διεύθυνση 0x3E για την επικοινωνία με τον δίαυλο I2C. Όμως επειδή χρησιμοποιούνται δύο, είναι απαραίτητο να αλλάξει η διεύθυνση του ενός. Για το σκοπό αυτό χρησιμοποιείται το example AddressChanger. Λεπτομέρειες υπάρχουν στην official τεκμηρίωση του Modulino.

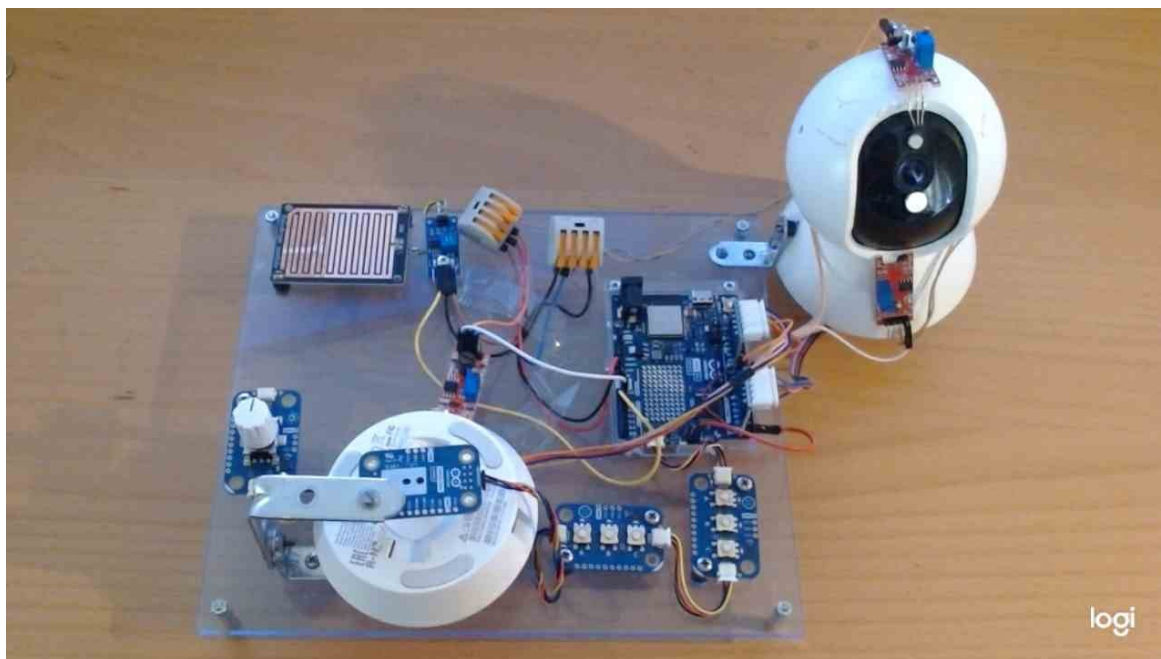
5.2.7 Rain Sensor Module

Ανιχνεύει την ύπαρξη ή όχι βροχής.

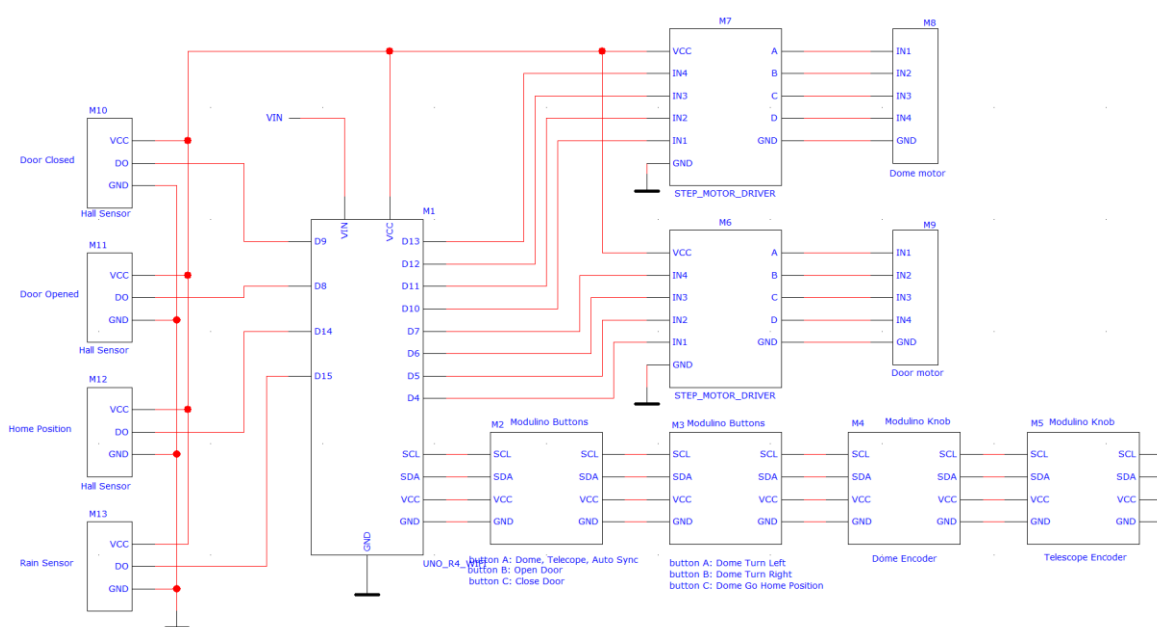


Εικόνα 31 - Αισθητήρας βροχής

5.3 Συναρμολογημένο σύστημα



Εικόνα 32 - Συναρμολογημένο σύστημα



Εικόνα 33 - Διάγραμμα συνδεσμολογίας

5.4 Περιγραφή του κυκλώματος

Το σύστημα είναι χτισμένο γύρω από το Arduino Uno R4 WiFi (M1) και το οποίο διαχειρίζεται την επικοινωνία με τον χρήστη, την ανάγνωση των αισθητήρων και την κίνηση των μοτέρ. Η αρχιτεκτονική του συστήματος χωρίζεται σε τρεις κύριους τομείς:

1. Διάυλος Επικοινωνίας I²C και Διεπαφή Χρήστη

Για τον τοπικό έλεγχο, χρησιμοποιούνται modules της οικογένειας Modulino, τα οποία επικοινωνούν με το Arduino μέσω του διαύλου I²C (Qwiic). Τα modules μπορεί να φαίνεται ότι συνδέονται σε σειρά, στην πραγματικότητα όμως συνδέονται παράλληλα πάνω στο δίαυλο I2C μειώνοντας έτσι δραστικά την πολυπλοκότητα της καλωδίωσης:

- *Modulino Buttons* (M2 & M3): Επιτρέπουν στον χρήστη να επιλέγει τις καταστάσεις λειτουργίας (Auto Sync, Dome/Telescope control) και να εκτελεί άμεσες εντολές, όπως το άνοιγμα/κλείσιμο της θυρίδας (Open/Close Door), την χειροκίνητη περιστροφή του θόλου καθώς και την επαναφορά του στην αρχική του θέση (Go Home).
- *Modulino Knobs* (M4 & M5): Λειτουργούν ως ψηφιακοί κωδικοποιητές (Encoders). Το M4 ελέγχει την περιστροφή του θόλου, ενώ το M5 χρησιμοποιείται για την προσομοίωση της θέσης του τηλεσκοπίου. Τα encoders αυτών των modules διαθέτουν επίσης και ένα push button το οποίο ενεργοποιείται με την πίεση του περιστρεφόμενου άξονα. Το συγκεκριμένο χαρακτηριστικό χρησιμοποιείται στο M5 ώστε να υποδηλώσουμε στον ελεγκτή ότι το τηλεσκόπιο βρίσκεται στην αρχική του θέση.

2. Μονάδα οδήγησης κινητήρων (step motors)

Η μονάδα αυτή ελέγχει δύο ανεξάρτητους βηματικούς κινητήρες (Stepper Motors) μέσω ψηφιακών εξόδων:

- *Περιστροφή Θόλου*: Ο driver M7 λαμβάνει σήματα από τις ψηφιακές εξόδους D10-D13 για να οδηγήσει το μοτέρ M8. Το μοτέρ είναι υπεύθυνο για την περιστροφή του θόλου.

- *Έλεγχος θυρίδας:* Ο driver M6 λαμβάνει σήματα από τις ψηφιακές εξόδους D4-D7 για να οδηγήσει το μοτέρ M9. Αυτό το μοτέρ αναλαμβάνει το άνοιγμα και το κλείσιμο της θυρίδας του θόλου.

3. Σύστημα Ανάδρασης και Ασφάλειας (Sensors)

Για την ακρίβεια της κίνησης και την προστασία του εξοπλισμού, χρησιμοποιούνται τέσσερις αισθητήρες, οι οποίοι παρέχουν ανάδραση σε πραγματικό χρόνο.

Πίνακας 3 – Αισθητήρες

Αισθητήρας	Pin Arduino	Λειτουργία
Door Closed (M10)	D9	Αναγνωρίζει το πλήρες κλείσιμο της θυρίδας.
Door Opened (M11)	D8	Αναγνωρίζει το πλήρες άνοιγμα της θυρίδας.
Home Position (M12)	D14	Χρησιμοποιείται για την ανίχνευση του αρχικού σημείου 0 του θόλου.
Rain Sensor (M13)	D15	Σε περίπτωση ανίχνευσης βροχής, το σύστημα δίνει εντολή για το κλείσιμο της θυρίδας.

6. Απομακρυσμένος Έλεγχος

Για την υλοποίηση του απομακρυσμένου ελέγχου (Remote Control) επιλέχθηκε η πλατφόρμα **Arduino Cloud**. Η συγκεκριμένη πλατφόρμα επιτρέπει την αμφίδρομη ανταλλαγή δεδομένων σε πραγματικό χρόνο, χρησιμοποιώντας ασφαλή πρωτόκολλα (όπως το MQTT [4]), μεταξύ του μικροελεγκτή (Arduino Uno R4 WiFi) και του διαδικτυακού πίνακα ελέγχου (Dashboard) της πλατφόρμας.

Τα βήματα που πρέπει να ακολουθηθούν είναι:

- Δημιουργία Λογαριασμού στο Arduino Cloud.
- Εγγραφή της πλακέτας μας στο Arduino Cloud ως νέο Device.
- Δημιουργία Thing.
- Συσχέτιση του Device που ορίσαμε προηγουμένως με το Thing.
- Δημιουργία Sketch.
- Σύνδεση στο δίκτυο

Οδηγίες για την υλοποίηση των παραπάνω εργασιών υπάρχουν [13] στην official τεκμηρίωση της πλατφόρμας Arduino Cloud.

Όταν ολοκληρωθούν τα παραπάνω βήματα, θα έχει δημιουργηθεί ένας ελάχιστος αρχικός κώδικας στο sketch ο οποίος είναι απαραίτητος για την επικοινωνία με το διαδίκτυο και το cloud.

Στη συνέχεια χρησιμοποιώντας το εργαλείο Dashboard της πλατφόρμας δημιουργούμε το γραφικό περιβάλλον το οποίο θα αλληλοεπιδρά με το χρήστη.

Dashboard είναι ο πίνακας ελέγχου της απομακρυσμένης πλακέτας. Το εργαλείο αυτό μας παρέχει τη δυνατότητα να τοποθετήσουμε σε ένα panel τα Widgets με τα οποία θα αλληλοεπιδράμε με την πλακέτα μας απομακρυσμένα. Τα Widgets χωρίζονται στα Interaction Widgets τα οποία είναι χειριστήρια μέσω των οποίων στέλνουμε απομακρυσμένες εντολές ή και τιμές σε μεταβλητές, Visualization Widgets μέσω των οποίων παίρνουμε δεδομένα, και τέλος τα Annotation Widgets.

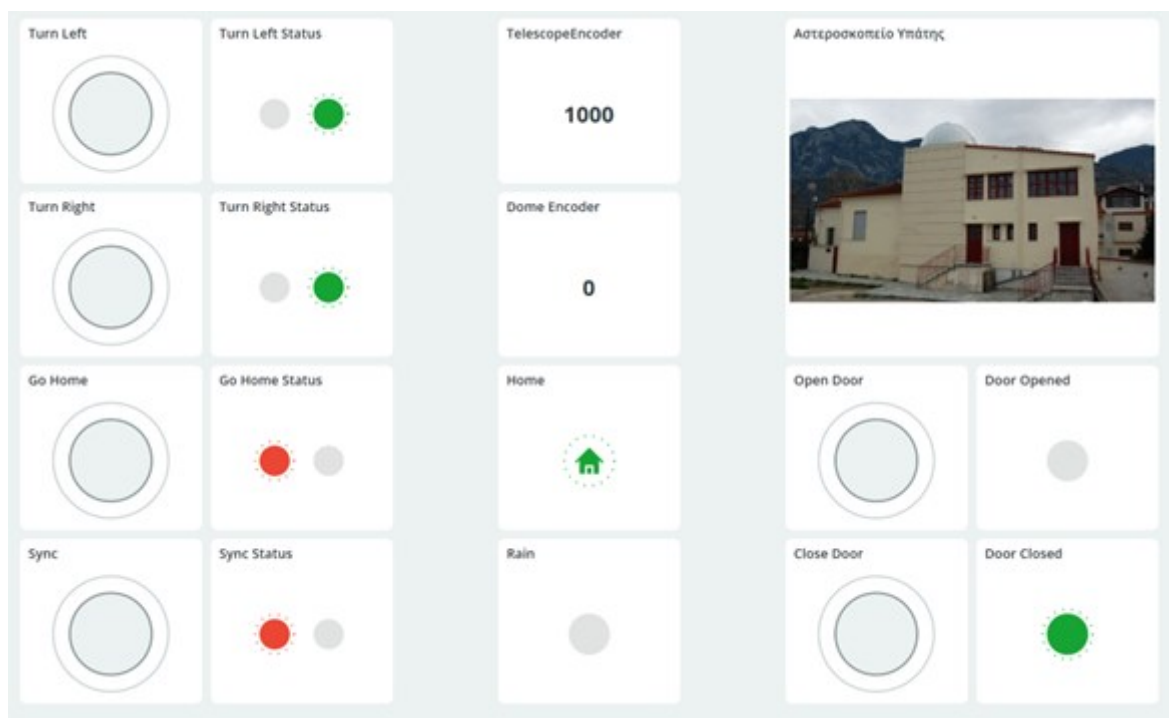
Στο Dashboard έχουμε πρόσβαση είτε από υπολογιστή είτε από κινητό μέσω της εφαρμογής Arduino IoT Remote.

Εκτός από το Dashboard, θα πρέπει να δημιουργήσουμε μεταβλητές cloud τις οποίες και θα συνδέσουμε με τα widgets. Ονομάζονται μεταβλητές cloud επειδή μέσω αυτών ελέγχουμε

και παρακολουθούμε το Thing. Μπορούμε για παράδειγμα να διαβάζουμε τιμές όπως τη θέση του θόλου την οποία μας τη δίνει ο encoder καθώς και να στέλνουμε λογικές εντολές όπως εντολή περιστροφής, κλείσιμο/άνοιγμα θυρίδας κλπ. Οι μεταβλητές cloud είναι αυτές μέσω των οποίων γίνεται η επικοινωνία της εφαρμογής μας στην πλακέτα με το cloud περιβάλλον και των widgets στο dashboard.

