Contents lists available at ScienceDirect

HardwareX

journal homepage: www.elsevier.com/locate/ohx

Hardware Article

Motor-driven autonomous system for controlling beamline iris diaphragm apertures

Daniel Pinheiro Leal[a], Jörg Krämer[b,*], Wilfried Nörtershäuser[b]

[a] Division of Engineering Science, University of Toronto, Canada
[b] Institut für Kernphysik, Technische Universität Darmstadt, Germany

A R T I C L E   I N F O

Article history:
Received 23 June 2020
Received in revised form 11 December 2020
Accepted 11 December 2020

Keywords:
Collinear laser spectroscopy
Beamline
Automated measurements
Arduino
Motorized iris diaphragm

A B S T R A C T

At the collinear apparatus for laser spectroscopy and applied sciences (COALA) at TU Darmstadt, collinear laser spectroscopy is applied to perform high-precision measurements of atomic transition frequencies and high voltages in an ultra-high vacuum beamline. In such laser spectroscopy beamlines, iris diaphragms are used to reduce ion beam divergence, and to ensure a good laser and ion beam overlap. Since the system that controls the diameter of the apertures presents strong hysteresis, an automated adjustment is desirable to enhance reproducibility in the aperture settings, and to reduce the effort in performing measurements where different diameters are required, especially in alternating collinear and anti-collinear measurements. To achieve this, the *Iris Mover* system was designed and implemented. The Iris Mover system consists of motor-driven iris apertures which can be easily controlled by users through a computer, and which accounts for hysteresis effects. Here, we explain the design process of the Iris Mover and demonstrate and discuss its functionality.

Specifications table

Hardware name	Iris Mover
Subject area	• Physical Sciences • Nuclear Physics • Collinear Laser Spectroscopy
Hardware type	• Measuring physical properties and in-lab sensors • Electrical engineering and computer science • Mechanical engineering
Open Source License	GNU GPL v3
Cost of Hardware	100 US$–150 US$
Source File Repository	https://doi.org/10.17605/OSF.IO/FND9V

* Corresponding author.
E-mail addresses: daniel.leal@mail.utoronto.ca (D. Pinheiro Leal), jkraemer@ikp.tu-darmstadt.de (J. Krämer).

https://doi.org/10.1016/j.ohx.2020.e00166
2468-0672/© 2020 The Authors. Published by Elsevier Ltd.

1. Hardware in context

Collinear Laser Spectroscopy (CLS) is a technique that is used to determine nuclear properties such as spins, magnetic and electric quadrupole moments, and mean square charge radii [1,2]. Furthermore, it has applications, e.g. for many-body QED systems [3,4], tests of special relativity [5] and can be used to measure high voltages to a very high degree of precision [6,7]. At TU Darmstadt, the collinear apparatus for laser spectroscopy and applied sciences (COALA) has been set up recently [8]. The beamline features multiple iris diaphragms to ensure a good superposition of laser and ion beams [9]. At COALA, the iris diameters are adjustable, and thus can be varied depending on the needs of each measurement. Similar iris diaphragm apertures are also used in other optical systems beyond CLS [10–12].

Before the development of the Iris Mover system, the iris apertures at COALA were adjusted manually by a linear feed-through system controlled by screw motion of a knob. There are two main problems with this manual system: First, manually rotating the knobs to adjust the iris diameter is a process that takes time, especially since it must be repeated multiple times for each of the three knobs in every run of the experiment. Second, the process of manually adjusting the diameters is very prone to errors and variability due to hysteresis, and thus harms the reproducibility of the aperture settings. In this context, having an autonomous system to control the apertures is very beneficial.

While motorized iris diaphragms exist on the market, their prices can be quite prohibitive. Furthermore, they would require a potentially complex installation procedure to be adapted to a beamline, or even be inviable due to the spatial constraints. The Iris Mover design presented here can be a useful reference to develop a similar motor-driven aperture controller accounting for the specificities of a beamline and reducing implementation costs, therefore being valuable as an open-source hardware (OSHW) project [13].

2. Hardware description

The designed system consists of motor-driven iris apertures that are controlled by users through serial messages sent from a computer to an Arduino UNO microcontroller. Upon receipt of the user input, which contains the desired diameter for the iris, the linear motion required to change the iris aperture is created by a stepper motor with a hollow threaded shaft, that contains a threaded rod. The motor is supported by a platform from which a side platform extends and contains two limit switches that serve as stops to prevent mechanical damage to the iris, by restricting the range of motion of the system. An activator piece attached to the threaded rod is used to activate the limit switches. Fig. 1 provides an overview of the assembled and installed iris mover system.

The fact that the Iris Mover allows the user to control the iris diameters through a computer, from where other features of the beamline are also controlled, represents a major improvement in terms of ease of use. Previously, users had to manually rotate a knob every time a new iris diameter was required. The new automated version can be even included into the control system, enabling to run fully automated measurement sequences that include changes of the iris diameter. Furthermore, it improves measurement reproducibility when compared with the previous manually controlled system, and accounts for hysteresis effects. For instance, the previous system had a scale to allow manual adjustment of the diameters. However, setting a certain diameter of the iris according to the scale while approaching the diameter from lower and higher values would lead to slightly different apertures. Moreover, the new automatized system presents a much lower cost when compared to commercially available solutions, with the entire hardware cost ranging between 100 US$ and 150 US$. Finally, the Iris Mover accounts for the beamline spatial constraints of the COALA beamline, thus making the assembly process easier when compared to a commercially available solution.

Therefore, the Iris Mover design can be useful as a reference for:

- Developing a similar system to automate the movement of iris diaphragms in collinear laser spectroscopy beamlines or any other beamline for pure ion transport.
- Developing a controller for automated iris apertures in optical systems.

The Iris Mover system is divided into three subsystems: mechanical, electrical, and firmware. The following subsections will give a general overview of these subsystems.

