brgkepler deine Wahl · deine Zukunft

Theorie und Bau von Teleskopnachführungen

Entwicklung und Konstruktion einer computergesteuerten Dobson-Nachführung

Robin Fitz

Graz, 26.02.2026

8bdgi

Bundesrealgymnasium Kepler

Keplerstraße 1, 8020, Graz

Betreut von: Mag. Norbert Siller

Abstract

Ein deutlicher Zuwachs an Beliebtheit sowie weitreichende Entwicklungen in der Computertechnik führten dazu, dass sich in den letzten Jahrzehnten im Bereich der Astrofotografie erhebliche Fortschritte zeigten. Diese sind insbesondere im Amateurbereich zu beobachten. Die vorliegende Arbeit befasst sich mit dem Aufbau sowie der Steuerung von Teleskopnachführungen, welche ein fundamentales Instrument für die Fotografie von Himmelsobjekten darstellen. Die beim Bau einer Nachführung auftretenden Herausforderungen sowie deren Lösungsmöglichkeiten werden analysiert und erläutert. Einen wesentlichen Teil der Arbeit bildet die Analyse und Dokumentation eines Selbstbauprojektes, in dem versucht wurde, ein Dobson-Teleskop mit einer computergesteuerten Nachführung auszurüsten. Zudem werden die technischen Aspekte des eigenen Projekts mit denen einer kommerziellen Lösung verglichen. Hieraus wird geschlossen, dass sich Unterschiede zwar nicht im Konstruktionsprinzip, dafür aber in der Genauigkeit der Herstellung ergeben.

Inhaltsverzeichnis

1. Einleitung

Die Amateurastronomie existiert als Hobby bereits seit langer Zeit. Die Entwicklung der modernen Technik ermöglichte jedoch einen enormen Zuwachs an Möglichkeiten für die Erforschung des Nachthimmels. Techniken wie die Astrofotografie mit modernen Kameras faszinieren Tag für Tag mehr Menschen für die Wunder der Astronomie. Dennoch sind für die heutigen Methoden wesentliche technische Kenntnisse erforderlich.

Eine der größten Herausforderungen bei der Beobachtung und Fotografie von Himmelsobjekten stellt die Erdrotation dar. Durch sie bewegen sich Sterne scheinbar kontinuierlich über den Himmel, und es kommt bereits nach kurzer Zeit zu einer Verschiebung des Bildausschnitts im Teleskop. Um diese Bewegung auszugleichen, wurden Nachführungen entwickelt. Je nach Bauart der Montierung unterscheidet sich deren Funktionsweise erheblich.

Ziel dieser Arbeit ist es deshalb, die Verschiedenheiten der Systeme darzustellen, während der Fokus auf die Nachführung bei einer Dobson-Montierung fällt. Dazu werden zunächst die astronomischen Grundlagen und die zusätzlichen Funktionen moderner Montierungen erläutert. Aufbauend darauf wird der Planungs- und Umsetzungsprozess des eigenen Projekts behandelt, einschließlich der aufgetretenen Schwierigkeiten und möglicher Lösungsansätze. Somit werden theoretische Grundlagen mit praktischer Ingenieurarbeit verbunden.

2. Teleskopmontierungen und ihre Koordinatensysteme

Unter einer Teleskopmontierung versteht man eine Konstruktion, meist auf einem Stativ oder einer festen Plattform verbaut, die es dem Beobachter ermöglicht, sein Teleskop zu befestigen und in Richtung verschiedener Himmelsobjekte auszurichten. Im Wesentlichen sind in der Astronomie zwei Arten von Montierungen und Koordinatensystemen in Gebrauch. Die Unterschiede der beiden Systeme in Art und Anzahl der Achsen sowie der Abhängigkeit auf Ort und Zeit werden im Folgenden dargestellt. Anschließend werden die mathematischen Grundlagen der Transformation der unterschiedlichen Koordinaten veranschaulicht. Zunächst sollen jedoch die Begriffe der Sternzeit und des Frühlingspunkts erläutert werden. In der folgenden Abbildung ist eine Übersicht der Koordinatensysteme gegeben, auf die später Bezug genommen wird.