Για κάθε μεταβλητή cloud που δημιουργούμε ενημερώνεται αυτόματα και ο κώδικας στο Sketch. Συγκεκριμένα, ορίζεται η μεταβλητή και προθέτονται νέες συναρτήσεις ελέγχου όπου εκεί μπορούμε προαιρετικά να γράψουμε κώδικα ο οποίος θα τρέχει όταν υπάρχει κάποια μεταβολή στην αντίστοιχη μεταβλητή.

Όταν ολοκληρώσουμε και τη δημιουργία των μεταβλητών Cloud, έχουμε έτοιμο τον βασικό κώδικα απομακρυσμένης επικοινωνίας και μπορούμε μέσω των μεταβλητών να στέλνουμε στο απομακρυσμένο Thing εντολές και να διαβάζουμε από αυτό δεδομένα.



Εικόνα 34 - Screenshot του ολοκληρωμένου dashboard

Μεταβλητές Cloud

Οι μεταβλητές του συστήματος δημιουργούνται αυτόματα στο αρχείο thingProperties.h και χωρίζονται σε δύο βασικές κατηγορίες, ανάλογα με την κατεύθυνση ροής της πληροφορίας (Read/Write ή Read Only).

Χωρίζονται

- α) στις μεταβλητές εντολών ελέγχου μέσω των οποίων ο χρήστης αλλάζει την τιμή τους μέσω του απομακρυσμένου UI (π.χ. πατώντας ένα κουμπί) και το Arduino λαμβάνει την εντολή για να εκτελέσει την αντίστοιχη κίνηση, και
- β) στις μεταβλητές τύπου read only μέσω των οποίων ελέγχουμε την κατάσταση του συστήματος και οι οποίες ενημερώνονται από το λογισμικό εφαρμογής του Arduino βάσει των αισθητήρων και παρουσιάζονται στο χρήστη από τα widgets του Dashboard.

Πίνακας 4 – Cloud Εντολές Ελέγχου

Όνομα Μεταβλητής	Τύπος	Περιγραφή / Λειτουργία
turnLeft	bool	Χειροκίνητη περιστροφή του θόλου προς τα αριστερά. Εκτελεί την ίδια λειτουργία με το button A του M3 του κυκλωματικού διαγράμματος.
turnRight	bool	Χειροκίνητης περιστροφή του θόλου προς τα δεξιά. Εκτελεί την ίδια λειτουργία με το button B του M3 του κυκλωματικού διαγράμματος.
goHome	bool	Αυτόματη επιστροφή του θόλου στο σημείο μηδέν. Εκτελεί την ίδια λειτουργία με το button C του M3 του κυκλωματικού διαγράμματος.
sync	bool	Ενεργοποίηση/Απενεργοποίηση της λειτουργίας "Auto Sync" για την παρακολούθηση του τηλεσκοπίου. Απαραίτητη προϋπόθεση για να ενεργοποιηθεί είναι η θέση του τηλεσκοπίου να έχει μηδενιστεί με την πίεση του άξονα του encoder καθώς επίσης ο θόλος να βρίσκεται στο σημείο 0. Εκτελεί την ίδια λειτουργία με το button A του M2 του κυκλωματικού διαγράμματος.
openDoor	bool	Προκαλεί το άνοιγμα της θυρίδας παρατήρησης αν δεν είναι ανοιχτή. Εκτελεί την ίδια λειτουργία με το button B του M2 του κυκλωματικού διαγράμματος.
closeDoor	bool	Προκαλεί το κλείσιμο της θυρίδας παρατήρησης αν δεν είναι κλειστή. Εκτελεί την ίδια λειτουργία με το button C του M2 του κυκλωματικού διαγράμματος.

Οι παρακάτω μεταβλητές λειτουργούν ως ενδείκτες στο Dashboard, αντικατοπτρίζοντας την κατάσταση των αισθητήρων και την κατάσταση λειτουργίας :

Πίνακας 5 - Ενδείκτες κατάστασης του συστήματος

Όνομα Μεταβλητής	Τύπος	Περιγραφή / Λειτουργία
ledDomeLeft	bool	Ανάβει πράσινο όταν η εντολή “Turn Left” μπορεί να χρησιμοποιηθεί, διαφορετικά ανάβει κόκκινο.
ledDomeRight	bool	Ανάβει πράσινο όταν η εντολή “Turn Right” μπορεί να χρησιμοποιηθεί, διαφορετικά ανάβει κόκκινο.
ledDomeHome	bool	Ανάβει πράσινο όταν η εντολή “Go Home” μπορεί να χρησιμοποιηθεί, διαφορετικά ανάβει κόκκινο.
ledDomeAuto (Sync Status)	bool	Ανάβει πράσινο όταν ο θόλος βρίσκεται σε κατάσταση αυτόματου συγχρονισμού με το τηλεσκόπιο.
positionTelescope (telescopeEncoder)	int	Εμφανίζει τη θέση του encoder του τηλεσκοπίου στην κλίμακα 0 – 29 (30 θέσεις) σε σχέση με την αναφορά του η οποία μηδενίζεται με την πίεση του άξονα του encoder.
positionDome (Dome Encoder)	int	Εμφανίζει τη θέση του encoder του θόλου στην κλίμακα 0 – 29 (30 θέσεις) σε σχέση με την αναφορά του η οποία μηδενίζεται με τη θέση 0.
homeSensor (Home)	bool	Ανάβει πράσινο όταν ο θόλος βρίσκεται στο σημείο 0.
openedSensor (Door Opened)	bool	Ανάβει πράσινο όταν η θυρίδα είναι τελείως ανοιχτή.
closedSensor (Door Closed)	bool	Ανάβει πράσινο όταν η θυρίδα είναι τελείως κλειστή.
rainSensor (Rain)	bool	Ανάβει πορτοκαλί όταν έχει ενεργοποιηθεί ο αισθητήρας βροχής

7. Λογισμικό εφαρμογής

Ολόκληρος ο κώδικας παρουσιάζεται στο παράρτημα της εργασίας. Εδώ περιγράφεται η αρχιτεκτονική του λογισμικού.

7.1 Εισαγωγή στη μη-αποκλειστική εκτέλεση (Non-Blocking Architecture)

Μία από τις μεγαλύτερες προκλήσεις στη δημιουργία αυτοματισμών με μικροελεγκτές (όπως το Arduino) είναι η ταυτόχρονη διαχείριση πολλών εργασιών. Η χρήση συναρτήσεων καθυστέρησης (π.χ. `delay()`) ή η εκτέλεση μεγάλων βημάτων στους κινητήρες, "μπλοκάρει" τον επεξεργαστή, καθιστώντας τον τυφλό απέναντι σε κρίσιμα σήματα, όπως η ανάγνωση ενός αισθητήρα βροχής ή η λήψη μιας εντολής από το διαδίκτυο.

Για την αντιμετώπιση αυτού του προβλήματος, στον κώδικα του συστήματος ελέγχου εφαρμόστηκε η αρχιτεκτονική της *Μηχανής Πεπερασμένων Καταστάσεων* (*Finite State Machine - FSM*) [7]. Η αρχιτεκτονική αυτή επιτρέπει τη *Συνεργατική Πολυδιεργασία* (*Cooperative Multitasking*): ο κύκλος εκτέλεσης (`loop()`) δεν σταματά ποτέ. Αντίθετα, "σπάει" την εκτέλεση σε μικρά, γρήγορα βήματα, περνώντας διαδοχικά από διαφορετικές καταστάσεις.

7.2 Ο Κύκλος Χρονισμού (`timingState`)

Ο κεντρικός έλεγχος της ροής του προγράμματος γίνεται μέσω της μεταβλητής *timingState*. Ο μικροελεγκτής εκτελεί κυκλικά τα παρακάτω στάδια:

1. *timeToUpdateCloud*: Γίνεται η επικοινωνία με το Arduino IoT Cloud (`ArduinoCloud.update()`). Εκτελείται σε δικό του state ώστε να διασφαλίζεται ότι ο συγχρονισμός μέσω WiFi δεν προκαλεί καθυστερήσεις στα μοτέρ.
2. *timeForBtn1* & *timeForBtn2*: Γίνεται η ανάγνωση των εντολών από τα τοπικά κουμπιά των Modulino Buttons (μέσω του διαύλου I2C).
3. *timeForDome*: Η ρουτίνα για την περιστροφή του θόλου.
4. *timeForDoor*: Η ρουτίνα για τον έλεγχο της θυρίδας.

5. *timeToReadEncoders*: Διαβάζονται οι τρέχουσες θέσεις των ψηφιακών κωδικοποιητών (Modulino Knobs) του θόλου και του τηλεσκοπίου, και γίνεται ο έλεγχος για το αν είναι συγχρονισμένα (*synced*).
6. *timeForSerial*: Χρησιμοποιείται για αποσφαλμάτωση (*debugging*) και αποστολή μηνυμάτων στη σειριακή θύρα.

Μετά την ολοκλήρωση της τελευταίας κατάστασης, ο κύκλος ξεκινάει πάλι από την αρχή, εκτελώντας χιλιάδες επαναλήψεις το δευτερόλεπτο.

7.3 Υπο-Μηχανές Καταστάσεων (Sub-State Machines) για την Κίνηση

Μέσα στις καταστάσεις χρονισμού των κινητήρων (*timeForDome* και *timeForDoor*), υπάρχουν ανεξάρτητες Μηχανές Καταστάσεων που ελέγχουν τους βηματικούς κινητήρες.

Η Μηχανή Καταστάσεων του Θόλου (*domeState*)

Ελέγχει τον βηματικό κινητήρα του θόλου και περιλαμβάνει τις εξής υπο-καταστάσεις:

- *stopped*: Ο θόλος βρίσκεται σε κατάσταση ακινησίας, εν αναμονή εντολής.
- *movingHome*: Ο θόλος περιστρέφεται 2 βήματα σε κάθε κύκλο μέχρι ο αισθητήρας Hall (*homeSensor*) να ανιχνεύσει το σημείο 0.
- *movingLeft* / *movingRight*: Κίνηση του θόλου προς τη ζητούμενη κατεύθυνση, τοπικά ή απομακρυσμένα ύστερα από εντολή του χρήστη και για όσο χρόνο διαρκεί αυτή. Η κίνηση εκτελεί 2 βήματα σε κάθε κύκλο.
- *movingSync*: Ο θόλος βρίσκεται σε κατάσταση συγχρονισμού με το τηλεσκόπιο. Σε αυτό το state, καλείται η συνάρτηση *Syncing()*, η οποία υπολογίζει τη συντομότερη διαδρομή και μετακινεί το θόλο ώστε να ακολουθεί την κίνηση του τηλεσκοπίου. Σε αυτή την κατάσταση δεν δέχεται εντολή κίνησης από το χρήστη. Κινείται 2 βήματα σε κάθε κύκλο μέχρι να ταυτιστεί η ένδειξη του encoder του θόλου με αυτή του τηλεσκοπίου.

Η Μηχανή Καταστάσεων της Θυρίδας (*doorState*)

- *opened* / *closed*: Η θυρίδα βρίσκεται αντίστοιχα σε τελείως ανοιχτή ή κλειστή θέση.
- *opening* / *closing*: Εκτελείται η κίνηση του μοτέρ της θυρίδας σε πολύ μικρά βήματα (των 2 steps) σε κάθε κύκλο. Ο μικροελεγκτής εκτελεί 2 βήματα, βγαίνει από το

loop, διαβάζει τους αισθητήρες και ξαναμπαίνει. Αν εκτελούσε μεγάλο αριθμό βημάτων, το σύστημα δεν θα μπορούσε να διακόψει άμεσα την κίνηση τη στιγμή που θα έφτανε στον τερματικό αισθητήρα.

7.4 Αλγόριθμος Αυτόματου Συγχρονισμού (Auto Sync) και Συντομότερης Διαδρομής

Εισαγωγή στον Κυκλικό Χώρο Καταστάσεων

Κατά τον αυτόματο συγχρονισμό του θόλου με το τηλεσκόπιο, το σύστημα πρέπει να διασφαλίζει ότι η θυρίδα είναι πάντα ευθυγραμμισμένη με το οπτικό πεδίο του τηλεσκοπίου. Το σύστημα εντοπισμού θέσης χρησιμοποιεί ψηφιακούς κωδικοποιητές (encoders) οι οποίοι χωρίζουν την πλήρη περιστροφή του θόλου (360°) σε $N = 30$ διακριτά βήματα. Συνεπώς, ο χώρος καταστάσεων της θέσης είναι ένας κυκλικός δακτύλιος με τιμές από 0 έως 29.

Το πρόβλημα της γραμμικής προσέγγισης

Μια απλοϊκή (γραμμική) προσέγγιση συγχρονισμού θα σύγκρινε απλώς την τρέχουσα θέση του θόλου (Pos_{dome}) με τη θέση-στόχο του τηλεσκοπίου (Pos_{tel}). Αν ίσχυε $Pos_{tel} > Pos_{dome}$ ο κινητήρας θα περιστρεφόταν δεξιόστροφα.

Ωστόσο, σε ένα κυκλικό σύστημα, αυτή η λογική οδηγεί συχνά στη μακρύτερη δυνατή διαδρομή. Για παράδειγμα, αν ο θόλος βρίσκεται στη θέση 5 και το τηλεσκόπιο στραφεί στη θέση 25, η απλοϊκή κίνηση προς τα εμπρός ($5 \rightarrow 6 \rightarrow \dots \rightarrow 25$) απαιτεί 20 βήματα, ενώ η αντίστροφη κίνηση, διασχίζοντας το σημείο μηδέν ($5 \rightarrow 4 \rightarrow \dots \rightarrow 0 \rightarrow 29 \rightarrow \dots \rightarrow 25$), απαιτεί μόλις 10 βήματα.

Ο Αλγόριθμος Συντομότερης Διαδρομής (Shortest Path Algorithm)

Για τη βελτιστοποίηση της κίνησης, την εξοικονόμηση ενέργειας και τη μείωση του χρόνου υστέρησης (latency) μεταξύ τηλεσκοπίου και θόλου, αναπτύχθηκε στη ρουτίνα Syncing() ένας αλγόριθμος εύρεσης συντομότερης διαδρομής.

Ο αλγόριθμος υπολογίζει την απόλυτη διαφορά των δύο θέσεων. Δεδομένου ότι το μέγιστο μήκος του κύκλου είναι $N = 30$, η μέγιστη δυνατή απόσταση μεταξύ δύο οποιωνδήποτε σημείων (η διάμετρος) είναι:

$$\text{MaxDistance} = \frac{N}{2} = 15$$

Ο έλεγχος κατεύθυνσης υλοποιείται ελέγχοντας αν η απόλυτη διαφορά των θέσεων υπερβαίνει τη μέγιστη ημι-κυκλική απόσταση (15 βήματα). Η λογική διακλαδώνεται ως εξής:

1. Όταν $Postel > Posdome$:

Εάν $|Postel - Posdome| \geq 15$, τότε η συντομότερη διαδρομή είναι προς τα πίσω (αρνητικά βήματα κινητήρα), διασχίζοντας το σημείο μηδέν. Ειδιάλλως, η κίνηση γίνεται κανονικά προς τα εμπρός.