2.1. Mechanical subsystem

The mechanical subsystem is the one concerned with providing appropriate support to the components of the iris mover (such as the stepper motor, the limit switches, and the circuitry), while also allowing the necessary mechanical movements to vary the diameter of the iris. The mechanical design process also took into consideration the spatial constraints imposed by the devices that already existed in the beamline. To accomplish these goals, the following parts were designed using the Siemens NX 10 CAD software: a main platform to support the stepper motor, a side platform and switch mounts to support the limit switches, an activator piece to trigger the limit switches, and attaching pieces to facilitate the attachment of the Iris Mover on the beamline.

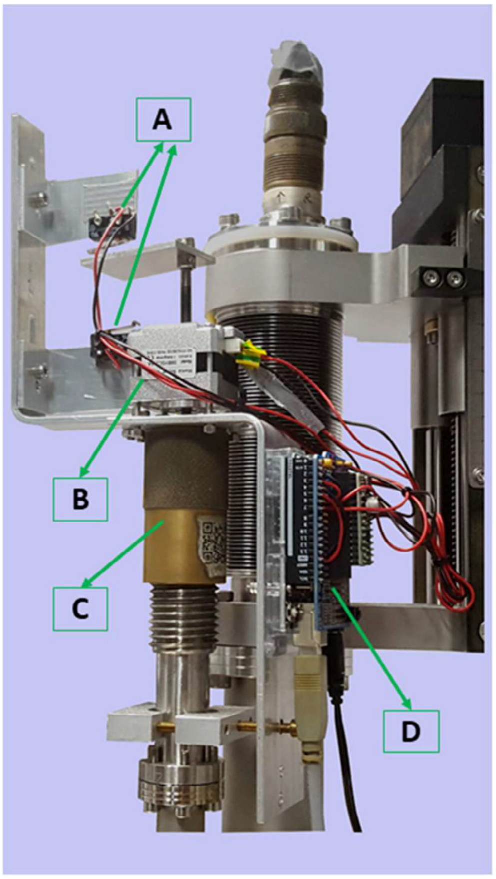


Fig. 1. Overview of the Iris Mover system. A: Limit Switches, B: Stepper Motor, C: Knob, D: Circuitry.

2.2. Electrical subsystem

The electrical subsystem is the one concerned with all the electrical connections and components needed to guarantee an efficient and failure-free operation of the iris mover system. To achieve the goals of the system, the stepper motor must be connected to a motor driver, and the driver and the limit switches must be connected to the Arduino. Fig. 2 shows the diagram of the circuit used by the Iris Mover.

As the resolution of full steps was sufficient for our application, signals MS1, MS2, and MS3 of the motor driver did not need to be controlled by the Arduino UNO. They are pulled down by internal resistors, and the motor operates in full-step mode. Higher resolution can be achieved by making use of these signals to control the driver's microstepping capability, if necessary.

The power to the system is provided through a wall plug with 12 V output. The 12 V are also used to power the motor driver, by connecting the V_{IN} pin of the Arduino to the V_{MOT} pin of the driver, as shown in Fig. 2. The driver must deliver up to 4A of current for the motor, so the supply should be able to provide at least this amount of current.

2.3. Firmware subsystem

The logic of operation of the Iris Mover is controlled by an Arduino UNO microcontroller. The Arduino development platform is very common in OSHW projects, being frequently used to develop inexpensive, automated sensing and measurement systems [14], and is at the core of several OSHW projects ranging from irrigation [15] and automatic animal feeders [16], to temperature control of microbial electrochemical technology systems [17].

The logic of the system consists of a 4-state state-machine, shown in Fig. 3. The states will be explained in detail in Section 6.

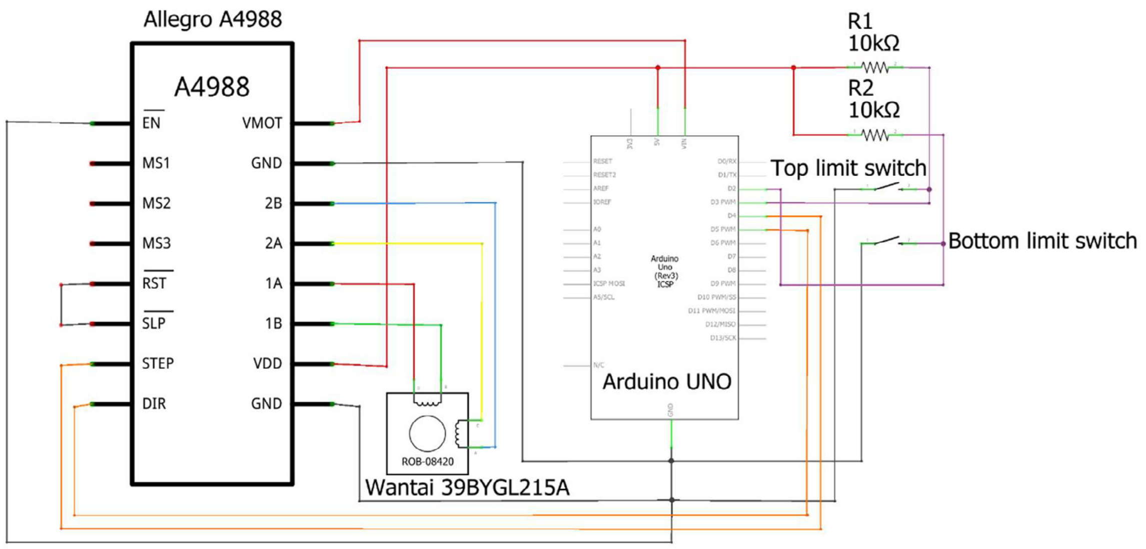


Fig. 2. The motor driver allows the Arduino to control the stepper motor. The limit switches are connected to GND and 5 V through a pull-up resistor, allowing the Arduino to identify when they have been triggered.