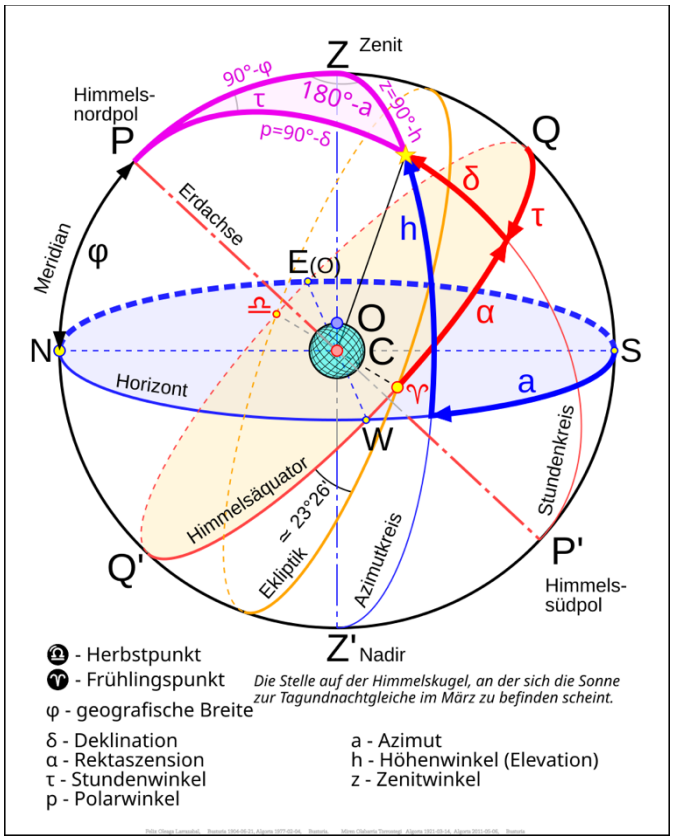

Abb. 1 Übersicht der Koordinatensysteme

2.1 Die Sternzeit und der Frühlingspunkt

Im Gegensatz zur konventionellen Sonnenzeit wird die *Sternzeit* Θ nicht in Relation zur Sonne, sondern zu den Sternen gemessen. 24^h in Sternzeit sind als die Zeitspanne definiert, in der die Sterne scheinbar einmal den Himmel umrunden.[1] In der Astronomie dienen die Sterne als Referenz, da sich ihre Positionen langfristig nur geringfügig ändern. Da sich die Sterne am Himmel pro Umrundung um ungefähr 3^m 56^s schneller bewegen als die Sonne, ergeben sich zwei verschiedene Arten der Zeitmessung.[2]

$$24^h\ Sonnenzeit = 24^h 3^m 56{,}55^s\ Sternzeit$$

$$24^h\ Sternzeit = 23^h 56^m 4{,}09^s\ Sonnenzeit$$

[1] vgl. Karttunen u. a., 2017, S. 34
[2] vgl. Karttunen u. a., 2017, S. 34f.

Der Referenzpunkt für den Tageszyklus der Sternzeit ist der Frühlingspunkt.[3] „Dieser ist definiert als einer der Schnittpunkte zwischen Sonnenbahn *(Ekliptik)* und Äquator, und zwar der Ort der Sonne zu Frühlingsbeginn.“[4]

Für den alltäglichen Gebrauch ist die Sternzeit jedoch ungeeignet, da sich die Sonne bewegt und die Sternzeit nicht mit dem täglichen Wechsel von Tag und Nacht übereinstimmt.[5]

2.2 Das Äquatoriale System

Mit dem äquatorialen Koordinatensystem können Positionen am Himmel genau definiert werden, ohne dabei von der Position des Beobachters oder der aktuellen Uhrzeit abhängig zu sein. Das Zentrum des Koordinatensystems ist der Mittelpunkt der Erde, wobei der Himmelsäquator die Grundebene bildet. Dieser ist die Projektion des Erdäquators auf den Himmel.[6] Der Frühlingspunkt bildet hier wie bei der Sternzeit den zentralen Referenzpunkt.[7] Die *Rektaszension* α verläuft entlang des Kreises des Himmelsäquators und wird meist in Zeiteinheiten gemessen. Eine volle Drehung um den Himmelsäquator beträgt 360° oder 24^h in Sternzeit. Die *Deklination* δ beschreibt den Abstand vom Himmelsäquator. Sie reicht von 0° bis 90° in Richtung des nördlichen Himmelspols und von 0° bis -90° in Richtung des südlichen Himmelspols. In Abbildung 1 ist in Rot ein äquatoriales Koordinatensystem veranschaulicht. Mit den beiden Koordinaten der *Rektaszension* α und der *Deklination* δ kann nun ein Punkt am Himmel beschrieben werden.[8] Als Beispiel kann der Stern Sirius mit den Koordinaten $\alpha = 6^h45^m8^s$ und $\delta = -16°43'37"$ genommen werden, wo α den Abstand vom Frühlingspunkt, und δ den Abstand vom Himmelsäquator beschreibt.

Äquatoriale Montierungen orientieren sich am äquatorialen Koordinatensystem und besitzen daher eine Rektaszensionsachse, sowie eine senkrecht darauf liegende Deklinationsachse. Diese sind in Abbildung 2 zu erkennen. Beim Verwenden einer solchen Montierung wird die Rektaszensionsachse so ausgerichtet, dass sie möglichst genau parallel zur Erdrotationsachse

[3] vgl. Hanslmeier, 2020, S.12
[4] Hanslmeier, 2020, S. 9
[5] vgl. Hanslmeier, 2020, S. 12f.
[6] vgl. Karttunen u. a., 2017, S. 17
[7] vgl. Hanslmeier, 2020, S. 9
[8] vgl. Karttunen u. a., 2017, S. 17 f.

liegt.[9] Hierfür ist meist eine kleine azimutale Plattform verbaut, welche im Bild ebenfalls zu sehen ist.