2. Όταν $Postel < Posdome$:

Εάν $|Posdome - Postel| \geq 15$, τότε η συντομότερη διαδρομή είναι προς τα εμπρός (θετικά βήματα κινητήρα), διασχίζοντας το σημείο μηδέν. Ειδιάλλως, η κίνηση γίνεται προς τα πίσω.

Η παραπάνω λογική ενσωματώθηκε στον κώδικα, εξασφαλίζοντας ότι ο θόλος επιλέγει δυναμικά την ταχύτερη διαδρομή, μειώνοντας τον χρόνο συγχρονισμού.

7.5 Ανάγνωση Ψηφιακών Κωδικοποιητών (encoders) και Χωρική Αντιστοίχιση

Εισαγωγή

Για να συγχρονίζεται ο θόλος με το τηλεσκόπιο (Auto Sync), απαιτείται συνεχής γνώση της απόλυτης θέσης τόσο του τηλεσκοπίου όσο και του θόλου. Η ανάγνωση αυτών των δεδομένων πραγματοποιείται κατά την κατάσταση `timeToReadEncoders` της κεντρικής μηχανής καταστάσεων. Ο κώδικας διαβάζει τις τιμές από τους δύο ψηφιακούς κωδικοποιητές (Modulino Knobs) μέσω του διαύλου I²C, αντιστοιχίζοντάς τες σε ένα κυκλικό σύστημα συντεταγμένων 30 βημάτων (από 0 έως 29).

Διόρθωση Ανεστραμμένου Υλικού

Κατά τη συναρμολόγηση του μοντέλου, προέκυψε ένας μηχανολογικός περιορισμός: ο κωδικοποιητής του θόλου (`encoderDome`) έπρεπε να τοποθετηθεί ανάποδα ως προς τον άξονα περιστροφής του (βλέπε [5.2.5 Modulino Knob \(rotary encoder\)](#)). Αυτό είχε ως αποτέλεσμα η δεξιόστροφη φυσική κίνηση του θόλου να καταγράφεται από τον αισθητήρα ως αρνητική μετατόπιση. Το πρόβλημα αυτό επιλύθηκε προγραμματιστικά. Ο μικροελεγκτής διαβάζει την τιμή και την πολλαπλασιάζει με το -1, αντιστρέφοντας το πρόσημο: `positionDome = -encoderDome.get( );`

Με αυτή την απλή μετατροπή, το σύστημα αναφοράς του θόλου ταυτίζεται με το σύστημα αναφοράς του τηλεσκοπίου, εξασφαλίζοντας ότι η αύξηση της τιμής και στους δύο κωδικοποιητές υποδηλώνει την ίδια φυσική κατεύθυνση περιστροφής.

Διαχείριση Κυκλικών Ορίων (Boundary Management)

Επειδή η περιστροφή είναι συνεχής (360°), οι μεταβλητές θέσης δεν μπορούν να αυξάνονται ή να μειώνονται επ' άπειρον. Ο κώδικας εφαρμόζει μια λογική επαναφοράς ορίων για να διατηρεί τις τιμές αυστηρά στο εύρος 0 έως 29:

Όταν η τιμή φτάσει το 30 (ολοκλήρωση πλήρους κύκλου), η μεταβλητή μηδενίζεται. Επιπλέον, ο μηδενισμός στο θόλο συμβαίνει και όταν ο μαγνητικός αισθητήρας βρεθεί στη θέση εκκίνησης (homeSensor), εξασφαλίζοντας την εξάλειψη οποιουδήποτε σφάλματος.

Όταν η περιστροφή γίνεται προς τα πίσω και η τιμή πέσει κάτω από το 0, ο αλγόριθμος προσθέτει το 30. Για παράδειγμα, η θέση -1 μετατρέπεται στη θέση 29.

Μια κρίσιμη λεπτομέρεια της υλοποίησης είναι η χρήση της συνάρτησης `.set()`. Όταν τα όρια ξεπεραστούν, το Arduino δεν διορθώνει απλώς την τοπική μεταβλητή (`positionDome` ή `positionTelescope`), αλλά στέλνει εντολή πίσω στα Modulino modules για να μηδενίσουν ή να διορθώσουν την εσωτερική τους μνήμη:

```
encoderTelescope.set(positionTelescope);
```

Αυτό διασφαλίζει ότι οι μεταβλητές και οι αισθητήρες παραμένουν συγχρονισμένοι.

Σημαία Συγχρονισμού (Sync Flag)

Στο τέλος της ρουτίνας, ο κώδικας εκτελεί την σύγκριση:

```
if (positionDome == positionTelescope)
```

Αν οι δύο τιμές ταυτίζονται, η μεταβλητή `synced` τίθεται σε `true`. Αυτή η σημαία είναι σημαντική για την υπόλοιπη λειτουργία του συστήματος, καθώς αποτελεί προϋπόθεση για να μπορεί ο χρήστης να ενεργοποιήσει την αυτόματη λειτουργία "Auto Sync" από το τοπικό κουμπί ή το απομακρυσμένο Dashboard. Αν το σύστημα αναγνωρίσει ότι είναι ήδη συγχρονισμένο, αποτρέπει άσκοπες κινήσεις των κινητήρων, εξοικονομώντας ενέργεια και μειώνοντας τη φθορά των γραναζιών.

8. Τρόπος Λειτουργίας

Για την καλύτερη κατανόηση της συμπεριφοράς του αυτοματισμού, παρακάτω περιγράφεται ο τρόπος λειτουργίας.

8.1 Εκκίνηση Συστήματος

Με την παροχή ρεύματος, εκτελείται η ρουτίνα `setup()`.

Αρχικά ενεργοποιούνται η σειριακή θύρα και το `matrix`. Στη συνέχεια αρχικοποιούνται τα `properties` τα οποία σχετίζονται με το `cloud`, και γίνεται η διασύνδεση στο διαδίκτυο και στο `Arduino cloud`. Σε αυτή τη φάση στο `matrix` εμφανίζονται οι διαφορετικές καταστάσεις (πρόοδος) της σύνδεσης. Όταν ολοκληρωθεί η σύνδεση στο `cloud`, το `matrix` σβήνει. Στο σημείο αυτό είναι απαραίτητο να σημειώσουμε ότι όταν θέλουμε να επανεκκινήσουμε το σύστημα, αυτό πρέπει να γίνει με `power off/on cycle`. Διαφορετικά, αν πατήσουμε το πλήκτρο `reset` δεν θα είναι δυνατή η σύνδεση λόγω κάποιων εσωτερικών `bugs`.

Στη συνέχεια γίνεται η ρύθμιση των συνδεδεμένων περιφερειακών και αποδίδονται οι αρχικές καταστάσεις.

Εάν ο θόλος δεν βρίσκεται αρχικά στη θέση μηδέν (ο `homeSensor` δεν ανιχνεύει μαγνήτη), η μεταβλητή κατάστασης `domeState` παίρνει την κατάσταση `movingHome`. Επίσης η μεταβλητή κατάστασης `doorState` παίρνει την κατάσταση `closing`. Κατόπιν αυτών αν ο θόλος δεν βρίσκεται στη θέση 0 τότε περιστρέφεται (εκτελώντας βήματα των 2 steps) και μόλις ο μαγνήτης του θόλου εισέλθει στην περιοχή του αισθητήρα `Hall (DomeHallPin)`, κάνει 50 επιπλέον βήματα για να κεντράρει τον μαγνήτη, μηδενίζει τον `encoder` του θόλου (`encoderDome.set(0)`) και ανάβει το LED `ledDomeHome`.

Η δε θυρίδα αν κατά την εκκίνηση δεν είναι κλειστή, κλείνει.

Τέλος, τα `led` κατάστασης των ενεργών `buttons` ανάβουν. Ενεργό `button` είναι αυτό το οποίο εάν πατηθεί θα δώσει κάποια εντολή.

Το σύστημα είναι πλέον έτοιμο να δεχθεί εντολές είτε τοπικά είτε απομακρυσμένα.

8.2 Έλεγχος θυρίδας

Το άνοιγμα κλείσιμο θυρίδας ελέγχεται από τα αντίστοιχα πλήκτρα. Ένα από τα δύο θα είναι ενεργό (αναμμένο `led`).

Επίσης, αν ενεργοποιηθεί ο αισθητήρας βροχής και η θυρίδα δεν είναι κλειστή, θα κλείσει αυτόματα.

8.3 Περιστροφή του θόλου

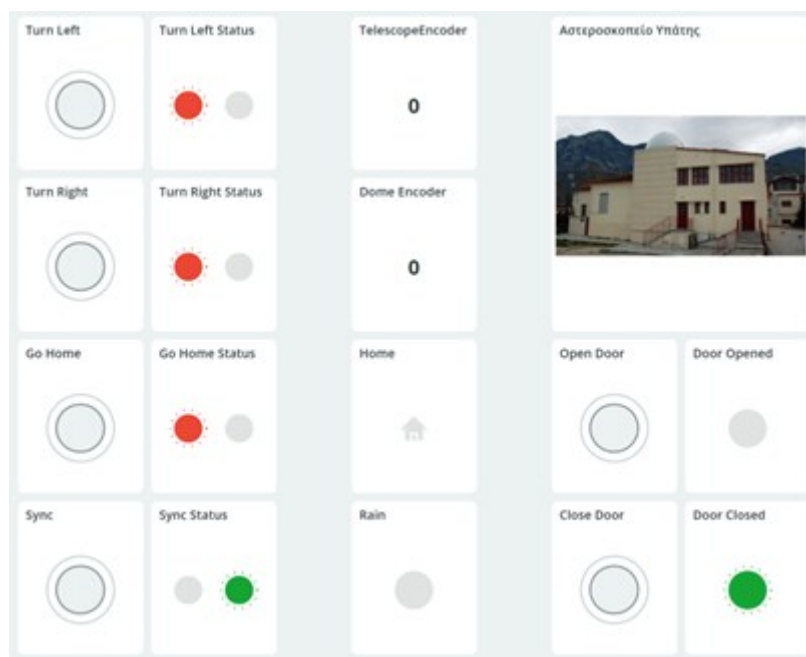
- **Χειροκίνητη περιστροφή:** Εάν δεν είναι ενεργοποιημένος ο αυτόματος συγχρονισμός, τα led των πλήκτρων περιστροφής του θόλου ανάβουν υποδηλώνοντας ότι ο χρήστης μπορεί να περιστρέψει ελεύθερα το θόλο. Πιέζοντας το ανάλογο πλήκτρο ο θόλος περιστρέφεται αριστερά ή δεξιά. Αν ο θόλος δεν βρίσκεται στο σημείο 0, πατώντας το πλήκτρο αυτόματης επαναφοράς ο θόλος περιστρέφεται αυτόματα μέχρι ο μαγνήτης του να βρεθεί εντός του πεδίου αντίχνευσης του αισθητήρα hall.
- **Αυτόματος συγχρονισμός με το τηλεσκόπιο:** Για να ξεκινήσει ο αυτόματος συγχρονισμός υπάρχουν κάποιες προϋποθέσεις. Ο θόλος και το τηλεσκόπιο πρέπει να βρίσκονται στο σημείο αναφοράς. Για το θόλο το περιγράψαμε πως γίνεται αυτό. Για το τηλεσκόπιο, το μοντέλο διαθέτει ένα encoder τύπου modolino knob το οποίο σκοπό έχει να προσομοιώνει την περιστροφή του τηλεσκοπίου. Το knob διαθέτει ένα πλήκτρο το οποίο ενεργοποιείται πιέζοντας τον άξονά του. Έτσι με την ενεργοποίηση αυτού του πλήκτρου το πρόγραμμα μηδενίζει την τιμή θέσης του τηλεσκοπίου δηλώνοντας ότι τώρα βρίσκεται στην αρχική του θέση. Αυτό χρειάζεται να γίνει μια μόνο φορά.

Με το τηλεσκόπιο και το θόλο στη θέση 0, ενεργοποιούμε τον αυτόματο συγχρονισμό με το αντίστοιχο πλήκτρο. Από αυτή τη στιγμή και μετά ο θόλος περιστρέφεται με την περιστροφή του τηλεσκοπίου είτε αριστερά είτε δεξιά παρακολουθώντας διαρκώς τη θέση του έως ότου ξαναπατήσουμε το πλήκτρο του αυτόματου συγχρονισμού.

8.4 Απομακρυσμένος έλεγχος

Το Dashboard που έχει κατασκευαστεί στη πλατφόρμα του Arduino Cloud διαθέτει εικονικά όλα τα πλήκτρα που διαθέτει η πλακέτα καθώς και απεικονίζει την κατάσταση όλων των αισθητήρων και τη θέση των encoders.

Έτσι παρόμοια με τον τοπικό χειρισμό που είδαμε προηγουμένως, χρησιμοποιώντας τώρα τα εικονικά buttons της πλατφόρμας εκτελούμε τις ίδιες εντολές.



9. Ενημέρωση της Κατάστασης του Συστήματος (visualization)

Η συνεχής ενημέρωση του χρήστη για την κατάσταση του αστεροσκοπείου αποτελεί κρίσιμη απαίτηση για την ασφαλή λειτουργία του συστήματος. Η υλοποίηση της οπτικοποίησης (visualization) στον συγκεκριμένο αυτοματισμό ακολουθεί μια πολυεπίπεδη αρχιτεκτονική, προσφέροντας ανατροφοδότηση (feedback) σε τρία διαφορετικά επίπεδα: τοπικό, απομακρυσμένο και διαγνωστικό.

Τοπική Οπτικοποίηση (Local Hardware Visualization)

Για τον χρήστη που βρίσκεται στο χώρο του αστεροσκοπείου, η ενημέρωση επιτυγχάνεται μέσω των ενσωματωμένων φωτεινών ενδείξεων (LEDs) στα Modulino Buttons.

Ο μικροελεγκτής ελέγχει δυναμικά τα LED μέσω των συναρτήσεων `buttons1.setLeds()` και `buttons2.setLeds()`, συνδέοντας τις μεταβλητές του κώδικα με άμεσο οπτικό σήμα:

Ενδείξεις: Όταν ανάβουν τα LED των πλήκτρων προσδιορίζουν ότι το συγκεκριμένο πλήκτρο είναι ενεργό και μπορεί να δεχθεί εντολή. Αυτό έμμεσα δείχνει την τρέχουσα κατάσταση του θόλου και της θυρίδας.

Λειτουργία Auto-Sync: Όταν το σύστημα κλειδώνει σε λειτουργία αυτόματης παρακολούθησης του τηλεσκοπίου, παραμένει σταθερά αναμμένο το `ledDomeAuto`.

Κατάσταση αρχικής θέσης θόλου: Όταν ο θόλος επιστρέψει και ασφαλίσει στην αρχική του θέση 0 μέσω του αισθητήρα hall, ενεργοποιείται το `ledDomeHome`, δίνοντας την οπτική επιβεβαίωση ότι ο θόλος βρίσκεται στην αρχική του θέση.