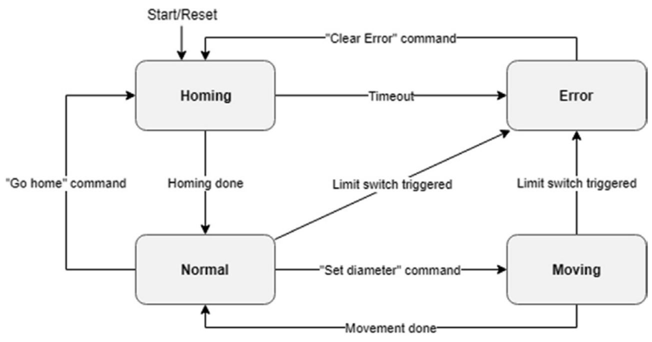


Fig. 3. State Diagram of the Iris Mover firmware.

3. Design files

3.1. Design files summary

Design file name	File type	Open source license	Location of the file
finalIrisMoverCode	Arduino Firmware (.ino)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
activator	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
attachingPieces	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
platform	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
sidePlatformFinal	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
switchMountLH	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
switchMountRH	3D CAD file (.stp)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V
irisMoverCircuit	Fritzing circuit schematic (.fzz)	GNU GPL v3	https://doi.org/10.17605/OSF.IO/FND9V

finalIrisMoverCode.ino: This file contains the entire firmware code needed for the Arduino in the Iris Mover system.
activator.stp: All .stp files can be accessed through free 3D CAD software, such as FreeCAD. This model represents the component used to trigger the limit switches.

attachingPieces.stp: Two of these components are used in the Iris Mover system to better attach the system to the beamline.
platform.stp: This is the main platform. It has holes on top for attachment to the beamline, securing the motor, securing the side platform, and allowing the threaded rod to go through. It has holes on the side to secure the Arduino and the circuitry, and to connect to the attaching pieces mentioned above.
sidePlatformFinal.stp: Side platform that extends out of the main platform and has two elongated holes where the switch mounts are attached.
switchMountLH.stp: Switch mount for the top limit switch.
switchMountRH.stp: Switch mount for the bottom limit switch. It is essentially a mirror image of the top switch mount.
irisMoverCircuit.fzz: This is a schematic of the electric circuit of the system. It can be opened using the open-source hardware tool Fritzing.

4. Bill of materials

4.1. Bill of materials

Designator	Component	Number	Cost per unit - Currency (US$)	Total cost-Currency (US$)	Source of materials	Material type
Stepper Motor	Wantai 39BYGL215A	1	29	29	https://www.tinyosshop.com/	Electronics
Arduino	Arduino UNO	1	23	23	https://store.arduino.cc/usa/	Electronics
Arduino Protoshield	Adafruit Proto Shield for Arduino	1	9.95	9.95	https://www.adafruit.com/	Electronics
Motor Driver	Allegro A4988	1	5.95	5.95	https://www.pololu.com/	Electronics
Limit Switch	Omron SS-5GL13	2	1.97	3.94	https://www.digikey.com/	Electronics
Male-to-male headers	Breakaway male headers: 1x40 pins	1	0.75	0.75	https://www.pololu.com/	Electronics
10 kOhm Resistor	10 kOhm Resistor	2				Electronics
M2 Bolts	M2 Bolts	4				Fasteners
M3 Bolts	M3 Bolts	10				Fasteners
M4 Bolts	M4 Bolts	7				Fasteners
M2 Nuts	M2 Nuts	4				Fasteners
M3 Nuts	M3 Nuts	4				Fasteners
M4 Nuts	M4 Nuts	4				Fasteners
Aluminum sheet	500 mm–500 mm–3 mm thick Aluminum sheet	1	26	26		Metal

5. Build instructions

5.1. Electrical subsystem

To realize the electrical subsystem, its components must be soldered according to the schematic in Fig. 2.

1. Male-to-male header pins are used to attach the Arduino UNO with the Arduino protoshield. These pins are then soldered onto the protoshield. Now that the Arduino is connected to the protoshield, all necessary connections can be made by soldering wires between the limit switches, stepper motor, and motor driver.
2. The motor driver circuit is soldered onto the protoshield.
3. Solder the resistors onto the protoshield. Fig. 4 shows the circuit board after soldering the motor driver and resistors.
4. Solder two wires onto each limit switch.

5. Make the connections of the stepper motor wires, limit switch wires, resistors, and motor driver to the corresponding pins in the Arduino protoshield as shown in Fig. 2.

5.2. Mechanical subsystem

The mechanical subsystem consists of multiple aluminum parts that are responsible for providing support for the multiple components of the Iris Mover (electric circuit, limit switches, stepper motor), as well as allowing the system to be securely connected to the beamline. This section will outline the purpose of each of these aluminum parts. The manufacture of these parts can be done by following the provided .stp files.

The main platform, shown in Fig. 5(a), consists of a 3 mm thick aluminum sheet bent at a 90-degree angle. To reduce the number of parts that had to be designed and manufactured, and minimize the changes to the beamline, the main platform is mounted on top of the previously existing knob.

The top of the main platform has 9 holes, as illustrated in Fig. 5(b). The top also has a cutout to reduce the space occupied by the platform and prevent overlap with other devices in the beamline. The side of the main platform has 5 holes, which allow the attachment of the circuitry and a connection to the attaching piece (to be discussed later in this section).

The side platform is a 3 mm thick aluminum component that extends out of the main platform. Its purpose is to accommodate the limit switch mounts (which will be discussed later). It features two 25 mm elongated holes to allow an adjustable positioning of the limit switch mounts, which are attached to the side platform by a screw and a nut. Fig. 6(a) illustrates the side platform in detail.