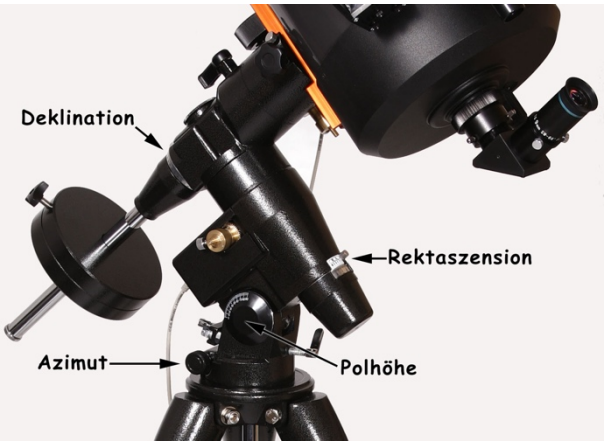

Abb. 2 Äquatoriale Montierung mit Beschriftung

Es gibt jedoch auch vereinfachte Ausführungen der äquatorialen Montierung, die zum Nachführen von kleinen, tragbaren Kameras und Objektiven verwendet werden. Auf eine Deklinationsachse wird der Einfachheit halber verzichtet. Daher wird die Kamera meist auf einem Kugelkopf befestigt, um sie dennoch frei bewegen zu können.[10]

2.3 Das Azimutale System

Aus der Sicht des Beobachters ist das natürlichste Koordinatensystem am Himmel das azimutale, beziehungsweise das horizontale System.[11] Mit den beiden Koordinaten *Höhe h* und *Azimut a* kann unkompliziert die Höhe und Richtung eines Objektes am Himmel bestimmt werden. Hierbei wird die Höhe h durch den Abstand eines Punktes vom Horizont nach oben (in Richtung Zenit) oder nach unten (in Richtung Nadir) definiert. Sie reicht von 90° bis -90°. Der Azimut wird dann von einem festen Punkt, meist dem Nord- oder Südpol entlang des Horizonts im Uhrzeigersinn gemessen und reicht von 0° bis 360°. In Abbildung 1 ist in Blau die Funktionsweise des azimutalen Systems veranschaulicht. Da hier der Grundkreis durch den Horizont des Beobachters definiert wird, ändern sich die Koordinaten allerdings mit einer

[9] vgl. Karttunen u.a., 2017, S. 60
[10] vgl. Jones, 2025
[11] vgl. Karttunen u. a., 2017, S. 16

Änderung der Position des Beobachters, sowie einer Änderung der Zeit, da sich Himmelsobjekte im Laufe der Zeit aufgrund der Erdrotation am Himmel scheinbar bewegen.[12] Um Koordinaten für den Beispielsstern Sirius angeben zu können, muss also ein Beobachtungsdatum, sowie ein Beobachtungsort gegeben sein. Die beiden Koordinaten $h = 25°02'18{,}6"$ und $a = 180°26'11{,}9"$ beschreiben nun die Position des Beispielssterns Sirius zum Zeitpunkt 00:00 Uhr am 1.1.2026 in Wien. Eine praktische Anwendung des azimutalen Koordinatensystems erweist sich im Vergleich zum äquatorialen System als sehr einfach. Da bei den angegebenen Koordinaten als Referenzpunkt für den Azimut der Nordpol verwendet wurde, kann man sich an diesem orientieren. Um den Stern Sirius zu finden, muss man sich vom Nordpol aus mithilfe eines Kompasses im Uhrzeigersinn um ungefähr 180° drehen, und der Stern ist am Himmel in ungefähr 25° Höhe sichtbar.

Azimutale Montierungen zeichnen sich durch ihren stabileren Aufbau aus, da hier keine aufwendige Konstruktion notwendig ist, die eine der Achsen mit der Polachse parallel legt. Die allermeisten sehr großen Teleskope werden azimutal montiert.[13]

Die im Jahr 1956 von John Dobson entwickelte Dobson-Montierung ist eine einfache Bauweise der azimutalen Montierung. Sie wurde für das Newton-Teleskop entwickelt und ist an vielen kostengünstigen Teleskopen für Amateurastronomen verbaut.[14] In der Abbildung rechts ist eine Dobson-Montierung zu sehen. Das Teleskop sitzt auf Rollen auf zwei tragenden Platten rechts und links, die wiederum auf einem Drehkreis montiert sind. So ist das Teleskop in Höhe und Azimut zu bewegen. Die Dobson-Montierung wird im späteren Verlauf der Arbeit näher behandelt.