Απομακρυσμένη Οπτικοποίηση (Arduino Cloud Dashboard)

Για τον απομακρυσμένο έλεγχο, η οπτικοποίηση υλοποιείται μέσω του Arduino IoT Cloud Dashboard, το οποίο προσφέρει μια γραφική διεπαφή χρήστη (GUI) προσβάσιμη από οποιονδήποτε Web Browser ή κινητή συσκευή. Η πλατφόρμα μεταφράζει τις μεταβλητές του κώδικα σε οπτικά Widgets (γραφικά στοιχεία):

Virtual LEDs: Boolean μεταβλητές όπως οι `ledDoorOpen` και `ledDoorClose` αντιστοιχίζονται σε εικονικά LED στο Dashboard. Έτσι, ο χρήστης βλέπει με χρωματική ένδειξη (π.χ. πράσινο/κόκκινο) αν η θυρίδα είναι ανοιχτή ή κλειστή.

Ποσοτικές Ενδείξεις: Οι ακέραιες μεταβλητές positionDome και positionTelescope προβάλλονται σε αριθμητικά πεδία, δείχνοντας σε πραγματικό χρόνο τη θέση (από 0 έως 29) του θόλου και του τηλεσκοπίου.

Συναγερμοί Κρίσιμων Συμβάντων: Ο αισθητήρας βροχής (rainSensor) συνδέεται με ένα ειδικό εικονίδιο προειδοποίησης. Όταν ανιχνευθεί βροχή, ενημερώνεται η οθόνη του χρήστη, δείχνοντας ότι ενεργοποιήθηκε ο μηχανισμός έκτακτης ανάγκης.

Διαγνωστική Οπτικοποίηση Δικτύου

Το Arduino Uno R4 διαθέτει οπτικοποίηση για την κατάσταση της δικτυακής συνδεσιμότητας. Αξιοποιώντας το ενσωματωμένο LED Matrix της πλακέτας, καλείται η συνάρτηση waitForArduinoCloudConnection(matrix). Αυτή η λειτουργία εμφανίζει γραφικά πάνω στην ίδια την πλακέτα (όπως σύμβολα φόρτωσης, εικονίδια WiFi ή ένα "Tick" επιτυχίας). Αυτό επιτρέπει στον χρήστη να γνωρίζει αν το σύστημα είναι συνδεδεμένο στο διαδίκτυο και στο Cloud, χωρίς να χρειάζεται να συνδέσει υπολογιστή για να διαβάσει τη Σειριακή Θύρα.

10. Αξιολόγηση του έργου

Στις δυο πρώτες ενότητες του κεφαλαίου συγκρίνονται οι αρχικές απαιτήσεις του συστήματος (Λειτουργικών και Μη Λειτουργικών) με την τελική υλοποίηση με σκοπό την τεκμηρίωση της σωστής λειτουργίας, καθώς και την αιτιολόγηση τυχόν αποκλίσεων που προέκυψαν κατά τη φάση της κατασκευής και του προγραμματισμού. Στη συνέχεια ασχολούμαστε με την αξιολόγηση η οποία χωρίζεται σε λειτουργική (υλικό), ποιοτική/ποσοτική (λογισμικό) και χρηστική (διεπαφή), ενώ παράλληλα αναδεικνύονται οι δυσκολίες που παρουσιάστηκαν και οι λύσεις που δόθηκαν κατόπιν επανασχεδιασμού.

10.1. Αξιολόγηση Μη Λειτουργικών Απαιτήσεων

Στο σύνολό τους, οι Μη Λειτουργικές Απαιτήσεις επιτεύχθηκαν πλήρως, κυρίως χάρη στην επιλογή του Arduino Uno R4 WiFi και της αρχιτεκτονικής λογισμικού.

- 3.1.1 *Αξιοπιστία*. Από τη στιγμή που το σύστημα είναι χτισμένο πάνω στην πλατφόρμα Arduino, η αξιοπιστία του συστήματος εξαρτάται άμεσα από την αξιοπιστία του Arduino.
- 3.1.2 *Απόκριση και ταχύτητα εκτέλεσης*. Ο χρόνος απόκρισης είναι εντός των απαιτήσεων όσον αφορά την απόκριση την εντολή, ο συνολικός χρόνος για το άνοιγμα/κλείσιμο θυρίδας εξαρτάται από τον υπάρχοντα μηχανισμό στο αστεροσκοπείο. Η απομακρυσμένη εντολή για την περιστροφή του θόλου εκτελείται μέσα σε διάστημα 5 δευτερολέπτων.
- 3.1.3 *Ασφάλεια*. Οι μηχανισμοί ανίχνευσης και διόρθωσης λαθών είναι ενσωματωμένοι στον πυρήνα του Arduino. Το Arduino Cloud εξασφαλίζει την πρόσβαση μόνο σε εξουσιοδοτημένους χρήστες. Τοπικά όμως, η εξασφάλιση πρόσβασης παρέχεται μόνο από την δυνατότητα ελέγχου της πρόσβασης στον φυσικό χώρο.
- 3.1.4 *Επεκτασιμότητα και Upgradability*. Όλα τα συστήματα Arduino είναι επεκτάσιμα και αναβαθμίσιμα.
- 3.1.5 *Ευχρηστία*. Το περιβάλλον χειρισμού είναι φιλικό προς τον χρήστη. Διαθέτει απομακρυσμένο έλεγχο.
- 3.1.6 *Συντήρηση*. Το σύστημα απαιτεί ελάχιστη συντήρηση.

10.2 Αξιολόγηση λειτουργικών απαιτήσεων

Οι βασικές λειτουργίες υλοποιήθηκαν, ωστόσο υπήρξαν συγκεκριμένες μηχανολογικές και προγραμματιστικές αποκλίσεις σε επίπεδο δικλίδων ασφαλείας.

- 3.2.1 *Βασικός Έλεγχος Θόλου*. Η περιστροφή υλοποιήθηκε αμφίδρομα (τοπικά και απομακρυσμένα). Η θέση του θόλου (0-29 βήματα) απεικονίζεται σε πραγματικό χρόνο στο απομακρυσμένο UI (positionDome). Η κίνηση σταματά αυτόματα όταν αφηθεί το πλήκτρο, χάρη στην ενημέρωση των μεταβλητών turnLeft και turnRight ως push-buttons.
- 3.2.2 *Αυτόματος συγχρονισμός θέσης θόλου με το τηλεσκόπιο*. Υλοποιήθηκε μέσω του movingSync state και του έξυπνου αλγορίθμου Syncing(), ο οποίος υπολογίζει τη συντομότερη διαδρομή στον κυκλικό άξονα.
- 3.2.3 *Έλεγχος θυρίδας*. Ενώ υπήρχε η αρχική απαίτηση η θυρίδα να μπορεί να παραμείνει σε ενδιάμεση θέση, τελικά ο κώδικας σχεδιάστηκε να οδηγεί τη θυρίδα στις τερματικές θέσεις (opened ή closed) χρησιμοποιώντας τους αισθητήρες Hall. Η παραμονή του κλείστρου ενός αστεροσκοπείου σε «μισάνοιχτη» κατάσταση ενέχει κινδύνους (π.χ. απότομες ριπές ανέμου). Έτσι, επιλέχθηκε προγραμματιστικά η δυαδική λογική (πλήρως ανοιχτό / πλήρως κλειστό) για μέγιστη ασφάλεια.
- 3.2.4 *Επιστροφή του θόλου στην αρχική του θέση*. Υλοποιήθηκε μόνο η δυνατότητα χειροκίνητης εντολής μετάβασης στην αρχική του θέση, ενώ η απαίτηση να κλείνει η θυρίδα και ο θόλος να επιστρέφει στην αρχική του θέση δεν υλοποιήθηκε.
- 3.2.5 *Λειτουργίες Ασφαλείας*. Σε περίπτωση βροχής, η θυρίδα κλείνει αυτόματα και ο χρήστης ενημερώνεται μέσα από το dashboard.
 - Σε περίπτωση διακοπής της επικοινωνίας ή αδράνειας για κάποιο ορισμένο χρόνο, θα πρέπει να επιστρέφει ο θόλος στην αρχική του θέση και η θυρίδα να κλείνει: Η υλοποίηση μηχανισμού Watchdog Timer (έλεγχος ping/αδράνειας) απαιτούσε συνεχή δέσμευση πόρων του μικροελεγκτή. Εφόσον το σύστημα ενσωματώνει αισθητήρα βροχής (ο οποίος λειτουργεί ανεξάρτητα από το δίκτυο), κρίθηκε ότι ο κίνδυνος έκθεσης του τηλεσκοπίου από απώλεια δικτύου καλύπτεται από το hardware.

- Σε περίπτωση που ο χρόνος περιστροφής υπερβεί ένα μέγιστο ή ολοκληρωθεί ένας πλήρης κύκλος, θα πρέπει να γίνεται αυτόματη διακοπή της περιστροφής και ενημέρωση του χρήστη. Σε περίπτωση που κατά την περιστροφή δεν μεταβάλλεται η ένδειξη του encoder, θα πρέπει να γίνεται αυτόματη διακοπή της περιστροφής και ενημέρωση του χρήστη. Θα πρέπει να υπάρχει ένας μέγιστος χρόνος αναμονής του σήματος θυρίδα τέρμα ανοιχτή/θυρίδα κλειστή κατά την εκτέλεση της αντίστοιχης εντολής ώστε αν η θυρίδα κινείται για μεγαλύτερο χρόνο, τότε η εντολή θα ακυρώνεται: Δεδομένου ότι το σύστημα αποτελεί μοντέλο προσομοίωσης χρησιμοποιώντας τους ελαφρείς μηχανισμούς της κάμερας Tapo C210, η πιθανότητα καταστροφικής μηχανικής εμπλοκής (stall) είναι μηδαμινή, σε αντίθεση με ένα πραγματικό αστεροσκοπείο πολλών τόνων. Η ενσωμάτωση χρονόμετρων ασφαλείας (timeout flags) για τα βήματα του Stepper.h θα προσέθετε περιττή πολυπλοκότητα στη Μηχανή Καταστάσεων του τοπικού συστήματος. Το συγκεκριμένο χαρακτηριστικό προτείνεται ως προσθήκη στις «Μελλοντικές Επεκτάσεις» της παρούσας εργασίας.

10.3 Λειτουργική Αξιολόγηση υλικού και τεχνολογιών

Για την επιλογή του υλικού λήφθηκαν υπόψη τα ακόλουθα κριτήρια:

1. Χαμηλό κόστος.
2. Απλή κατασκευή.
3. Αξιοπιστία λειτουργίας.
4. Συμβατότητα με τεχνολογίες IoT και embedded systems.
5. Ευκολία αναπαραγωγής και χρήσης.
6. Δυνατότητα επέκτασης και συντήρησης.

Η προτεινόμενη αρχιτεκτονική αξιολογείται ικανοποιητική για τις απαιτήσεις ενός εκπαιδευτικού αστεροσκοπείου, προσφέροντας αποδεκτή ισορροπία μεταξύ κόστους, λειτουργικότητας και επεκτασιμότητας.

Παράλληλα, με κάποιες επεκτάσεις-βελτιώσεις θα μπορούσε να χρησιμοποιηθεί και σε πραγματικό περιβάλλον.

10.3.1 Κεντρική μονάδα ελέγχου:

Η επιλογή της κεντρικής μονάδας είναι το πιο σημαντικό στάδιο για ένα IoT σύστημα. Είναι αυτό που πρέπει να αποφασισθεί αρχικά προτού προχωρήσουμε στην επιλογή των περιφερειακών. Υποψήφιες πλατφόρμες μεταξύ άλλων ήταν:

- **Raspberry Pi:** με πολύ μεγάλη υπολογιστική ισχύ, μεγάλο πλήθος εγκατεστημένης βάσης και εκτεταμένη τεκμηρίωση. Αποκλείστηκε όμως εξ αρχής καθώς παρότι προσφέρει πολύ μεγαλύτερη επεξεργαστική ισχύ, βασίζεται σε Λειτουργικό Σύστημα (Linux). Αυτό όμως θα πρόσθετε αυξημένη πολυπλοκότητα, μεγάλη κατανάλωση ενέργειας, και γενικά δεν προτείνεται για μη εποπτευόμενες εφαρμογές πραγματικού χρόνου και κατά συνέπεια είναι περισσότερο κατάλληλο για εφαρμογές καθοδηγούμενων από τον χρήστη. Επιπλέον, δεν είναι η καλύτερη επιλογή για την άμεση διασύνδεση με πλήθος αισθητήρων και ελέγχου ισχύος. Θα μπορούσε βέβαια να αποτελεί τον ενδιάμεσο ελεγκτή ο οποίος θα αναλάμβανε την επικοινωνία με το cloud όμως αυτό θα αύξανε το κόστος.
- **ESP32:** Παρότι ο μικροελεγκτής ESP32 αποτελεί μια εξαιρετικά δημοφιλή και οικονομική λύση για εφαρμογές IoT, δεν επιλέχθηκε κυρίως επειδή λειτουργεί με 3.3V με συνέπεια να είναι ευάλωτο σε βιομηχανικά περιβάλλοντα τα οποία είναι γεμάτα από θορύβους που προέρχονται από κινητήρες και ρελέ.
- **Arduino Uno R3:** Ο παλαιότερος και δοκιμασμένος R3 είναι αρχιτεκτονικής 8-bit, με πολύ περιορισμένη μνήμη SRAM. Η ταυτόχρονη διαχείριση του πρωτοκόλλου MQTT (για την επικοινωνία με το cloud) θα υπερφόρτωνε τη μνήμη, προκαλώντας αστάθεια. Επίσης δεν διαθέτει το απαραίτητο hardware WiFi για τη σύνδεση στο δίκτυο.
- **Ανάπτυξη in house embedded συστήματος** το οποίο θα σχεδιαζόταν σε λευκό χαρτί με επιλογή του κατάλληλου μικροελεγκτή καθώς και των περιφερειακών ολοκληρωμένων υποστήριξης. Αυτό όμως προϋποθέτει το σχεδιασμό του τυπωμένου κυκλώματος, μεγάλο χρόνο κατασκευής software, αυξημένο κόστος, λύση που προτείνεται κυρίως για επαγγελματικές και εμπορικές εφαρμογές και όχι τόσο για την εκπόνηση πτυχιακής.

- **Arduino Uno R4 WiFi:** Η Arduino με την πρόσφατη διάθεση του Arduino Uno R4 WiFi έκανε ένα άλμα προς τα εμπρός διευκολύνοντας πολύ την ανάπτυξη IoT εφαρμογών. Η πλακέτα διαθέτει ενσωματωμένο το ESP32-S3 module, το οποίο λειτουργεί αποκλειστικά ως συνεπεξεργαστής δικτύου. Το ESP32-S3 αναλαμβάνει όλο το «βάρος» της διαχείρισης της στοίβας δικτύου (TCP/IP, MQTT) και της κρυπτογράφησης (SSL/TLS). Ο μικροελεγκτής του Arduino Uno R4 ο οποίος σε αντίθεση με το Arduino Uno R3 είναι αρχιτεκτονικής 32 bit και σε συνδυασμό με το γρηγορότερο ρολόι αυξάνει τις επιδόσεις και τον αφήνει πλήρως ελεύθερο να διαχειρίζεται αποκλειστικά τον βρόχο ελέγχου (control loop) του υλικού. Έτσι, διασφαλίζεται ότι τα κρίσιμα ως προς τον χρόνο σήματα προς τους βηματικούς κινητήρες δεν διακόπτονται από απρόβλεπτες καθυστερήσεις ή διακοπές (interrupts) της σύνδεσης WiFi, αποτρέποντας την απώλεια βημάτων κατά την περιστροφή του θόλου. Επιπλέον, σε αντίθεση με τον κλασικό Uno (8-bit), ο επεξεργαστής Renesas RA4M1 (Cortex®-M4) επιτρέπει την ταχύτερη επεξεργασία των δεδομένων και την εκτέλεση του αλγορίθμου συντομότερης διαδρομής χωρίς καθυστερήσεις.