The limit switches are secured to the side platform using two limit switch mounts. The mounts are 3 mm thick aluminum sheets bent at a 90-degree angle, that contain 4 holes. Fig. 6(b) shows a 3D CAD model of the switch mount.

To be able to trigger the limit switches, an activator piece is attached to the threaded rod.

When the threaded rod moves too far up, or too far down, the activator piece presses the lever on the switch, causing it to be activated. The activation of the switch sends a signal to the Arduino logic, which stops the motion of the system, thus preventing mechanical damage. Fig. 7 shows the activator piece and the limit switches mounted in the iris mover system.

The whole mechanical system must be fixed in position. By attaching the main platform to the knob, and screwing the stepper and the side platform to the main platform, translational motion is prevented. However, it is also necessary to restrain the main platform from rotation. To achieve this, two "attaching pieces" are connected to the cylindrical part that supports the system, and the main platform is connected to the attaching pieces by a screw that goes through a hole on its side. Fig. 8 shows the attaching pieces and the connection point between them and the main platform.

The order of assembly of the mechanical subsystem should be:

1. Attach the limit switches to the switch mounts using 4 M2 bolts and nuts.
2. Attach the switch mounts to the side platform using 2 M4 bolts and nuts.
3. Attach the stepper's threaded rod to the linear feedthrough component using epoxy, as further discussed below.
4. The knob, which remains from the previous manual system, is secured to the beamline through its threaded interior.
5. Connect the circuitry to the side of the main platform using 4 M3 bolts and nuts.
6. Attach the main platform to the knob using 2 M3 bolts. The main platform could be attached to the beamline differently, however in our case we have decided to place it on top of the previously existing knob, by making two M3 threaded holes in the knob.

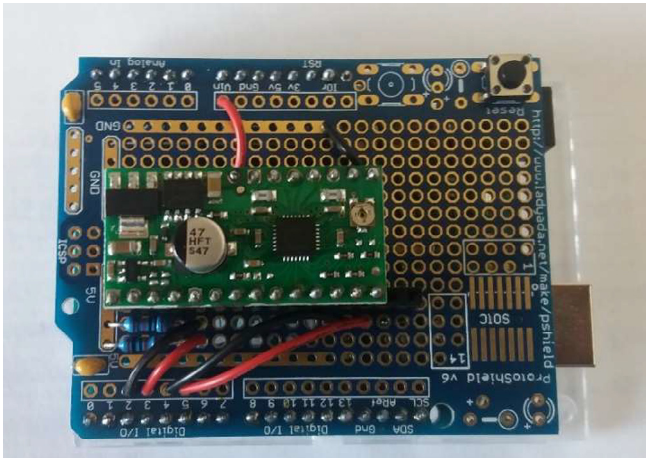


Fig. 4. The Arduino protoshield is connected to the Arduino UNO through male-to-male header pins, and the motor driver and resistors are soldered onto the protoshield.

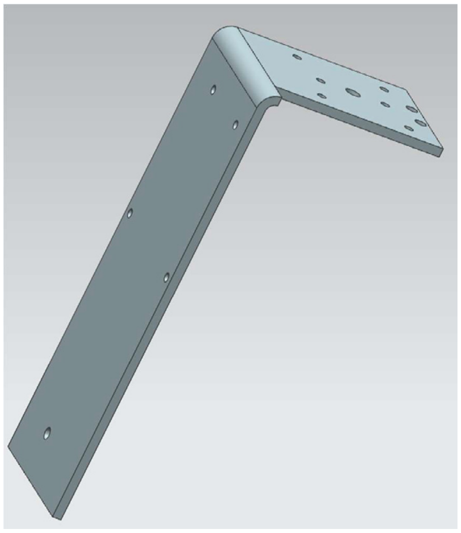


Fig. 5a. The main platform. It contains holes on top, for the side platform, stepper motor, threaded rod, and beamline connection, and on the side for the circuitry and attaching pieces.

7. Place the stepper motor over the main platform, allowing the threaded rod to go through the threaded hole in the motor.
8. Secure the stepper to the platform using 4 long M3 bolts.
9. Place the attaching pieces and use two long M4 bolts to attach them together.
10. Connect the main platform with the attaching pieces through a long M4 bolt.
11. Connect the side platform to the main platform using 2 M4 bolts and nuts.
12. Use two cylindrical parts with headless screws to secure the activator piece to the threaded rod. Nuts can also be used to secure the activator piece.

As shown in Fig. 9, the diameter of the iris is controlled by a linear feedthrough system, which moved according to the screw motion of a knob. This system existed in the beamline before the placement of the Iris Mover and was maintained.

The threaded rod of the stepper motor is connected to the linear feedthrough using epoxy glue.

5.3. Firmware subsystem

To setup the firmware subsystem, the user should download the provided Arduino file (finalIrisMoverCode.ino), and program it in the Arduino UNO microcontroller. The Arduino UNO should be connected to a computer through a USB cable, and the communication between Arduino and computer happens through serial strings.

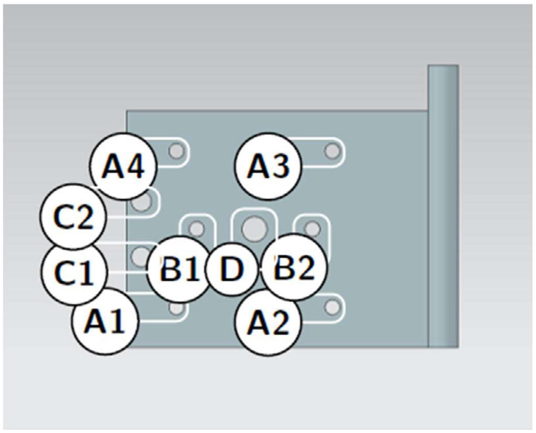


Fig. 5b. Top part - Holes for the stepper motor (A1, A2, A3, A4), Holes for attaching the platform to the knob (B1, B2), Holes for the side platform (C1, C2), Hole for the threaded rod (D).