Abb. 3: Kommerzielle Dobson-Montierung

[12] vgl. Karttunen u. a., 2017, S. 16f.
[13] vgl. Karttunen u. a., 2017, S. 61
[14] vgl. Karttunen u. a., 2017, S. 61

2.4 Trigonometrische Transformation der Koordinaten

Da in zum Beispiel Sternkatalogen Koordinaten eines Himmelsobjekts in Rektaszension und Deklination gegeben sind, ergibt sich das Problem, dass man dieses Objekt mit einer azimutalen Montierung nicht sehr leicht finden kann. Umgekehrt könnte man die Himmelsrichtung und den Azimutwinkel eines Objekts zu einer gewissen Zeit wissen, und möchte dieses Objekt katalogisieren. Hierfür werden Koordinatentransformationen benötigt. Für die Herleitung der Transformationsformeln werden sphärische Dreiecke verwendet, die auf einer Kugeloberfläche definiert sind. In einem derartigen Dreieck gelten der Sinus-, sowie der Cosinussatz.[15]

Das sogenannte astronomische Dreieck wird in Abbildung 1 in Violett dargestellt. Klassisch verbindet es den Zenit, den Himmelsnordpol und den Punkt, der momentan am Himmel betrachtet wird. Dadurch entstehen im Dreieck die Seiten $90° - \delta$, $90° - \varphi$ und $z = 90° - h$. Weiters sind die beiden Winkel τ und $180° - a$ bekannt. So kann ein Zusammenhang zwischen den beiden Systemen hergestellt werden. Mithilfe der folgenden Formeln können nun Koordinaten umgerechnet werden.[16]

$$sin\, z\, sin\, a = cos\, \delta\, sin\, t$$

$$cos\, z = sin\, \varphi\, sin\, \delta + cos\, \varphi\, cos\, \delta\, cos\, t$$

$$-\sin z \cos a = \cos \varphi \sin \delta - \sin \varphi \cos \delta \cos t$$

$$\cos \delta \sin t = \sin z \sin a$$

$$\sin \delta = \sin \varphi \cos z - \cos \varphi \sin z \cos a$$

$$\cos \delta \cos t = \cos \varphi \cos z + \sin \varphi \sin z \cos a$$

3. Teleskopnachführungen

Neben der Befestigung des Teleskops stellt die meist in die Montierung eingebaute Nachführung eine der fundamentalen Funktionen von Teleskopmontierungen dar. Die durch die Drehung der Erde verursachte scheinbare Bewegung von Himmelsobjekten sorgt dafür, dass ein

[15] vgl. Hanslmeier, 2020, S. 11
[16] vgl. Hanslmeier, 2020, S. 11

Himmelsausschnitt im Teleskop nie stillgehalten werden kann.[17] Vor allem bei hohen Brennweiten des Teleskops erschwert dies die visuelle Beobachtung und macht die Astrofotografie nahezu unmöglich, da bei den im Dunkeln erforderlichen langen Belichtungszeiten die Bewegung des Bildes zum Phänomen der Star Trails führt. Unter Star Trails oder Sternspuren versteht man die bei Langzeitbelichtungen entstehenden Lichtspuren der Sterne.[18]

In Abbildung 4 ist zu erkennen, wie Sterne im Laufe der Zeit scheinbar um den Himmelsnordpol rotieren.

Die Nachführung besitzt einen Mechanismus, der das Teleskop auf solche Weise bewegt, dass die Rotationsbewegung der Erde möglichst genau ausglichen wird.[19]

Abb. 4: Star Trails um den Himmelsnordpol (Eigenaufnahme)

3.1 Nachführungen bei Äquatorialen Montierungen

Da bei einer äquatorialen Montierung die Rektaszensionsachse parallel zur Polachse ausgerichtet wird, kann die Erdrotation mit einer konstanten Bewegung um nur diese eine Achse ausgeglichen werden. Ein solcher Aufbau eignet sich also hervorragend zum Nachführen und somit auch für die Astrofotografie. Die Präzision der Nachführung hängt jedoch stark davon ab, wie genau die Rektaszensionsachse beim Aufbau der Montierung ausgerichtet wurde.[20]

3.2 Nachführungen bei Azimutalen Montierungen

Das Nachführen bei azimutalen Montierungen ist um einiges komplexer, da das Teleskop in zwei Achsen gleichzeitig bewegt werden muss. Moderne Computersteuerungen erleichtern dies, da die Drehgeschwindigkeiten der Motoren der beiden Achsen in Echtzeit ausgerechnet werden kann.[21] Für diese Berechnungen sind jedoch Informationen wie die Position des Beobachters, die