Σε αντίθεση με ένα σύστημα που διαθέτει Linux όπως το Raspberry Pi, ο RA4M1 εκτελεί αποκλειστικά τον μεταγλωττισμένο κώδικα σε ατέρμονα βρόχο, προσφέροντας ασύγκριτα μεγαλύτερη σταθερότητα για συνεχή απομακρυσμένη λειτουργία.

Το σύστημα λειτουργεί σε λογική στάθμη 5V (σε αντίθεση με τα 3.3V του ESP32). Αυτό επιτρέπει την άμεση και ασφαλή οδήγηση βιομηχανικών ρελέ ισχύος και την ανάγνωση των αισθητήρων, χωρίς την ανάγκη παρεμβολής κυκλωμάτων μετατροπής τάσης, τα οποία θα αύξαναν τα πιθανά σημεία βλάβης του συστήματος.

10.3.2 Οικοσύστημα Modulino (I²C):

Η προσθήκη του διαύλου Qwiic/I²C [11] στο Arduino Uno έδωσε νέες δυνατότητες διασύνδεσης αισθητήρων και ενεργοποιητών. Με μόλις 4 αγωγούς μπορούμε πλέον να συνδέσουμε πλήθος αισθητήρων και εξόδων και ουσιαστικά να επεκτείνουμε πολύ εύκολα

το περιορισμένο από I/O Arduino και μάλιστα χωρίς το μειονέκτημα της αυξημένης χρήσης καλωδίων.

Πρόσφατα η Arduino διέθεσε έτοιμα modules που συνδέονται σε αυτή τη σειριακή θύρα με αποτέλεσμα να υπάρχει και μεγάλη μείωση στο πλήθος των απαραίτητων συνδέσεων. Τα συμβατά modules τα ονομάζει Modulino «κάτι» ανάλογα με το σκοπό που εξυπηρετούν.

Η ενσωμάτωση των συγκεκριμένων μονάδων προσέφερε την καλύτερη λύση για τις ανάγκες του πρωτότυπου στο οποίο χρησιμοποιήθηκαν δυο «Modulino Knobs» το καθένα εκ των οποίων ενσωματώνει ένα rotary encoder με ένα ενσωματωμένο button, και δυο «Modulino Buttons» με το καθένα από αυτά να διαθέτει τρία buttons και τρία leds.

Συγκριτικά με τις άλλες υπάρχουσες τεχνολογίες έχουμε:

- Εάν χρησιμοποιούσαμε παραδοσιακά encoders και buttons, το κάθε ένα θα απαιτούσε τουλάχιστον 3 αγωγούς, δημιουργώντας ένα σύνολο 14 συνδέσεων το οποίο θα ήταν ευάλωτο σε αποσυνδέσεις και σε ηλεκτρομαγνητικές παρεμβολές από τους κινητήρες και το περιβάλλον.
- Το SPI είναι μεν ταχύτερο όμως προορίζεται για επικοινωνία on board. Αντίθετα, το I²C επιτρέπει την τοποθέτηση πολλαπλών αισθητήρων χρησιμοποιώντας μόνο 4 αγωγούς (VCC, GND, SDA, SCL) συνολικά, κάνοντας το σύστημα εξαιρετικά τακτοποιημένο και επεκτάσιμο και επιπλέον επιτρέπει την επικοινωνία μεταξύ διαφορετικών πλακετών διατηρώντας μικρές αποστάσεις.

10.3.3 Γιατί Rotary Encoder και όχι Γυροσκόπιο:

Μία από τις δυσκολίες του έργου ήταν η καταγραφή της γωνίας περιστροφής του θόλου και του τηλεσκοπίου. Στην πρώτη φάση της υλοποίησης (Πρωτότυπο V1), χρησιμοποιήθηκαν γυροσκόπια (IMU sensors), όμως γρήγορα ήρθαν στην επιφάνεια τα μειονεκτήματά τους και αυτό οδήγησε την αντικατάστασή τους από τους Rotary Encoders.

Αρχικά, το γυροσκόπιο θεωρήθηκε ιδανικό καθώς δεν απαιτούσε μηχανική σύνδεση με τους άξονες περιστροφής (εύκολη τοποθέτηση). Οι δοκιμές ωστόσο ανέδειξαν το φαινόμενο της διολίσθησης (Gyroscopic Drift). Με την πάροδο του χρόνου, τα μικρά σφάλματα στις μετρήσεις συσσωρεύονταν, με αποτέλεσμα η υπολογιζόμενη γωνία να αποκλίνει σημαντικά από την πραγματική. Για την επίλυση θα απαιτούνταν πολύπλοκοι αλγόριθμοι φιλτραρίσματος, οι οποίοι θα επιβάρυναν τον επεξεργαστή χωρίς να εγγυώνται την απόλυτη ακρίβεια που απαιτεί ένα αστεροσκοπείο.

Η υιοθέτηση του rotary encoder έλυσε τα παραπάνω προβλήματα, προσφέροντας:

- **Σχετική Ακρίβεια:** Ο encoder μετρά διακριτά βήματα (clicks), κάτι που σημαίνει ότι η θέση είναι πάντα γνωστή και δεν μεταβάλλεται με τον χρόνο. Η θέση του μηδενίζεται από το λογισμικό όταν πιέζεται ο άξονας με το ενσωματωμένο κουμπί που διαθέτει.
- **Αξιοπιστία:** Κάθε μεταβολή στη γωνία μεταφράζεται σε ψηφιακό σήμα, εξαλείφοντας τον θόρυβο που επηρέαζε τις αναλογικές μετρήσεις του γυροσκοπίου.

Συμπερασματικά, η επιλογή του rotary encoder, κρίθηκε ως καλύτερη λύση.

10.3.4 Αισθητήρες Hall:

Για την ανίχνευση της θέσης "Home" επιλέχθηκαν αισθητήρες Hall έναντι των κλασικών μηχανικών διακοπών (Limit Switches) ή των οπτικών αισθητήρων:

- Γιατί δεν επιλέχθηκε μηχανικός διακόπτης: Οι μηχανικοί διακόπτες έχουν μεταλλικά μέρη όπως ελατήρια, και κινητά μέρη τα οποία οξειδώνονται από την υγρασία και φθείρονται μηχανικά με τον χρόνο. Ο αισθητήρας Hall, επειδή λειτουργεί με την ανίχνευση μαγνητικού πεδίου χωρίς καμία φυσική επαφή (contactless), έχει πρακτικά απεριόριστη διάρκεια ζωής.

Κατά τις δοκιμές οι αισθητήρες Hall αντιδρούσαν άμεσα στο μαγνητικό πεδίο. Σε ένα αστεροσκοπείο, η χρήση βιομηχανικών αισθητήρων Hall θα επιτρέψει τη λειτουργία ακόμα και κάτω από συνθήκες παγετού, όπου ένας μηχανικός διακόπτης θα μπορούσε να «κολλήσει».

- Γιατί δεν επιλέχθηκε οπτικός αισθητήρας: Οι οπτικοί αισθητήρες ενώ προσφέρουν μεγάλη ακρίβεια, στο περιβάλλον ενός ημι-ανοικτού θόλου επηρεάζονται από τη συσσώρευση σκόνης ή ακόμα και του πάγου πάνω στον φακό τους με αποτέλεσμα τη μη σωστή λειτουργία. Το μαγνητικό πεδίο που χρησιμοποιείται στον αισθητήρα Hall πλεονεκτεί καθώς διαπερνά αυτά τα εμπόδια ανεπηρέαστο.

10.3.5 Βηματικοί Κινητήρες (Stepper Motors):

Για τις ανάγκες του μοντέλου, ο έλεγχος της κίνησης ανατέθηκε σε βηματικούς κινητήρες. Η αξιολόγησή τους έδειξε:

- Ακρίβεια Γωνιακής Μετατόπισης: Η απόκλιση της γωνίας του θόλου σε σχέση με αυτήν του τηλεσκοπίου ήταν μικρότερη από 2° .
- Ροπή Συγκράτησης: Όταν δεν λειτουργεί ο κινητήρας, ο μειωτήρας του παρέχει την απαραίτητη ροπή ώστε ο θόλος να παραμένει σταθερός στη θέση του, λειτουργώντας ως «φρένο» χωρίς την κατανάλωση επιπλέον ενέργειας.
- Γιατί δεν επιλέχθηκε κινητήρας DC: Για να ελεγχθεί η θέση ενός κινητήρα DC με μειωτήρα, θα χρειαζόταν η προσθήκη ενός πολύπλοκου κλειστού βρόχου ελέγχου και οπτικών encoders. Οι βηματικοί κινητήρες, αντίθετα, μετακινούνται κατά μια σταθερή, γνωστή γωνία σε κάθε παλμό ρεύματος. Αυτό επέτρεψε στο λογισμικό ελέγχου τον συγχρονισμό της γωνίας, εξασφαλίζοντας ακρίβεια χωρίς την ανάγκη εξαρτημάτων ανατροφοδότησης.
- Γιατί δεν επιλέχθηκε servomotor: Παρόλο που τα servomotors προσφέρουν εξαιρετική ακρίβεια θέσης, τα περισσότερα έχουν φυσικό περιορισμό περιστροφής στις 180 μοίρες. Ο θόλος απαιτεί πλήρη περιστροφή 360° , καθιστώντας τους βηματικούς κινητήρες τη μόνη ρεαλιστική επιλογή για το πρωτότυπο.

10.3.6 Αισθητήρας Βροχής

Η προστασία του τηλεσκοπίου και των ηλεκτρονικών συστημάτων από βροχοπτώσεις είναι λειτουργική απαίτηση του έργου. Για τον σκοπό αυτό, ενσωματώθηκε ένας αισθητήρας ανίχνευσης νερού, η επιλογή του οποίου αξιολογείται παρακάτω σε σχέση με άλλες διαθέσιμες τεχνολογίες.

Κατά τον σχεδιασμό, εξετάστηκε το ενδεχόμενο να αξιοποιηθεί η σύνδεση του Arduino Uno R4 στο internet για τη λήψη δεδομένων καιρού σε πραγματικό χρόνο από διαδικτυακές υπηρεσίες. Η λύση αυτή εγκαταλείφθηκε αμέσως για δύο λόγους:

- Έλλειψη Ακρίβειας Μικροκλίματος: Τα APIs παρέχουν προβλέψεις με βάση περιφερειακούς μετεωρολογικούς σταθμούς. Μια ξαφνική τοπική μπόρα πάνω από

το αστεροσκοπείο θα μπορούσε να ξεκινήσει πριν ενημερωθεί η διαδικτυακή βάση δεδομένων.

- Εξάρτηση από το Δίκτυο: Η προσέγγιση μέσω API προϋποθέτει αδιάλειπτη σύνδεση στο διαδίκτυο. Εάν ο router τεθεί εκτός λειτουργίας, το αστεροσκοπείο μένει χωρίς ενημέρωση για τον καιρό. Ο αισθητήρας όμως συνδέεται ενσύρματα στον μικροελεγκτή, εξασφαλίζοντας άμεση απόκριση ανεξάρτητα από την κατάσταση του δικτύου.

Στο πρωτότυπο χρησιμοποιήθηκε ένας αισθητήρας τύπου πλέγματος (Resistive Rain Sensor), ο οποίος ανιχνεύει τη βροχή μέσω της μεταβολής της αγωγιμότητας όταν το νερό έρθει σε επαφή με τους αγωγούς του.

- Η λειτουργία του ήταν άμεση και συνεργάστηκε πολύ καλά με τη non-blocking αρχιτεκτονική του κώδικα, λειτουργώντας ως ασύγχρονη διακοπή και προκαλούσε το State Machine να μεταβεί σε κατάσταση "Emergency Close".
- Η αξιολόγησή του όμως ανέδειξε και έναν περιορισμό: οι αισθητήρες αγωγιμότητας επειδή είναι από χαλκό τείνουν να οξειδώνονται λόγω της υγρασίας. Κατά την προσαρμογή του συστήματος στο αστεροσκοπείο, προτείνεται η αντικατάστασή του από έναν χωρητικό ή οπτικό αισθητήρα βροχής βιομηχανικού τύπου (όπως αυτοί που χρησιμοποιούνται στους υαλοκαθαριστήρες οχημάτων). Οι αισθητήρες αυτοί δεν διαθέτουν μεταλλικές επαφές και έτσι είναι προστατευμένοι από οξείδωση και απαλλαγμένοι από ανάγκη συντήρησης.

10.3.7 Αντιμετώπιση Προβλημάτων και Επανασχεδιασμός

Κατά την ανάπτυξη του συστήματος προέκυψαν προβλήματα συγχρονισμού, χρονισμού εντολών και σταθερότητας επικοινωνίας, τα οποία οδήγησαν σε επανασχεδιασμό βασικών τμημάτων του λογισμικού.

Στην αρχή είχε χρησιμοποιηθεί blocking λογική μέσω καθυστερήσεων (delay), όμως γρήγορα διαπιστώθηκε ότι η προσέγγιση αυτή δημιουργούσε μεγάλη καθυστέρηση σε βασικά γεγονότα, όπως στην επικοινωνία με το διαδίκτυο, στον έλεγχο των κινητήρων και στην επικοινωνία με τους αισθητήρες.

Για τον λόγο αυτό επιλέχθηκε non-blocking αρχιτεκτονική βασισμένη σε finite state machines (FSM) και σε διαμοιρασμό του χρόνου. Η αλλαγή αυτή βελτίωσε σημαντικά:

- την απόκριση του συστήματος,
- την «ταυτόχρονη» εκτέλεση πολλαπλών λειτουργιών,
- τη σταθερότερη λειτουργία των κινητήρων,
- τις λειτουργίες ασφαλείας.

Ο επανασχεδιασμός προσέφερε σημαντική βοήθεια στην εξέλιξη της εργασίας καθώς μετέτρεψε το σύστημα από μια απλή υλοποίηση σε ένα αξιόπιστο και επεκτάσιμο ενσωματωμένο σύστημα ελέγχου.

Ανεστραμμένη τοποθέτηση encoder: Για τις ανάγκες του πρωτότυπου ήταν αναγκαίο ο κωδικοποιητής του θόλου να τοποθετηθεί ανάποδα ως προς τον άξονα περιστροφής του. Το αποτέλεσμα ήταν οι δεξιόστροφες περιστροφές να καταγράφονται ως αρνητική μετατόπιση. Η λύση δόθηκε μέσω του λογισμικού, αντιστρέφοντας το πρόσημο κατά την ανάγνωση, ευθυγραμμίζοντας έτσι τον υπολογισμό της θέσης του θόλου με αυτόν του τηλεσκοπίου.