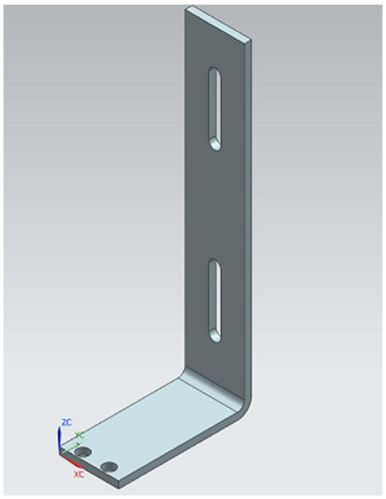


Fig. 6a. The side platform with holes for connection to the main platform and for supporting the limit switch mounts.

Furthermore, a calibration procedure must be done to compute the value of one of the parameters needed in the code. The parameter name is *minIrisDiameter*. This parameter is defined as the diameter of the iris when the bottom limit switch is being triggered. This value is very important to guarantee the precision of the system.

To compute this value, a leveling instrument and a camera can be used to look through the beamline, as shown in Fig. 10.

The leveling instrument points to a mirror setup, allowing the camera to capture images of the inner parts of the beamline at variable positions, and thus measure the iris diameters when a beam of light is illuminating the interior of the beamline. The images captured by the camera were then analyzed using the GIMP 2 software, as shown in Fig. 11.

By taking measurements of the iris in known diameter apertures, and noting the number of pixels in the image, a correspondence between the number of pixels in the image and the diameter value can be determined, and thus the value of *minIrisDiameter* can be computed from the image. The user should then update the value of this parameter in the code.

Notice that the number of pixels in the image offers a close approximation of the actual iris diameter, but that this measurement procedure has a degree of uncertainty associated with it. For instance, the image profile is not perfectly circular, and the pixel intensity values are not perfectly bright for the interior of the iris and perfectly dark outside it. In our analysis of results, the image profile was assumed to be perfectly circular, and the reading was obtained by measuring the horizontal diameter of the profile. The measurement error related to this procedure was taken to be 10% of the reading of the number of pixels, rounded to one significant figure (e.g. in Image 11, the pixel reading is 76 pixels, thus the value is taken to be 76 ± 0.8 pixels for the analysis). This is how the uncertainty values later reported in Section 7. Validation and Characterization are obtained.

6. Operation instructions

The user of the Iris Mover can operate the system through serial strings that are sent from a computer to the Arduino UNO, which then generates the appropriate signals to control the stepper motor.

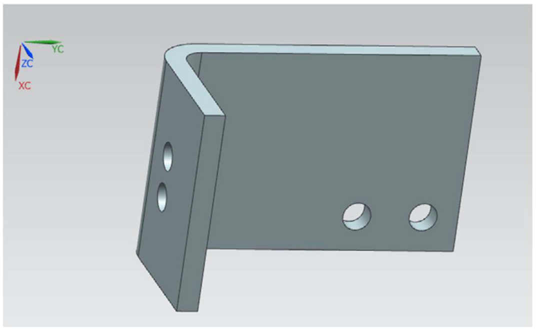


Fig. 6b. Limit switch mount has holes for connecting to the side platform, and for securing the limit switches.

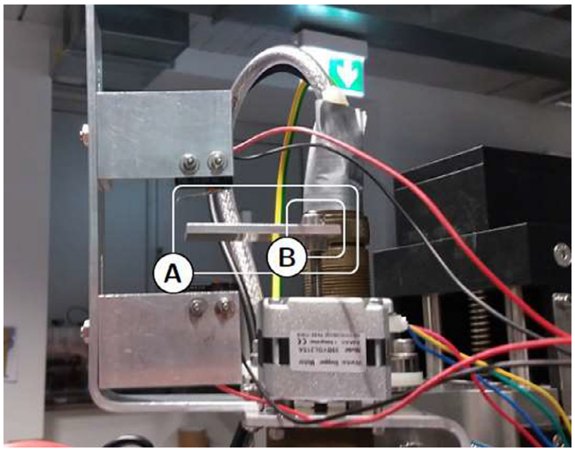


Fig. 7. The activator piece is used to trigger the limit switches. A – Activator piece. B – Attachment of the activator piece to the threaded rod that is mechanically moved by the stepper motor.

The logic of operation of the Iris Mover was previously shown in Fig. 3. The states of the state machine are further explained here:

1. Homing: This is the initialization state. The motor's rod will be moved down until the bottom limit switch is triggered, and then moved up until a certain pre-determined diameter is reached. If the procedure does not finish in a certain time limit, a timeout error occurs. If the procedure is successful, the machine transitions into the normal state.
2. Normal: The normal state is reached after the homing procedure is done. In this state, the machine is accepting diameter inputs, and such a command will make it transition to the moving state. A "Go home" command will initiate the homing procedure, and a triggering of a limit switch will cause an error.
3. Moving: The motor will be moved until a certain diameter defined by the user input is reached, and then transition into the normal state. An activation of any of the limit switches will immediately stop the motor, and the system will transition into the error state.

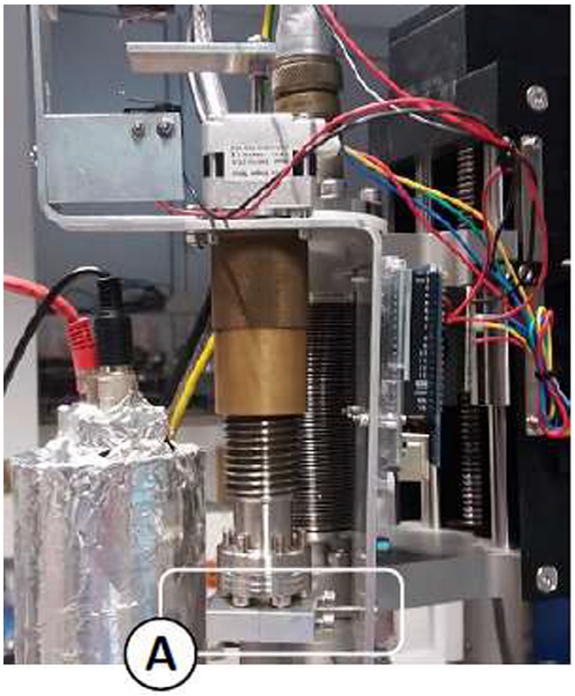


Fig. 8. The "attaching pieces" are used to prevent rotational motion of the system.