[17] vgl. Hanslmeier, 2015, S. 12
[18] vgl. Sebastian, 2023
[19] vgl. Hanslmeier, 2015, S. 12f.
[20] vgl. Hanslmeier, 2015, S. 13
[21] vgl. Hanslmeier, 2020, S. 126

aktuelle Zeit, sowie die Position am Himmel, auf die das Teleskop zeigt, erforderlich. Deshalb müssen zu Beginn der Beobachtung Kalibrierungsverfahren durchgeführt werden. Ein solches Verfahren beinhaltet gewöhnlich das manuelle Ausrichten auf 1-3 Sterne, um dem System die absolute Position des Teleskops am Himmel zu geben.[22] Ziel dieser Verfahren ist es ebenfalls, die Schräglage der Montierung zu erheben, da eine Verwendung des azimutalen Systems impliziert, dass die Azimutachse der Montierung normal auf die Horizontebene liegt. Da dies nicht immer der Fall ist, muss die Computersteuerung die Schräglage korrigieren, um eine präzise Nachführung zu ermöglichen. Einige moderne Montierungen haben bereits einen elektronischen Neigungssensor. Somit genügt das manuelle Ausrichten auf nur einen Stern, da dem System die Neigung bereits bekannt ist.[23]

Da sich das Bildfeld eines Teleskops bei azimutaler Montierung nicht mit der Rotation der Erde dreht, ergibt sich als weiteres Problem, dass sich das Bildfeld mit der Zeit scheinbar um sich selbst dreht. Folglich erscheinen in Aufnahmen bei der Astrofotografie Sternspuren um den Mittelpunkt des Bildfeldes. Das Ausmaß der Rotation lässt sich von der Himmelsausrichtung und dem Breitengrad des Teleskops bestimmen.[24] Für weite Teile des Himmels ist der Effekt im Vergleich zur Erdrotation aber zumindest für etwas kürzere Belichtungszeiten vernachlässigbar.[25]

3.3 GoTo

Viele moderne Montierungen besitzen eine sogenannte GoTo-Funktion. Mit ihr kann die Montierung das Teleskop automatisch auf ein bestimmtes Himmelsobjekt aus einer Computerdatenbank ausrichten. Dies erspart einiges an Zeit und Aufwand bei der Beobachtung, da das gewünschte Objekt nicht mehr aufwendig per Hand gesucht werden muss.[26]

Ähnlich wie beim Nachführen mit einer azimutalen Montierung benötigt das System für die GoTo-Funktion die absoluten Koordinaten des Teleskops am Himmel. Eine äquatoriale Montierung muss hierfür zu Beginn der Beobachtung neben dem Ausrichten der

[22] vgl. californiaskys.com, 2021, Understanding Polar & GoTo Alignments
[23] vgl. Zermelo, 2021, Levelling alt-az mounts for goto operation
[24] vgl. Iovenitti, 2021, S. 3
[25] vgl. Hanslmeier, 2015, S. 12
[26] vgl. Hanslmeier, 2015, S. 19

Rektaszensionsachse noch entsprechend für GoTo kalibriert werden. Diese Prozedur ist, wie oben besprochen bei azimutalen Montierungen bereits zum Nachführen erforderlich. [27]

3.4 Autoguiding

Selbst bei professionellen Nachführungen tritt aufgrund von unvermeidlichen Ungenauigkeiten bei der Herstellung des Getriebes ein sogenannter *Periodic Error* auf, ein periodisch auftretender Fehler in der reibungslosen Nachführung. Dies führt wiederum zum Problem, dass bei der Astrofotografie leichte Sternspuren entstehen.[28]

Mithilfe einer zweiten Kamera, die oft auf einem zweiten, kleineren Teleskop angebracht wird, werden Abweichungen von der idealen Nachführungsgeschwindigkeit anhand der Bewegung der Sterne im Kamerabild dauerhaft überwacht. Sobald eine Ungenauigkeit in der Bewegung festgestellt wird, wird ein Impuls an die Nachführung gesendet, um den Fehler zu korrigieren. Diesen Prozess nennt man Autoguiding.[29]

4. Herausforderungen beim Eigenbau einer azimutalen Nachführung

Bevor wir den Bau von Nachführungen im Kontext unseres eigenen Projekts näher betrachten, erscheint es sinnvoll, die allgemeinen Herausforderungen bei solchen Arbeiten darzustellen. Um eine funktionelle Teleskopnachführung zu bauen, sind erweiterte Kenntnisse in den Bereichen Mechanik, Elektronik und Softwaretechnik nötig. Im Folgenden werde ich eine Übersicht über insbesondere die Thematiken geben, die für unser Projekt relevant sind.