Λογική λειτουργίας Θυρίδας: Ενώ η αρχική απαίτηση προέβλεπε η θυρίδα να μπορεί να ακινητοποιηθεί σε οποιαδήποτε ενδιάμεση θέση, στην πορεία η λειτουργία αυτή επανεκτιμήθηκε και επανασχεδιάστηκε. Εφαρμόστηκε δυαδική λογική (η θυρίδα μεταβαίνει μόνο σε κατάσταση "Πλήρως Ανοιχτή" ή "Πλήρως Κλειστή"). Αυτό έγινε για λόγους προστασίας από τον άνεμο, καθώς σε πραγματικές συνθήκες, μια ημιανοιχτή θυρίδα αστεροσκοπείου λειτουργεί ως «πανί» και θα είναι ευάλωτη σε ριπές ανέμου που θα μπορούσαν να προκαλέσουν μηχανικές βλάβες.

10.4 Ποιοτική και Ποσοτική Αξιολόγηση Λογισμικού

Η ανάπτυξη του κώδικα για τον έλεγχο του θόλου του αστεροσκοπείου δεν περιορίστηκε στη στοιχειώδη αποστολή εντολών προς τους κινητήρες, αλλά σχεδιάστηκε και με γνώμονα την αξιοπιστία. Σε αυτή την ενότητα αξιολογείται η λειτουργική απόδοση του λογισμικού, η ανθεκτικότητα της αρχιτεκτονικής του, καθώς και η συμπεριφορά του τηλεπικοινωνιακού πρωτοκόλλου.

10.4.1 Ποιοτική Αξιολόγηση Λογισμικού

Στα περισσότερα απλά συστήματα με Arduino γίνεται εκτεταμένη χρήση καθυστερήσεων τύπου `delay()` στο πρόγραμμα με αποτέλεσμα ο μικροελεγκτής να «παγώνει» και να μη μπορεί να διαβάσει άλλους αισθητήρες. Για παράδειγμα αν η περιστροφή του θόλου χρειαζόταν 10 δευτερόλεπτα και στο 2ο δευτερόλεπτο ξεκινούσε να βρέχει, το σύστημα δεν θα αντιδρούσε μέχρι να ολοκληρωθεί η κίνηση. Έτσι, η κλήση της συνάρτησης `ArduinoCloud.update()` σε ξεχωριστή κατάσταση (αντί μέσα στον κύριο βρόχο) ήταν καθοριστική για τη ανεξαρτησία της τοπικής λειτουργίας (π.χ. ασφάλεια θυρίδας από τον αισθητήρα βροχής) από τη διαθεσιμότητα του διαδικτύου.

Η λογική του προγράμματος διαχωρίστηκε σε διακριτές καταστάσεις (π.χ. `IDLE`, `MOVING_DOME`, `EMERGENCY_CLOSE...`) με αποτέλεσμα η κίνηση των βηματικών κινητήρων να εκτελείται σταδιακά με διαφανή όμως τρόπο χωρίς να μπλοκάρει το πρόγραμμα. Αυτή η προσέγγιση επιτρέπει στον βρόχο (`main loop`) να εκτελείται χιλιάδες φορές το δευτερόλεπτο. Ως εκ τούτου η ανάγνωση του αισθητήρα βροχής και των άλλων αισθητήρων δεν καθυστερεί και έτσι για παράδειγμα δεν θα υπάρχει καθυστέρηση στο κλείσιμο της θυρίδας σε περίπτωση βροχής την ώρα που το σύστημα είναι απασχολημένο.

10.4.2 Ποσοτική Αξιολόγηση Λογισμικού

Για την ποσοτική αξιολόγηση του λογισμικού διεξήχθησαν δοκιμές απόδοσης σε δύο άξονες: την ταχύτητα εκτέλεσης του βρόχου και την εξοικονόμηση κινήσεων μέσω του αλγορίθμου `Auto-Sync`.

Λόγω της αρχιτεκτονικής `non-blocking`, ο μέσος χρόνος ολοκλήρωσης ενός κύκλου ελέγχου μετρήθηκε σε λιγότερο από 5 χιλιοστά του δευτερολέπτου. Αυτό εγγυάται ότι οι βηματικοί κινητήρες λαμβάνουν τους παλμούς κίνησης στην ώρα τους, χωρίς έντονα φαινόμενα «jitter» (τρέμουλο) που προκαλούνται από καθυστερήσεις του λογισμικού.

Αποδοτικότητα του `Auto-Sync`: Ο αλγόριθμος "Συντομότερης Διαδρομής" που υλοποιήθηκε, αποδείχθηκε πολύ αποδοτικός. Εάν η απόλυτη διαφορά των θέσεων υπερβαίνει τη μέγιστη ημι-κυκλική απόσταση (15 βήματα), ο θόλος επιλέγει πάντα την συντομότερη διαδρομή (έστω και περνώντας από το σημείο μηδέν), μειώνοντας τον

συνολικό χρόνο περιστροφής και τη φθορά των κινητήρων κατά 50% στις χειρίστες περιπτώσεις. Στον παρακάτω πίνακα παρατίθενται επιλεγμένα σενάρια δοκιμών που επιβεβαιώνουν την ποσοτική υπεροχή του αλγορίθμου έναντι της απλής γραμμικής μετάβασης.

Πίνακας 6 - Ποσοτική αξιολόγηση εξοικονόμησης κινήσεων μέσω του αλγορίθμου Shortest Path

Σενάριο	Τρέχουσα Θέση Θόλου	Νέα Θέση Τηλεσκοπίου	Απλή Γραμμική Μετάβαση (Βήματα/Κατεύθυνση)	Μετάβαση με Αλγόριθμο Auto-Sync (Βήματα/Κατεύθυνση)	Ποσοστό Εξοικονόμησης Χρόνου/Κίνησης
1	2	10	8 (Δεξιόστροφα)	8 (Δεξιόστροφα)	0%
2	5	28	23 (Δεξιόστροφα)	7 (Αριστερόστροφα)	~70%
3	25	1	24 (Αριστερόστροφα)	6 (Δεξιόστροφα)	75%
4	0	15	15 (Δεξιόστροφα)	15 (Δεξιόστροφα)	0%

Όπως παρατηρείται στα Σενάρια 2 και 3, όταν το τηλεσκόπιο περνάει από το σημείο αναφοράς, ο αλγόριθμος μειώνει τον χρόνο κίνησης του θόλου έως και 75%.

Χρόνος Απόκρισης (Latency): Κατά τις δοκιμές από το απομακρυσμένο Dashboard (π.χ. Turn Left), οι εντολές εκτελούνταν στο Arduino σε λιγότερο από 2 δευτερόλεπτα. Αυτό υπερκαλύπτει την μη λειτουργική απαίτηση (Ενότητα 3.1.2) που όριζε τον μέγιστο χρόνο έναρξης εκτέλεσης εντολής στα 5 δευτερόλεπτα.

10.4.3 Επικοινωνία

Η απομακρυσμένη επικοινωνία του Arduino Uno R4 WiFi με το Arduino Cloud αποτελεί ίσως την πιο σημαντική λειτουργία σε επίπεδο τηλεμετρίας.

Το Arduino Cloud κάνει χρήση του πρωτοκόλλου MQTT (Message Queuing Telemetry Transport). Σε αντίθεση με το κλασικό HTTP (όπου ο client πρέπει συνεχώς να "ρωτάει"

τον server αν υπάρχει νέα εντολή - polling), το MQTT βασίζεται στο μοντέλο Publish/Subscribe.

- **Ποιοτικό πλεονέκτημα:** Ο μικροελεγκτής δεν σπαταλά υπολογιστικούς πόρους ελέγχοντας το δίκτυο. Μόλις ο χρήστης πατήσει ένα κουμπί στο Dashboard (π.χ. "Άνοιξε τη Θυρίδα"), ο server προωθεί άμεσα (push) την αλλαγή στη μεταβλητή.
- **Ποσοτικό πλεονέκτημα (Bandwidth):** Τα πακέτα δεδομένων του MQTT είναι εξαιρετικά μικρά (λίγα bytes). Αυτό καθιστά το σύστημα ιδανικό για αστεροσκοπεία σε απομακρυσμένες περιοχές, όπου η σύνδεση στο διαδίκτυο μπορεί να γίνει μέσω παλαιότερων δικτύων κινητής τηλεφωνίας (4G/LTE) ή ακόμα και όταν το σήμα είναι χαμηλό.

Η βιβλιοθήκη IoT του Arduino διαχειρίζεται με διαφανή τρόπο τα σφάλματα δικτύου και είναι σχεδιασμένη ώστε στην περίπτωση προβλημάτων να μην επηρεάζεται το κυρίως πρόγραμμα από αυτά. Εάν χαθεί η σύνδεση δικτύου για οποιοδήποτε λόγο, η συνάρτηση `ArduinoCloud.update()` διαχειρίζεται αυτόματα τη διακοπή χωρίς να επηρεάζεται η τοπική λειτουργία πράγμα που σημαίνει ότι ο έλεγχος του θόλου και η ασφάλεια της θυρίδας μπορούν να εξακολουθούν να ελέγχονται τοπικά (χειροκίνητη λειτουργία μέσω Modulino Knob), μέχρις ότου να αποκατασταθεί η σύνδεση.

10.5 Αξιολόγηση Ενημέρωσης Κατάστασης Συστήματος (Visualization)

Η δυνατότητα παρουσίασης των δεδομένων που μας παρέχει το Arduino Cloud Dashboard δεν αποτελεί απλώς ένα εργαλείο ελέγχου, αλλά το βασικό μέσο επικοινωνίας μεταξύ του απομακρυσμένου χρήστη και της τοπικής εγκατάστασης. Η αξιολόγηση πραγματοποιείται με βάση την πληροφορία, την ταχύτητα ενημέρωσης και τη χρηστικότητα.

Ταχύτητα Ενημέρωσης

Ένα από τα βασικά ζητούμενα ήταν η διαρκής ενημέρωση του χρήστη για την τρέχουσα λειτουργία και την κατάσταση του συστήματος.

Για παράδειγμα, τα Virtual LEDs "Door Opened", "Door Closed" παρέχουν οπτική ένδειξη σε πραγματικό χρόνο για την κατάσταση της θυρίδας. Το λογισμικό ενημερώνει πολύ γρήγορα τις μεταβλητές στο Cloud κάθε αλλαγή που συμβαίνει στους αισθητήρες.

Η καθυστέρηση (latency) μεταξύ της αλλαγής κατάστασης των αισθητήρων και της απομακρυσμένης οπτικής ένδειξης μετρήθηκε σε επίπεδα μικρότερα των 3 δευτερολέπτων, γεγονός που καθιστά το visualization αρκετά αξιόπιστο.

Σαφήνεια και Διασθητικότητα

Ανάλογα με το είδος της πληροφορίας χρησιμοποιήθηκαν διαφορετικά είδη "widgets" στο Dashboard.

Οι αριθμητικές ενδείξεις των encoders οι οποίες αντιστοιχούν στην τρέχουσα θέση τους επιτρέπουν στον χρήστη να γνωρίζει τη σχετική θέση του θόλου ως προς το τηλεσκόπιο χωρίς να χρειάζεται να υπολογίσει αριθμητικά δεδομένα μοιρών. Όταν έχουμε ταύτιση αυτών των ενδείξεων τότε έχουμε και ταύτιση της γωνίας τους ως προς την αρχική τους θέση.

Το χρώμα της κάθε φωτεινής ένδειξης αποδόθηκε ανάλογα με τη βαρύτητα του συμβάντος για παράδειγμα το κόκκινο χρώμα υποδηλώνει κατάσταση κινδύνου ή έκτακτης ανάγκης, διευκολύνοντας την άμεση αντίδραση του χρήστη.

Συνοχή Δεδομένων

Ένα άλλο σημείο που αξιολογήθηκε είναι η δυνατότητα που παρέχεται στο χρήστη να παρακολουθεί το συγχρονισμό μεταξύ της θέσης του τηλεσκοπίου και της θέσης του θόλου στην οθόνη. Με την ενεργοποίηση της λειτουργίας του "Auto-Sync", το Dashboard εμφανίζει ταυτόχρονα και τις δύο τιμές και αυτή η δυνατότητα του να μπορεί ο χρήστης να βλέπει στην ίδια οθόνη ότι η τιμή "Dome Position" τείνει να εξισωθεί με την τιμή "Telescope Position" παρέχει μια αίσθηση ασφάλειας ότι ο αυτοματισμός λειτουργεί με το σωστό τρόπο.

Όπως αναφέρθηκε στην περιγραφή του λογισμικού, η αντιστροφή της τιμής του encoder ήταν απαραίτητη ώστε η ένδειξη στο Dashboard να είναι σε συγχρονισμό με τη φυσική κίνηση (δεξιόστροφα/αριστερόστροφα) και έτσι να μη μπερδεύει το χρήστη.

Προσβασιμότητα και Φορητότητα

Το visualization αξιολογήθηκε τόσο σε περιβάλλον σταθερού υπολογιστή όσο και σε περιβάλλον κινητού (Arduino IoT Cloud App).

Το Arduino Cloud παρέχει την δυνατότητα εποπτείας και από κινητές συσκευές. Σε αυτή την περίπτωση τα widgets προσαρμόζονται αυτόματα στην οθόνη του κινητού και αυτό

εξασφαλίζει ότι ο χρήστης μπορεί να επιβλέπει/ελέγχει το θόλο ακόμη και εν κινήσει. Είναι μια σημαντική δυνατότητα για την επίτευξη του στόχου της «απομακρυσμένης διαχείρισης».

Διάγνωση σύνδεσης στο δίκτυο

Το LED Matrix πάνω στο Arduino Uno R4 WiFi χρησιμοποιείται από τη βιβλιοθήκη για την ένδειξη της σύνδεσης στο WiFi και επιτρέπει την άμεση διάγνωση δικτυακών προβλημάτων χωρίς να απαιτείται η σύνδεση υπολογιστή.

10.6 Επίτευξη Στόχων

Σε αυτό το τμήμα εξετάζεται κατά πόσον το τελικό σύστημα ικανοποιεί τους στόχους που τέθηκαν κατά την έναρξη της πτυχιακής εργασίας.

Το σύστημα που αναπτύχθηκε αποτελεί μια λύση τηλεμετρίας και ελέγχου, η οποία βασίζεται στο Arduino Uno R4 WiFi και στο Arduino IoT Cloud. Η κατασκευή του μοντέλου του θόλου κάνοντας χρήση τμήματα της κάμερας Tapo C210 επέτρεψε τον έλεγχο των αλγορίθμων και της σταθερότητας του λογισμικού.