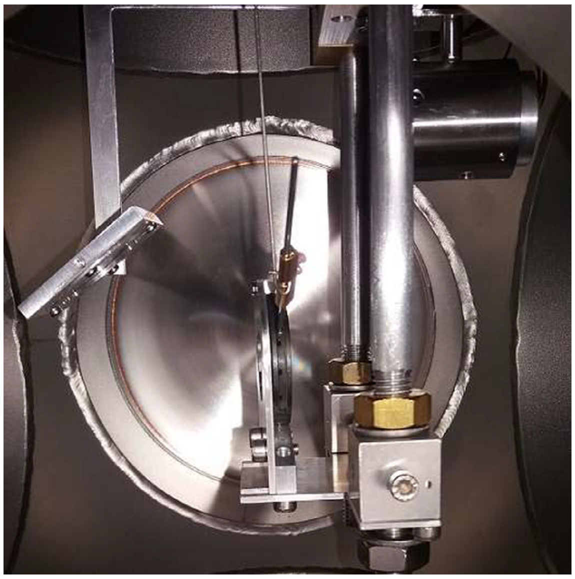


Fig. 9. Linear feedthrough is responsible for the mechanical movement to adjust the aperture.

4. Error: All motion is stopped, and no user input of diameters is processed. If the command "Clear error" is received, the motor will move the rod away from the currently activated switch and initiate a homing procedure if such motion removes the triggering of the switch.

The communication between the Arduino and the computer is done by strings sent through serial. Serial communication is handled using Arduino's built-in serial functions.

The communication protocol is the following:

<**n a d**>

where **n** is the station number (1, 2, or 3), **a** is the required action (Set diameter: 1, Get status: 2, Clear error: 3, Go home: 4, Move up for fine adjustments: 5, Move down for fine adjustments: 6), and **d** is the requested diameter (0 if a is not 1, and an integer where the last digit represents the fractional part of the diameter and the remaining digits represent its integer part if a is 1) for the strings that the computer sends to the Arduino, and:

<**s e p**>

where **s** is the current state (Homing: 1, Normal: 2, Moving: 3, Error: 4), **e** is the cause of the error (No error: 0, Top Switch Triggered: 1, Bottom Switch Triggered: 2, Both Switches Triggered: 3, No Switches Triggered: 4, Timeout: 5), and **p** is the current diameter (0 if unknown, and an integer where the last digit represents the fractional part of the diameter and the remaining digits represent its integer part, otherwise) for the string that the Arduino sends to the PC as a response to a Get Status request. The characters '<' and '>' are used as initiation and termination characters respectively, in each message

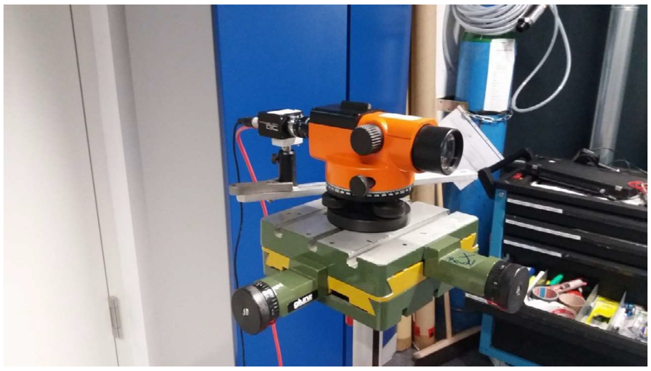


Fig. 10. Leveling instrument and camera used to measure iris diameters.

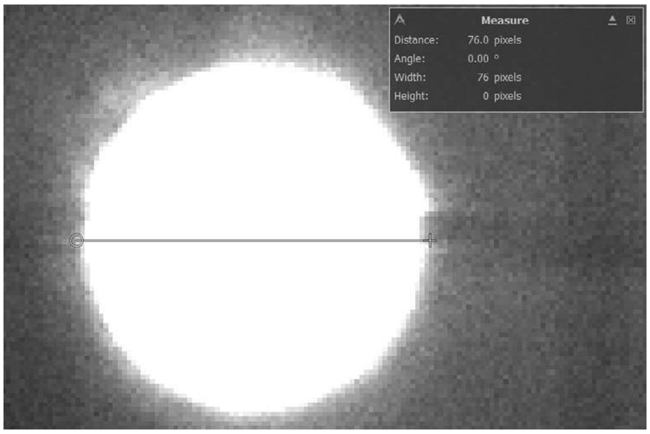


Fig. 11.. An image captured through the camera and leveling instrument setup being analyzed using the GIMP 2 software.

sent using the proposed protocol. Note that the station number (**n**) parameter in the strings sent from the computer exists because three instances of the Iris Mover were installed in the COALA beamline, and all of them were being controlled by the same computer. The baud rate used in the system is 9600.

Such serial strings can be sent and received through terminals such as Tera Term or PuTTY, or a graphical user interface (GUI) can be designed to make the communication easier and more friendly.

At the TU Darmstadt laboratory, a GUI for the Iris Mover was developed using the laboratory's software framework, as shown in Fig. 12. Since it is based on the lab's framework, the GUI source code is not provided together with the design files associated with this paper. A similar interface could be developed for a new application of the IrisMover, or the system can be controlled through a terminal as mentioned previously, in the absence of a GUI.