4.1 Mechanische Aspekte

Neben der Stabilität der Montierung ist wohl der wichtigste Faktor die Präzision des Antriebs. Dieser muss Aufgrund der sehr geringen Geschwindigkeit, die für das Nachführen von Teleskopen notwendig ist, in beiden Achsen über ein Getriebe mit einer besonders hohen Übersetzung verfügen. Optimale Übersetzungen hängen von der Konfiguration der Motoren ab,

[27] vgl. californiaskys.com, 2021, Understanding Polar & GoTo Alignments
[28] vgl. Hanslmeier, 2015, S. 14
[29] vgl. Hanslmeier, 2015, S. 15

liegen aber oft bei einem Verhältnis von über 300:1. Die Ungenauigkeiten, die durch das Getriebe entstehen, müssen für eine reibungslose Nachführung minimiert werden.[30]

4.1.1 Implementierung von Schrittmotoren

Für Teleskopnachführungen werden im Wesentlichen zwei Arten von Motoren verwendet. Gleichstrom-Servomotoren und Schrittmotoren, wobei Schrittmotoren aufgrund ihrer kostengünstigen und einfachen Bauart etwas häufiger zu finden sind. Da wir diese auch in unserem eigenen Selbstbauprojekt verwenden, soll die Funktionsweise von Schrittmotoren in einer Teleskopnachführung im Folgenden verständlich gemacht werden.

Schrittmotoren eignen sich deshalb besonders für Anwendungen, wo exakte Bewegungen erforderlich sind, da sie sich nicht in kontinuierlicher Rotation bewegen, sondern in einzelnen, immer gleich langen Schritten.[31] Um mit diesen Motoren eine kontinuierliche Rotation zu erzeugen, sind spezielle Schrittmotortreiber erforderlich.[32] Besonders weit verbreitet sind Motoren mit 200 Schritten pro Umdrehung, oder 1,8° pro Schritt.[33] Da für viele Anwendungen eine solche Auflösung unzureichend ist, besitzen die meisten Treiber eine Funktion namens Microstepping. Mithilfe von Microstepping lassen sich einzelne Schritte in mehrere kleine Schritte unterteilen.[34] Eine Aufteilung in 32 Microsteps pro ganzem Schritt ist so problemlos möglich und ergibt eine Auflösung von $200 \cdot 32 = 6400$ Schritten pro Umdrehung. Manche Treiber bieten eine noch größere Unterteilung, jedoch sinkt dann die Genauigkeit zwischen den einzelnen Microsteps immens.[35]

Um die benötigte Übersetzung und Auflösung der Schrittmotoren für unser Projekt zu berechnen, müssen die Eigenschaften des später verwendeten Kamerasensors, sowie die Brennweite des Teleskops in Betracht gezogen werden. Da Kamerasensoren aus einzelnen Pixeln bestehen, darf die durch einen einzelnen Motorschritt verursachte Bewegung höchstens so groß sein, als der Winkel, den ein Pixel am Himmel abbildet. Andernfalls verschiebt sich das Bild auf der Kamera bei jedem Motorschritt um mehr als einen Pixel, was zu einer unscharfen Abbildung führt.

---

[30] vgl. OnStep, 2025, Drive Design
[31] vgl. Posch, 2025
[32] vgl. Jones, 1998
[33] vgl. Walter, 2016
[34] vgl. Jones, 1998
[35] vgl. OnStep, 2025

Um den Winkel zu berechnen, den ein Kamerapixel am Himmel abbildet, können wir die Größe eines Pixels durch die Brennweite des Teleskops dividieren:[36]

$$\theta_{rad} = \frac{s}{f}$$

Wobei s die Größe eines Pixels in mm, und f die Teleskopbrennweite in mm darstellt. Da dieser Winkel im Radiant gegeben ist, müssen wir ihn in Bogensekunden umrechnen, um ihn sinnvoll darstellen zu können. Da $1\ rad = 57{,}2958°$, $1° = 60'$ und $1' = 60"$ gilt, ergibt sich $1\ rad = 206265"$. So kann eine einfache Formel für die Berechnung der Bogensekunden pro Pixel aufgestellt werden:

$$\theta_{arcsec} = \frac{206265 \cdot\ s}{f}$$

Da unsere Kamera eine Pixelgröße von 0,00594 mm, und unser Dobson-Teleskop eine Brennweite von 1200 mm hat, ergibt sich aus der obigen Gleichung eine Winkelgröße von ca. 1,021 Bogensekunden pro Pixel. Demnach darf sich das Teleskop für ein scharfes Bild mit einem Schritt nicht weiter als 1,021 Bogensekunden bewegen. Im besten Fall ist die Auflösung unseres Antriebs jedoch so hoch, dass später durch ein Autoguiding-System Korrekturen um mehrere Motorschritte erfolgen können, ohne dabei das Teleskop um mehr als einen Kamerapixel zu bewegen.