- **Τοπικός έλεγχος:** Με τη χρήση των τοπικών χειριστηρίων και ενδείξεων ο χρήστης έχει τον έλεγχο της περιστροφής του θόλου και της θυρίδας τοπικά.
- **Απομακρυσμένος Έλεγχος:** Μέσω του Arduino Cloud Dashboard, ο χρήστης έχει τον έλεγχο της περιστροφής του θόλου και της θυρίδας από μια απομακρυσμένη συσκευή με πρόσβαση στο διαδίκτυο. Η χρήση του πρωτοκόλλου MQTT εξασφαλίζει υψηλή ταχύτητα, υψηλή αξιοπιστία και χαμηλή κατανάλωση δεδομένων.
- **Αυτόματος Συγχρονισμός - Auto Sync:** Ο αλγόριθμος "Shortest Path" που αναπτύχθηκε επιτρέπει στον θόλο να ακολουθεί το τηλεσκόπιο με τη μέγιστη δυνατή αποδοτικότητα, μειώνοντας τις περιττές κινήσεις και τον χρόνο αναμονής.
- **Ασφάλεια και Προστασία - Rain Sensor:** Η ενσωμάτωση του αισθητήρα βροχής διασφαλίζει ότι ο θόλος θα κλείσει αυτόματα σε περίπτωση κακοκαιρίας, προστατεύοντας τον εξοπλισμό ακόμη και αν ο χρήστης δεν είναι παρών.

- **Χαμηλό Κόστος:** Το κόστος των υλικών της κατασκευής του πρωτοτύπου παρέμεινε σε πολύ χαμηλά επίπεδα σε σχέση με τις εμπορικές λύσεις, αποδεικνύοντας ότι οι τεχνολογίες Open Source μπορούν να προσφέρουν γρήγορα αποτελέσματα με ελάχιστο κόστος.

10.7 Μελλοντικές Προεκτάσεις

Η παρούσα εργασία θέτει τις βάσεις για ένα σύστημα που μπορεί να εξελιχθεί περαιτέρω με νέες λειτουργίες και βελτιώσεις.

Τεχνικές και Λειτουργικές Βελτιώσεις

Οπτική παρακολούθηση: Η αξιολόγηση του συστήματος έδειξε ότι, παρά την αξιοπιστία των αισθητήρων και του dashboard, η πλήρης απομακρυσμένη διαχείριση ενός αστεροσκοπείου απαιτεί την ενσωμάτωση **συστημάτων οπτικής επιτήρησης**. Η ανάγκη αυτή χωρίζεται σε δύο επίπεδα:

1. **Επιτήρηση του χώρου και των εγκαταστάσεων:** Είναι απαραίτητη η χρήση μιας κάμερας τοποθετημένης εντός του θόλου ώστε να επιτρέπει στον χρήστη να ελέγχει ότι ο θόλος περιστρέφεται κανονικά, η θυρίδα έχει ανοίξει/κλείσει πλήρως και ότι δεν υπάρχουν εμπλοκές στα καλώδια του τηλεσκοπίου κατά την περιστροφή. Στο πρωτότυπο, η παρακολούθηση γίνεται μέσω εφαρμογής της κάμερας (Tapo App). Μια ακόμη βελτίωση θα ήταν η ενσωμάτωση του stream της κάμερας απευθείας μέσα στο Arduino Cloud Dashboard, ώστε ο χρήστης να έχει στην ίδια οθόνη τόσο τα τηλεμετρικά δεδομένα όσο και την εικόνα από το εσωτερικό του θόλου.
2. **Οπτική Επιβεβαίωση Στόχου:** Εκτός από την επίβλεψη του χώρου, ο χρήστης χρειάζεται να βλέπει επίσης τι βλέπει το ίδιο το τηλεσκόπιο. Αυτό επιτυγχάνεται μέσα από την ξεχωριστή εφαρμογή του τηλεσκοπίου, όμως θα διευκόλυνε πολύ αν μπορούσε να ενσωματωθεί σε ένα παράθυρο μέσα στο dashboard του Arduino Cloud.
3. **Ιστορικό συμβάντων:** Το σύστημα μπορεί να επεκταθεί με δυνατότητα καταγραφής ιστορικού συμβάντων (event logging) μέσω του οποίου θα αποθηκεύονται σημαντικές λειτουργίες και καταστάσεις του θόλου. Η καταγραφή

αυτή θα διευκολύνει τη διάγνωση προβλημάτων, την παρακολούθηση της λειτουργίας του συστήματος και τη συντήρηση του εξοπλισμού.

4. **Αποστολή ειδοποιήσεων:** Θα μπορούσε ακόμη να ενσωματωθεί σύστημα αποστολής ειδοποιήσεων προς τον χρήστη μέσω email, push notifications ή εφαρμογών ανταλλαγής μηνυμάτων. Με αυτόν τον τρόπο, ο χειριστής θα μπορεί να ενημερώνεται άμεσα για κρίσιμα συμβάντα, όπως αυτόματο κλείσιμο της θυρίδας λόγω βροχής, απώλεια επικοινωνίας, υπέρβαση χρόνου περιστροφής ή εμφάνιση σφάλματος αισθητήρων. Η δυνατότητα αυτή αυξάνει σημαντικά την ασφάλεια και την αξιοπιστία της απομακρυσμένης λειτουργίας του αστεροσκοπείου.

Υποστήριξη ASCOM ή/και INDI: Η ανάπτυξη ειδικών drivers ώστε το σύστημα να είναι συμβατό με τα πρωτόκολλα ASCOM ή/και INDI, θα επιτρέψει τον πλήρη έλεγχο του θόλου μέσα από επαγγελματικά λογισμικά αστρονομίας, καθιστώντας τον συγχρονισμό εντελώς αυτόματο με την κίνηση του τηλεσκοπίου.

Ενίσχυση Ασφάλειας και Fail-safe Μηχανισμών

- Η προσθήκη ενός ανεξάρτητου κυκλώματος που θα κλείνει αυτόματα τη θυρίδα σε περίπτωση σημαντικής αστοχίας του λογισμικού ή απώλειας ρεύματος (μέσω UPS), θα προσφέρει υψηλό επίπεδο προστασίας.
- Η προσθήκη αισθητήρα ανέμου θα επιτρέψει στο σύστημα να κλείνει τη θυρίδα όχι μόνο σε περίπτωση βροχής, αλλά και όταν οι ριπές του ανέμου υπερβαίνουν ένα συγκεκριμένο όριο ασφαλείας, προστατεύοντας τη μηχανική ευστάθεια του θόλου.
- **Ανίχνευση Εμπλοκής:** Ενσωμάτωση δικλίδων ασφαλείας στον κώδικα, ώστε εάν ο μικροελεγκτής δίνει εντολή κίνησης αλλά ο encoder δεν καταγράφει μεταβολή θέσης, το σύστημα να διακόπτει την παροχή ρεύματος στον κινητήρα. Αυτό θα προστατεύσει τους κινητήρες και τα γρανάζια του θόλου σε περίπτωση που κάποιο αντικείμενο μπλοκάρει τη ράγα περιστροφής.
- **Ενοποίηση Video Stream στο IoT Dashboard:** Μια ακόμα χρήσιμη επέκταση θα ήταν η ενοποίηση των διαφορετικών εφαρμογών (Arduino IoT App, Tapo App, Astronomy Software) σε μία ενιαία πλατφόρμα. Η χρήση ενός κεντρικού server θα μπορούσε να συγκεντρώσει τον έλεγχο του Arduino και την εικόνα των καμερών σε ένα περιβάλλον, προσφέροντας στον χρήστη μια αναβαθμισμένη εμπειρία.

11. Προσαρμογή του συστήματος στο αστεροσκοπείο της Υπάτης

Η μετάβαση από το μοντέλο προσομοίωσης σε εφαρμογή στο "Κακογιάννειο Αστεροσχολείο Υπάτης", απαιτεί την προσαρμογή των ηλεκτρονικών ελέγχου του Arduino με τον βιομηχανικό ηλεκτρολογικό εξοπλισμό του θόλου.

1. **Έλεγχος Ισχύος:** Το σύστημα που αναπτύχθηκε σχεδιάστηκε για τις ανάγκες της πτυχιακής. Προκειμένου να μπορεί να εγκατασταθεί και να λειτουργήσει στο αστεροσκοπείο πρέπει οι έξοδοι του Arduino να συνδεθούν μέσω κατάλληλων optocouplers και drivers σε ρελέ βιομηχανικού τύπου τα οποία με τη σειρά τους θα δίνουν εντολές στους πίνακες ισχύος που ελέγχουν τους τριφασικούς ηλεκτροκινητήρες του θόλου.
2. **Αναβάθμιση Αισθητήρων:** Οι πλακέτες αισθητήρων Hall που χρησιμοποιήθηκαν στο μοντέλο θα πρέπει να αντικατασταθούν από βιομηχανικούς Επαγωγικούς Αισθητήρες Εγγύτητας (Inductive Proximity Sensors) ή ισχυρούς βιομηχανικούς διακόπτες ορίων (Limit Switches) με στεγανότητα επιπέδου IP67 για προστασία από την υγρασία και τη σκόνη.
3. **Κωδικοποιητές (Encoders):** Στη θέση των Modulino Knobs, θα πρέπει να εγκατασταθούν βιομηχανικοί Absolute Rotary Encoders, απευθείας συνδεδεμένοι στους άξονες του τηλεσκοπίου και του θόλου, προσφέροντας υψηλότερη ανάλυση (πλήθος βημάτων) για απόλυτη ακρίβεια σκόπευσης.
4. **Αναβάθμιση-προσαρμογή** του λογισμικού ώστε να υποστηρίζει τις αλλαγές στο hardware.

12. Συμπεράσματα - Προτάσεις

Η παρούσα πτυχιακή εργασία είχε ως αντικείμενο τον σχεδιασμό και την ανάπτυξη ενός συστήματος απομακρυσμένου ελέγχου περιστροφής θόλου αστεροσκοπείου, βασισμένου σε τεχνολογίες χαμηλού κόστους και ανοιχτής αρχιτεκτονικής. Η ανάπτυξη του συστήματος πραγματοποιήθηκε λαμβάνοντας υπόψη τις ανάγκες και τα χαρακτηριστικά λειτουργίας του αστεροσκοπείου της Υπάτης, με βασικό στόχο τη δημιουργία μιας αξιόπιστης και επεκτάσιμης λύσης αυτοματισμού.

Κατά τη διάρκεια της εργασίας πραγματοποιήθηκε ανάλυση των λειτουργικών και μη λειτουργικών απαιτήσεων, σχεδιάστηκε η αρχιτεκτονική του συστήματος και κατασκευάστηκε ένα λειτουργικό πρωτότυπο. Το σύστημα περιλαμβάνει τοπικό και απομακρυσμένο έλεγχο της περιστροφής του θόλου και της θυρίδας, αυτόματο συγχρονισμό με τη θέση του τηλεσκοπίου, παρακολούθηση αισθητήρων και λειτουργίες ασφαλείας για την προστασία του εξοπλισμού.

Η επιλογή της πλατφόρμας Arduino Uno R4 WiFi σε συνδυασμό με αισθητήρες Hall, rotary encoders και βηματικούς κινητήρες, επέτρεψε την ανάπτυξη ενός συστήματος χαμηλού κόστους κατασκευής και με δυνατότητα επεκτασιμότητας. Παράλληλα, η αξιοποίηση τεχνολογιών IoT και cloud επικοινωνίας κατέστησε δυνατή την απομακρυσμένη παρακολούθηση και διαχείριση του θόλου μέσω διαδικτύου.

Ιδιαίτερη σημασία είχε η ανάπτυξη του λογισμικού. Η χρήση non-blocking αρχιτεκτονικής και finite state machines (FSM) συνέβαλε στη σταθερότητα και την αξιοπιστία του συστήματος, επιτρέποντας την ταυτόχρονη διαχείριση κινητήρων, αισθητήρων, λειτουργιών ασφαλείας και cloud επικοινωνίας χωρίς σημαντικές καθυστερήσεις. Επιπλέον, ο αλγόριθμος αυτόματου συγχρονισμού και επιλογής συντομότερης διαδρομής συνέβαλε στη βελτιστοποίηση της λειτουργίας του θόλου και στη μείωση των περιττών κινήσεων.

Κατά την ανάπτυξη αναδείχθηκε επίσης η σημασία της συνεχούς αξιολόγησης και του επανασχεδιασμού κατά την υλοποίηση ενός embedded συστήματος ελέγχου. Κατά τη διάρκεια των πρώτων δοκιμών εντοπίστηκαν προβλήματα συγχρονισμού και επικοινωνίας, μη ομαλή περιστροφή των κινητήρων, και μια μεγάλη σε διάρκεια χρόνου εκτέλεσης της συνάρτησης βιβλιοθήκης επικοινωνίας με το cloud και τα οποία στη συνέχεια αντιμετωπίστηκαν με τροποποιήσεις στην αρχιτεκτονική του λογισμικού. Η διαδικασία αυτή συνέβαλε στη βελτίωση της λειτουργίας.

Τα αποτελέσματα των δοκιμών έδειξαν ότι το σύστημα μπορεί να ανταποκριθεί στις απαιτήσεις ενός εκπαιδευτικού αστεροσκοπείου, παρέχοντας λειτουργίες που συναντώνται σε πολύ ακριβότερα συστήματα αυτοματισμού.

Ωστόσο, υπάρχουν περιθώρια περαιτέρω βελτίωσης και επέκτασης του συστήματος. Μελλοντικά θα μπορούσε να προστεθεί υποστήριξη πρωτοκόλλων ASCOM ή INDI για διασύνδεση με έτοιμο λογισμικό, καταγραφή ιστορικού συμβάντων και δεδομένων, αποστολή ειδοποιήσεων, καθώς και λειτουργίες παρακολούθησης και πρόβλεψης καιρικών συνθηκών.

Η αξιοποίηση των διαθέσιμων τεχνολογιών για embedded systems, IoT και αυτοματισμού μπορεί να προσφέρει οικονομικές λύσεις για την αναβάθμιση εκπαιδευτικών και ερασιτεχνικών αστεροσκοπείων, ενισχύοντας τη δυνατότητα απομακρυσμένης παρατήρησης, την ασφάλεια του εξοπλισμού και την ευχρηστία του συστήματος.

Βιβλιογραφία – Ηλεκτρονικές Πηγές

- [1] M. Ismail, S. H. Elbanna, M. A. El-Dabah, and F. Elnagahy, "A Complete Affordable Control System for Remote Astronomical Observing Accompanied by an Intelligent Controller," IEEE Access. <https://ieeexplore.ieee.org/document/9887914/>
- [2] "Design of a Low Cost Control System for an Astronomical Observatory," Rev. Ingenierías Universidad de Medellín.
http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1794-91652020000100103
- [3] M. A. El-Dabah et al., "Low cost automated astronomical observatory for remote operation: A requisite analysis." <https://www.researchgate.net/publication/319118320>
- [4] OASIS, "MQTT Version 5.0," OASIS Standard. <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>
- [5] ASCOM Initiative, "ASCOM - Standards for Astronomy." <https://ascom-standards.org/>
- [6] INDI Library, "INDI - Instrument Neutral Distributed Interface." <https://www.indilib.org/>
- [7] E. White, Making Embedded Systems: Design Patterns for Great Software, 2nd ed. Sebastopol, CA: O'Reilly Media, 2024.
- [8] J. Jezný and M. Čurilla, "Position Measurement with Hall Effect Sensors," American Journal of Mechanical Engineering, vol. 1, no. 7, pp. 231-235, 2013.
- [9] D. Ibrahim, Mastering the Arduino Uno R4. Elektor International Media, 2023.
- [10] Arduino Documentation, "Arduino and Stepper Motor Configurations." <https://docs.arduino.cc/learn/electronics/stepper-motors/>
- [11] Arduino Documentation, "Inter-Integrated Circuit (I²C) Protocol." <https://docs.arduino.cc/learn/communication/wire/>

[12] Arduino Documentation, "Modulino Knob - How-To Guide."