7. Validation and Characterization

Using the same methodology as used to determine the *minIrisDiameter* parameter (see Section 5.3. Firmware Subsystem), the diameter of the iris was measured multiple times after being set by the Iris Mover upon receiving user input. The expected and observed values, as well as their absolute differences are shown in Table 1. Measurement uncertainties exist, as noted in Section 5.3. Firmware Subsytem, and are taken into account during the analysis. In this table the results are noted in the same order as they were measured. That is, the measurements were made initially coming from lower values, and then returning from the higher values, which allows the investigation on whether the hysteresis problem of the manual system remained an issue.

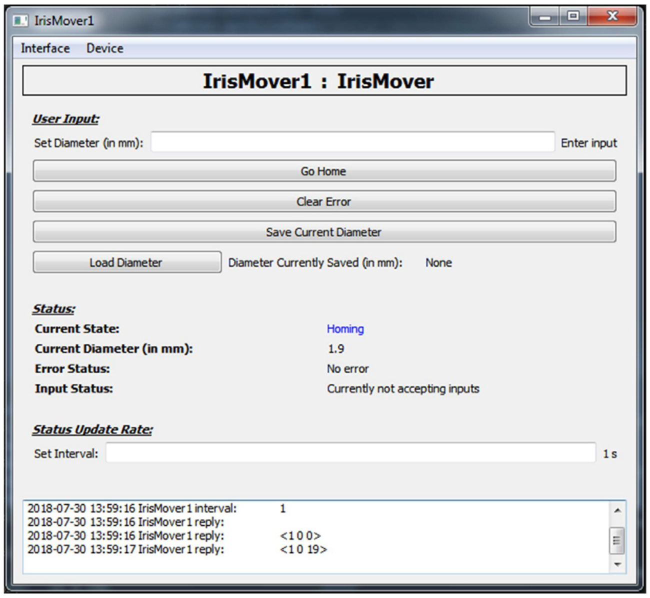
As shown in Table 1 the observed values of the iris apertures were consistent with the desired values accounting for the reading and measurement uncertainties, and thus the Iris Mover was found to be accurate for use in the COALA beamline.

The results in Table 2 show the difference observed in diameter readings when approaching various diameters from lower and higher diameter values. Each entry in the second column of Table 2 is given by $\Delta d = |d_+ - d_-|$, where d_+ is the diameter reading when approaching diameter d from higher values, and d_- is the diameter reading when approaching diameter d from lower values.

The Iris Mover manages to reduce the problem of hysteresis by overshooting by about 2 mm when approaching diameters which are lower than the current diameter, and then moving up again by the same amount it overshot. Thus, every diameter is approached coming from lower values. As shown in Table 2, the observed values coming from lower and higher diameters agreed within the uncertainties, and thus the overshooting approached was successful in reducing the hysteresis that existed in the previous manual system.

8. Conclusion

In this paper, the Iris Mover, a motor-driver open source hardware system able to control the iris diaphragm apertures in a collinear laser spectroscopy beamline, was introduced. The Iris Mover system was divided into its three subsystems: electrical, mechanical, and firmware, and the assembly instructions, operation instructions, as well as design decisions for each of these subsystems were presented and discussed. The Iris Mover was found to be very accurate when used to vary the iris diameter in measurements taken in the range between 2.0 mm and 12.5 mm, and also observed to greatly reduce the hysteresis problem that existed in the manually controlled apparatus that preceded it. Furthermore, with its price estimated between 100 US$ and 150 US$, it is much cheaper than other similar solutions readily available on the market. Moreover, it was designed to fit the spatial constraints of the beamline, and not interfere with the various other systems and mechan-



Set Diameter (in mm): Enter input

Diameter Currently Saved (in mm): None

Current State: Homing
Current Diameter (in mm): 1.9
Error Status: No error
Input Status: Currently not accepting inputs

Set Interval: 1 s

```
2018-07-30 13:59:16 IrisMover1 interval:        1
2018-07-30 13:59:16 IrisMover1 reply:
2018-07-30 13:59:16 IrisMover1 reply:           <1 0 0>
2018-07-30 13:59:17 IrisMover1 reply:           <1 0 19>
```

Fig. 12. GUI used to control the Iris Mover.

Table 1
Expected and observed aperture values obtained by using the Iris Mover.

Expected (mm)	Observed (mm)	Absolute Difference (mm)
2.0 ± 0.6	1.7 ± 0.8	0.3 ± 1
4.5 ± 0.6	4.0 ± 0.9	0.5 ± 1
7.5 ± 0.5	7 ± 1	0.5 ± 1
10.0 ± 0.5	11 ± 1	1 ± 2
12.5 ± 0.7	13 ± 2	0.5 ± 2
12.5 ± 0.7	14 ± 2	1.5 ± 2
10.0 ± 0.5	12 ± 2	2 ± 2
7.5 ± 0.5	8 ± 2	0.5 ± 2
4.5 ± 0.6	4.8 ± 0.9	0.3 ± 1
2.0 ± 0.6	1.7 ± 0.8	0.3 ± 1

Table 2
Difference in diameter values when approaching from lower and higher values.

Diameter (mm)	Absolute Difference (mm)
2.0 ± 0.6	0 ± 1
4.5 ± 0.6	0.8 ± 1
7.5 ± 0.5	1 ± 2
10.0 ± 0.5	1 ± 2
12.5 ± 0.7	1 ± 3

ical parts that already exist. By following the design considerations outlined here for the Iris Mover and referring to the design files made open source in this project, other researchers can reproduce this system or develop similar controllers for apertures in beamlines or other optical systems.