Wenn unsere Motoren wie zuvor besprochen mit 6400 Schritten pro Umdrehung arbeiten, liegt der Winkel eines einzelnen Schritts bei $\frac{360°}{6400} = 0{,}05625°$. Umgerechnet in Bogensekunden liegt der Winkel bei 202,5". Will man nun eine Auflösung von 4 Schritten pro Pixel, oder 0,25 Bogensekunden pro Schritt, so muss der Antrieb über ein Getriebe mit einer Übersetzung von ungefähr 1:800 verfügen, da $\frac{200}{800} = 0{,}25$. Die Herstellung eines solchen Getriebes in angemessener Qualität stellt vor allem im Amateurbereich eine große Herausforderung dar, da meist der Zugriff zu professionellen Maschinen fehlt. Faktoren wie Spiel und Steifigkeit spielen hier eine große Rolle.[37]

[36] vgl. NASA, S. 1f.
[37] vgl. Kellerer, 2024, Designing a Gearbox for a Star Tracking Mount

4.1.2 Übersicht unterschiedlicher Getriebearten

Um die hohen Übersetzungen zu erreichen, werden in Teleskopmontierungen verschiedene Arten von Getrieben verwendet. Diese unterscheiden sich in ihren möglichen Übersetzungen, der Steifigkeit, der Effizienz sowie der Gleichmäßigkeit der Kraftübertragung.[38]

Klassische Teleskopmontierungen verwenden Schneckengetriebe. Dessen Hauptvorteil ist die hohe Übersetzung, die in einer einzigen Stufe möglich ist. Allerdings ist die Präzision in der Herstellung besonders wichtig, um eine gleichmäßige Übertragung zu ermöglichen. Zudem ist die Effizienz eines solchen Aufbaus sehr niedrig. Eine weitere Lösung ist der Riemenantrieb. Dieser wird häufig in Kombination mit anderen Übertragungsmechanismen verwendet und bietet hohe Effizienz. In professionellen Systemen sind des Öfteren Reibungsantriebe oder Direktantriebe zu finden. Diese beiden Konzepte sind Aufgrund der hohen Gleichmäßigkeit und dem aufgrund dem Mangel traditioneller Zahnräder praktisch nicht vorhandenen Spiel sehr beliebt. Die Herstellung und Entwicklung ist jedoch sehr aufwendig.[39]

4.2 Elektronische Aspekte

Betrachtet man den Nachführungsbau aus einer elektrotechnischen Perspektive, so fällt auf, dass einige Bauteile und Schaltungen verbaut werden müssen, um die entsprechende Steuerung durch Software zu ermöglichen.

4.2.1 Steuerung durch Mikrocontroller

Die Rolle eines Schrittmotortreibers besteht rein daraus, einzelne Impulse in Motorschritte umzuwandeln. Dabei bewegt der Treiber bei jedem empfangenen Impuls den Motor um einen Schritt. Aufgrund von dieser Funktionsweise müssen oft mehrere Hundert Impulse pro Sekunde an den Treiber gesendet werden, um eine kontinuierliche Rotation zu erzeugen. Infolgedessen wird ein Mikrocontroller benötigt, der die nötige Anzahl an Impulsen für eine Bewegung generiert.[40]Dieser wird auch als zentraler Steuerungscomputer für die Montierung verwendet, um Funktionen wie GoTo oder Autoguiding zu ermöglichen.[41] Häufig sind hierfür verschiedene

[38] vgl. Melsheimer, Comparing Telescope Drive Technologies
[39] vgl. Melsheimer, Comparing Telescope Drive Technologies
[40] vgl. Oriental motor, Stepper Motors, Stepper Motor Drivers and Controllers
[41] vgl. OnStep, 2025, Home

Arduino-Module in Gebrauch.[42] Für die Koordinateneingabe bei GoTo, oder für fundamentale Funktionen wie das Ein- und Ausschalten der Nachführung muss es eine Möglichkeit geben, mit dem System zu kommunizieren. Hierfür kann ein PC oder Smartphone über verschiedene Wege verbunden werden. Für einen stabilen und zuverlässigen Anschluss besteht die Möglichkeit, einen PC über ein USB-Kabel mit dem Mikrocontroller zu verbinden.[43] Will man etwas mehr Flexibilität, so können zusätzliche Module angebracht werden, die die Verbindung über WLAN oder Bluetooth ermöglichen.[44]

4.2.2 Stromversorgung

Neben der Steuerung durch Mikrocontroller ist die entsprechende Stromversorgung ebenfalls von großer Wichtigkeit. Diese kann sowohl durch verschiedene Arten von Akkus als auch durch ein konventionelles Netzteil erfolgen.[45] Die Stromspannung muss den Anforderungen der Motoren und Treibern entsprechend gewählt werden, liegt jedoch meist zwischen 12 V und 24 V.[46] Da die Steuerungsmodule und Motoren unterschiedliche Spannung benötigen, ist oft ein Gleichstrom-Abwärtswandler nötig, um die höhere Motorspannung für den Mikrocontroller brauchbar zu machen.[47] Wird ein Akku verwendet, so muss die nötige Kapazität anhand des Stromverbrauchs der Montierung ebenfalls kalkuliert werden. Als Beispiel hierfür kann ein Antrieb bei einer azimutalen Montierung genommen werden. Angenommen, die beiden Motoren verbrauchen beim Nachführen bei einer Spannung von 24 V jeweils 1,4 A. Zusätzlich verbraucht die restliche Elektronik 0,2 A, sodass insgesamt eine Last von 3 A entsteht. Um mit dieser Montierung eine Laufzeit von 4 Stunden zu erreichen, muss ein Akku bei einer Spannung von 24 V eine Kapazität von mindestens 12 Ah haben, da $3\ A \cdot 4\ h = 12\ Ah$.