<https://docs.arduino.cc/tutorials/modulino-knob/how-general/>

[13] Arduino, "Arduino Cloud." <https://app.arduino.cc/>

[14] Εργαλειοτεχνική Κρήτης, "Αστεροσκοπείο."

<https://ergaleiotechniki.gr/product/asteroskopeio/>

[15] Wikipedia, "Αστεροσκοπείο". <https://el.wikipedia.org/wiki/Αστεροσκοπείο>

[16] Wikipedia, "Τηλεσκόπιο". <https://el.wikipedia.org/wiki/Τηλεσκόπιο>

[17] Δήμος Λαμιέων, "Κακογιάννειο Αστεροσχολείο Υπάτης."

<https://www.lamia.gr/basiki-selida/kakogianneio-asterosholeio-ypatis>

Παράρτημα Α: ΚΩΔΙΚΑΣ

```
/*  
  Sketch generated by the Arduino IoT Cloud Thing "Untitled"  
  https://create.arduino.cc/cloud/things/f035959f-4585-4db9-  
  8592-9fb536190f39  
  
  Arduino IoT Cloud Variables description  
  
  The following variables are automatically generated and  
  updated when changes are made to the Thing  
  
  int positionDome;  
  int positionTelescope;  
  bool close;  
  bool closeDoor;  
  bool closedSensor;  
  bool goHome;  
  bool homeSensor;  
  bool ledDomeAuto;  
  bool ledDomeHome;  
  bool ledDomeLeft;  
  bool ledDomeRight;  
  bool openDoor;  
  bool openedSensor;  
  bool rainSensor;  
  bool sync;  
  bool turnLeft;  
  bool turnRight;  
  
  Variables which are marked as READ/WRITE in the Cloud  
  Thing will also have functions  
  which are called when their values are changed from the  
  Dashboard.  
  These functions are generated with the Thing and added at  
  the end of this sketch.  
*/  
  
#include "thingProperties.h"  
#include <Arduino_LED_Matrix.h>  
#include <Arduino_CloudConnectionFeedback.h>  
#include <Arduino_Modulino.h>  
#include <Stepper.h>  
  
ArduinoLEDMatrix matrix;  
  
// Δημιουργία των αντικειμένων ModulinoKnob  
ModulinoKnob encoderDome(0x3a);          // default Address
```

```
ModulinoKnob encoderTelescope(0x39); // Has been changed
with AddressChanger

// Δημιουργία των αντικειμένων ModulinoButtons
ModulinoButtons buttons1(0x3e); // default Address
ModulinoButtons buttons2(0x3d); // Has been changed with
AddressChanger

// Ορισμός των μεταβλητών των πλήκτρων για την κίνηση
// του θόλου αριστερά, δεξιά, Home position,
bool btnDomeLeft, btnDomeRight, btnDomeHome, btnAutoSync;

// Ορισμός των μεταβλητών των πλήκτρων για το
άνοιγμα/κλείσιμο θυρίδας
bool btnDoorOpen, btnDoorClose;
// Ορισμός των αντίστοιχων leds
bool ledDoorOpen, ledDoorClose;

// Δημιουργία των αντικειμένων Stepper Motor
Stepper DomeStepper = Stepper(2048, 13, 11, 10, 12);
Stepper DoorStepper = Stepper(2048, 7, 5, 4, 6);

const int DomeHallPin = A0; // Ορισμός του pin του
αισθητήρα hall του θόλου
const int DoorOpenHallPin = 8; // Ορισμός του pin του
αισθητήρα hall του ανοίγματος της θυρίδας
const int DoorCloseHallPin = 9; // Ορισμός του pin του
αισθητήρα hall του κλεισίματος της θυρίδας
const int RainSensorPin = A1; // Ορισμός του pin του
αισθητήρα βροχής

int prevPositionTelescope;

enum {movingHome, movingLeft, movingRight, movingSync,
stopped}; // καταστάσεις του θόλου (domeState)
uint8_t domeState;

enum {opened, closed, opening, closing}; // καταστάσεις της
θυρίδας (doorState)
uint8_t doorState;

enum {timeToUpdateCloud, timeForBtn1, timeForBtn2,
timeForDome, timeForDoor, timeToReadEncoders,
timeForSerial}; // καταστάσεις χρονισμού (timingState)
uint8_t timingState;

bool synced; // Αν true, υποδηλώνει τον συγχρονισμό του
τηλεσκόπιου με τον θόλο.
```

```
void setup() {  
  
    // Initialize serial and wait for port to open:  
    Serial.begin(115200);  
  
    // This delay gives the chance to wait for a Serial  
    Monitor without blocking if none is found  
    delay(2500);  
  
    matrix.begin();  
  
    // Defined in thingProperties.h  
    initProperties();  
  
    // Connect to Arduino IoT Cloud  
    ArduinoCloud.begin(ArduinoIoTPreferredConnection,false);  
  
    waitForArduinoCloudConnection(matrix);  
  
    /*  
        The following function allows you to obtain more  
information  
        related to the state of network and IoT Cloud  
connection and errors  
        the higher number the more granular information you'll  
get.  
        The default is 0 (only errors).  
        Maximum is 4  
    */  
    setDebugMessageLevel(2);  
    ArduinoCloud.printDebugInfo();  
  
    // Initialize Modulino I2C communication  
    Modulino.begin();  
  
    // Detect and connect to encoder modules  
    encoderDome.begin();  
    encoderTelescope.begin();  
  
    // Ορισμός των pin για τις εισόδους των αισθητήρων  
    pinMode(DomeHallPin, INPUT);  
    pinMode(DoorOpenHallPin, INPUT);  
    pinMode(DoorCloseHallPin, INPUT);  
    pinMode(RainSensorPin, INPUT);  
  
    // Detect and connect to buttons modules  
    buttons1.begin();  
    buttons2.begin();  
}
```

```
// Αρχικοποίηση των button leds
buttons1.setLeds(ledDomeLeft, ledDomeRight, ledDomeHome);
buttons2.setLeds(ledDomeAuto, ledDoorOpen, ledDoorClose);

// Ρύθμιση της ταχύτητας σε 10 στροφές ανά λεπτό (RPM)
// max = 15
DomeStepper.setSpeed(10);
DoorStepper.setSpeed(10);

doorState = closing; // κλείσιμο θυρίδας

if(!digitalRead(DomeHallPin))
    domeState = movingHome; // Περιστροφή του θόλου έως την
    αρχική του θέση
else
    domeState = stopped;

positionTelescope = 1000; // Αρχικοποίηση τηλεσκοπίου σε
    τυχαία θέση
}

void loop() {

    homeSensor = digitalRead(DomeHallPin);
    openedSensor = digitalRead(DoorOpenHallPin);
    closedSensor = digitalRead(DoorCloseHallPin);
    rainSensor = !digitalRead(RainSensorPin);

    if(encoderTelescope.isPressed())
    {
        ResetTelescope();
    }

    switch(timingState)
    {
        case timeToUpdateCloud:
            ArduinoCloud.update();
            timingState = timeForBtn1;
            break;

        case timeForBtn1:
            if(buttons1.update())
            {
                if (buttons1.isPressed('A'))
                    btnDomeLeft = true;
                else
                    btnDomeLeft = false;
                if (buttons1.isPressed('B'))
```

```
        btnDomeRight = true;
    else
        btnDomeRight = false;
    if (buttons1.isPressed('C'))
        btnDomeHome = true;
    else
        btnDomeHome = false;
}
if ( (btnDomeHome || goHome) && ledDomeHome)
    domeState = movingHome; // Περιστροφή του θόλου έως
την αρχική του θέση
    buttons1.setLeds(ledDomeLeft, ledDomeRight,
ledDomeHome);
    timingState = timeForBtn2;
    break;

case timeForBtn2:
    if(buttons2.update())
    {
        if(buttons2.isPressed('A')) // πατήθηκε το Auto Sync
            autoSyncPressed();
        if(buttons2.isPressed('B'))
        {
            btnDoorOpen = true;
        }
        else
            btnDoorOpen = false;
        if(buttons2.isPressed('C'))
        {
            btnDoorClose = true;
        }
        else
            btnDoorClose = false;
    }
    buttons2.setLeds(ledDomeAuto, ledDoorOpen,
ledDoorClose);
    timingState = timeForDome;
    break;

case timeForDome:
    switch(domeState)
    {
        case movingHome:
            if(!homeSensor)
                DomeStepper.step(2);
            else
            {
                DomeStepper.step(50); // Για να μετακινηθεί στο
κέντρο του μαγνήτη
```

```
encoderDome.set(0); // Ο θόλος τώρα βρίσκεται
στην αρχική του θέση
    ledDomeLeft = true;
    ledDomeRight = true;
    domeState = stopped;
}
break;
case movingLeft:
    DomeStepper.step(2);
    if(!btnDomeLeft && !turnLeft)
        domeState = stopped;
    break;
case movingRight:
    DomeStepper.step(-2);
    if((!btnDomeRight) && (!turnRight))
        domeState = stopped;
    break;
case movingSync:
    if(!ledDomeAuto)
        domeState = stopped;
    else if(positionTelescope != positionDome)
        Syncing();
    else
        domeState = stopped;
    break;
case stopped:
    if(!ledDomeAuto)
    {
        ledDomeLeft = true;
        ledDomeRight = true;
        if ((btnDomeLeft && !btnDomeRight) || (turnLeft
&& !turnRight))
            domeState = movingLeft;
        else if(btnDomeRight || turnRight)
            domeState = movingRight;
    }
    else if(positionTelescope != positionDome)
        domeState = movingSync;
    break;
default:
    break;
}
timingState = timeForDoor;
break;

case timeForDoor:
    switch(doorState)
    {
        case opened:
```

```
        if(btnDoorClose || closeDoor || rainSensor)
            doorState = closing;
        break;
    case closed:
        if((btnDoorOpen || openDoor) && (!rainSensor))
            doorState = opening;
        break;
    case opening:
        if(!openedSensor)
            DoorStepper.step(2);
        else
        {
            ledDoorOpen = false;
            ledDoorClose = true;
            doorState = opened;
        }
        break;
    case closing:
        if(!closedSensor)
            DoorStepper.step(-2);
        else
        {
            ledDoorOpen = true;
            ledDoorClose = false;
            doorState = closed;
        }
        break;
    default:
        break;
}
timingState = timeToReadEncoders;
break;

case timeToReadEncoders:
    positionDome = -encoderDome.get(); // επειδή είναι
    ανάποδα τοποθετημένο (από κάτω προς τα πάνω)
    if(homeSensor || positionDome == 30)
        encoderDome.set(0);
    else if(positionDome < 0){
        positionDome += 30;
        encoderDome.set(-positionDome); // για να
    τοποθετηθεί στη σωστή του θέση επειδή είναι ανάποδα
    }
    if(positionTelescope < 1000)
    {
        positionTelescope = encoderTelescope.get();
        if(positionTelescope < 0)
        {
            positionTelescope += 30;
        }
    }
}
```

```
        encoderTelescope.set(positionTelescope);
    }
    else if(positionTelescope == 30)
    {
        positionTelescope = 0;
        encoderTelescope.set(positionTelescope);
    }
}
if(positionDome == positionTelescope)
    synced = true;
else
    synced = false;
timingState = timeForSerial;
break;

case timeForSerial:
//    Serial.print("doorState: ");
//    Serial.println(doorState);
//    Serial.print("Dome position: ");
//    Serial.println(positionDome);
//    Serial.print("Telescope position: ");
//    Serial.println(positionTelescope);
    timingState = timeToUpdateCloud;
    break;
default:
    break;
}

if(!homeSensor && !ledDomeAuto)
    ledDomeHome = true;
else
    ledDomeHome = false;
}

void Syncing()
{
    if(positionTelescope > positionDome)
    {
        if((positionTelescope - positionDome >= 15))
            DomeStepper.step(2);
        else
            DomeStepper.step(-2);
    }
    else if(positionTelescope < positionDome)
    {
        if((positionDome - positionTelescope >= 15))
            DomeStepper.step(-2);
        else
```

```
        DomeStepper.step(2);
    }

}

void ResetTelescope()
{
    encoderTelescope.set(0); // Το τηλεσκόπιο βρίσκεται στην
    αρχική του θέση
    positionTelescope = 0;
    prevPositionTelescope = 0;
    if(positionDome == positionTelescope)
        synced = true;
    else
        synced = false;
}

void autoSyncPressed(void)
{
    if(ledDomeAuto)
        ledDomeAuto = false;
    else if(synced)
    {
        ledDomeAuto = true;
        ledDomeLeft = false;
        ledDomeRight = false;
        domeState = movingSync;
    }
}

/*
    Since TurnLeft is READ_WRITE variable, onTurnLeftChange()
    is
    executed every time a new value is received from IoT
    Cloud.
*/
void onTurnLeftChange() {
    // Add your code here to act upon TurnLeft change
}

/*
    Since TurnRight is READ_WRITE variable,
    onTurnRightChange() is
    executed every time a new value is received from IoT
    Cloud.
*/
void onTurnRightChange() {
    // Add your code here to act upon TurnRight change
}
```

```
}

/*
    Since BtnDoorOpen is READ_WRITE variable,
    onBtnDoorOpenChange() is
        executed every time a new value is received from IoT
    Cloud.
*/
void onBtnDoorOpenChange() {
    // Add your code here to act upon BtnDoorOpen change
}

/*
    Since BtnDoorClose is READ_WRITE variable,
    onBtnDoorCloseChange() is
        executed every time a new value is received from IoT
    Cloud.
*/
void onBtnDoorCloseChange() {
    // Add your code here to act upon BtnDoorClose change
}

/*
    Since CloseDoor is READ_WRITE variable,
    onCloseDoorChange() is
        executed every time a new value is received from IoT
    Cloud.
*/
void onCloseDoorChange() {
    // Add your code here to act upon CloseDoor change
}

/*
    Since OpenDoor is READ_WRITE variable, onOpenDoorChange()
    is
        executed every time a new value is received from IoT
    Cloud.
*/
void onOpenDoorChange() {
    // Add your code here to act upon OpenDoor change
}

/*
    Since GoHome is READ_WRITE variable, onGoHomeChange() is
    executed every time a new value is received from IoT
    Cloud.
*/
void onGoHomeChange() {
    // Add your code here to act upon GoHome change
}
```

```
}

/*
  Since Sync is READ_WRITE variable, onSyncChange() is
  executed every time a new value is received from IoT
  Cloud.
*/
void onSyncChange() {
  // Add your code here to act upon Sync change
  if(sync)
    autoSyncPressed();
}
```

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.