In terms of future work and improvements to the system, it is possible to add a camera to the presented setup to monitor the iris diaphragm and provide diameter readings to the Arduino controller system. The iris diameter can be determined from image readings using an approach similar to the one described in Section 5.3 Firmware Subsystem. This way, the diameter of the Iris can be tracked at each instant, potentially improving accuracy.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We wish to acknowledge Dr. Kristian König (Michigan State University), Phillip Imgram (TU Darmstadt), and Tim Ratajczyk (TU Darmstadt) for the great help during the development, assembly, and calibration of the Iris Mover device and for reviewing the article's manuscript.

The University of Toronto Engineering Science Research Opportunities (ESROP - Global) and the TU Darmstadt International Research Experience (IREP) programs provided scholarships during the research.

Human and animal rights

This work does not involve the use of any human or animal subjects.

References

[1] P. Campbell, I.D. Moore, M.R. Pearson, Laser spectroscopy for nuclear structure physics, Prog. Part. Nucl. Phys. 86 (2016) 127–180, https://doi.org/10.1016/j.ppnp.2015.09.003.
[2] R. Neugart, J. Billowes, M.L. Bissell, K. Blaum, B. Cheal, K.T. Flanagan, G. Neyens, W. Nörtershäuser, D.T. Yordanov, Collinear laser spectroscopy at ISOLDE: new methods and highlights, J. Phys. G: Nucl. Part. Phys. 44 (6) (2017) 064002, https://doi.org/10.1088/1361-6471/aa6642.
[3] E. Riis et al, Lamb shifts and hyperfine structure in 6 Li + and 7Li+ : Theory and experiment, Phys. Rev. A 49 (1994) 207, https://doi.org/10.1103/PhysRevA.49.207.
[4] A. Krieger, W. Nörtershäuser, C.h. Geppert, K. Blaum, M.L. Bissell, N. Frömmgen, M. Hammen, K. Kreim, M. Kowalska, J. Krämer, R. Neugart, G. Neyens, R. Sánchez, D. Tiedemann, D.T. Yordanov, M. Zakova, Frequency-comb referenced collinear laser spectroscopy of Be+ for nuclear structure investigations and many-body QED tests, Appl. Phys. B 123 (1) (2017), https://doi.org/10.1007/s00340-016-6579-5.
[5] E. Riis, L.-U. Andersen, N. Bjerre, O. Poulsen, S.A. Lee, J.L. Hall, Test of the isotropy of the speed of light using fast-beam laser spectroscopy, Phys. Rev. Lett. 60 (2) (1988) 81–84, https://doi.org/10.1103/PhysRevLett.60.81.
[6] J. Krämer, K. König, C.h. Geppert, P. Imgram, B. Maaß, J. Meisner, E.W. Otten, S. Passon, T. Ratajczyk, J. Ullmann, W. Nörtershäuser, High-voltage measurements on the 5 ppm relative uncertainty level with collinear laser spectroscopy, Metrologia 55 (2) (2018) 268–274, https://doi.org/10.1088/1681-7575/aaabe0.
[7] S. Götte, K.-M. Knaak, N. Kotovski, H.-J. Kluge, G. Ewald, K.D.A. Wendt, Test of collinear spectroscopy for precise high-voltage determination, Rev. Sci. Instrum. 75 (4) (2004) 1039–1050, https://doi.org/10.1063/1.1651635.
[8] K. König et al, Rev. Sci. Instrum. (2020).
[9] P. Imgram, K. König, J. Krämer, T. Ratajczyk, B. Maaß, P. Müller, F. Sommer, W. Nörtershäuser, High-precision collinear laser spectroscopy at the Collinear Apparatus for Laser Spectroscopy and Applied Physics (COALA), Hyperfine Interact. 241 (1) (2020), https://doi.org/10.1007/s10751-019-1690-8.
[10] A. Gopal et al, Observation of energetic terahertz pulses from relativistic solid density plasmas, New J. Phys. 14 (2012), https://doi.org/10.1088/1367-2630/14/8/083012.
[11] Fukuda, Masao, K. Hatanaka, T. Yorita, H. Yamamoto, T. Saito, H. Ueda, H Tamura, M. Kibayashi, K. Nagayama, Y. Yasuda, Morinobu, Shigeru, Rcnp, Development of high-quality intense proton beam at the rcnp cyclotron facility. <https://accelconf.web.cern.ch/IPAC2011/papers/weps080.pdf>, 2011 (accessed 06.03.2020).
[12] Nobutaka Shimizu, Tetsuya Shimizu, Seiki Baba, Kazuya Hasegawa, Masaki Yamamoto, Takashi Kumasaka, Development of an online UV–visible microspectrophotometer for a macromolecular crystallography beamline, J. Synchrotron Radiat. 20 (6) (2013) 948–952, https://doi.org/10.1107/S0909049513022887.
[13] J.M. Pearce, Building research equipment with free, open-source hardware, Science 337 (6100) (2012) 1303–1304, https://doi.org/10.1126/science.1228183.
[14] Daniel K. Fisher, Peter J. Gould, Open-source hardware is a low-cost alternative for scientific instrumentation and research, Morern Instrum. 01 (02) (2012) 8–20.
[15] Gerardo M. Spinelli, Zach L. Gottesman, Jonathan Deenik, A low-cost Arduino-based datalogger with cellular modem and FTP communication for irrigation water use monitoring to enable access to CropManage, HardwareX 6 (2019) e00066, https://doi.org/10.1016/j.ohx.2019.e00066.
[16] Jinook Oh, Riccardo Hofer, W. Tecumseh Fitch, An open source automatic feeder for animal experiments, HardwareX 1 (2017) 13–21, https://doi.org/10.1016/j.ohx.2016.09.001.
[17] Carlos Sánchez, Paolo Dessì, Maeve Duffy, Piet N.L. Lens, OpenTCC: An open source low-cost temperature-control chamber, HardwareX 7 (2020) e00099, https://doi.org/10.1016/j.ohx.2020.e00099.