4.3 Softwaretechnische Aspekte

Abgesehen von der Hardware gehört zu einer betriebsbereiten Nachführungssteuerung ebenfalls eine Software, die letztendlich alle Funktionen ermöglicht. Zum einen müssen bei der Verwendung von Schrittmotoren die Treiber mit regelmäßigen Schrittimpulsen versorgt werden.

---

[42] vgl. OnStep, 2023, Basic DIY Controller Examples
[43] vgl. OnStep, 2025, Connections and Applications
[44] vgl. OnStep, 2025, Smart Web Server (Ethernet and WiFi)
[45] vgl. OnStep, 2023, Basic DIY Controller Examples
[46] vgl. OnStep, 2025, Drive Design
[47] vgl. OnStep, 2023, Basic DIY Controller Examples

Die Impulsfrequenz muss anhand der nötigen Motorgeschwindigkeiten berechnet werden.[48] Diese sind bei einer azimutalen Montierung ebenfalls in Echtzeit für beide Achsen zu kalkulieren. Ist eine GoTo-Funktion gewünscht, so muss die Software über eine Datenbank von Himmelsobjekten verfügen. Da diese Objekte in äquatorialen Koordinaten gegeben sind, müssen Koordinatentransformationen angewendet werden, um für die azimutale Montierung von Nutzen zu sein.

Das selbstständige Schreiben eines solchen Steuerungsprogramms ist von großer Komplexität. Es gibt jedoch die fertige Software OnStep, welche speziell für den Selbstbau von Nachführungen geschrieben wurde. OnStep kann auf einen Mikrocontroller geladen werden und unterstützt eine große Auswahl an Funktionen. Die Steuerung von Schrittmotoren ist ebenfalls bereits integriert.[49]

5. Dokumentation des eigenen Projekts

In einer gemeinsamen Arbeitsgruppe haben wir uns mit dem Versuch beschäftigt, eine Nachführung mit GoTo-Funktion für ein Skywatcher Dobson-Teleskop zu bauen. Eine Übersicht des Planungs- und Bauprozesses gibt einen Einstieg in unsere Vorhaben. Näher wird auf die technischen Herausforderungen, sowie auf deren potenziellen Lösungsmöglichkeiten eingegangen.

5.1 Projektidee und Anforderungen

Die ursprüngliche Idee für den Selbstbau einer Nachführung stammte von unseren beeindruckenden Erfahrungen bei der visuellen Beobachtung, die wir fotografisch festhalten wollten. Bei der Brennweite von 1200 mm unseres Skywatcher Dobson-Teleskops ergab sich jedoch bereits nach wenigen Sekunden Belichtungszeit das Problem der Sternspuren. Weiters war es oft sehr zeitaufwendig, Himmelsobjekte manuell zu finden. So hatten wir uns zum Ziel gesetzt, unsere Dobson Montierung umzubauen, um die Astrofotografie zu vereinfachen und die Bedienung des Teleskops zu automatisieren.

[48] vgl. Oriental motor, Stepper Motors, Stepper Motor Drivers and Controllers
[49] vgl. OnStep, Home

Diese Nachführung sollte präzise genug sein, um die Fotografie von Deep-Sky Objekten wie Nebeln oder Galaxien zu ermöglichen. Mithilfe eines kleinen Steuerungscomputers sollten Himmelsobjekte ausgewählt werden können, sodass das Teleskop sich automatisch auf diese ausrichtet. Die Konstruktion musste transportabel sein und die Stromversorgung musste über einen Akku erfolgen, sodass das Teleskop auch an abgelegenen Beobachtungsstandorten eingesetzt werden konnte.

Zusätzlich zu den technischen Anforderungen hatten wir uns einen weiteren Grundsatz festgelegt: Die Kosten für das Projekt sollten möglichst gering bleiben, und in jedem Fall unter dem Preis einer kommerziellen Lösung.

5.2 Planung und Materialien

Anfänglich hatten wir es uns zum Ziel gesetzt, möglichst schnell einen funktionsfähigen Prototypen zu bauen. Eine solche Herangehensweise sollte uns ermöglichen, auftretenden Problemen frühzeitig auf den Grund zu gehen, und diese lösen zu können. Allerdings wollten wir darauf achten, unser bestehendes Dobson-Teleskop nicht auf eine zu destruktive Art zu bearbeiten, damit wir bei Fehlkonstruktionen ohne große Einschränkungen wieder von vorne starten konnten.

Deshalb hatten wir uns dazu entschieden, die Getriebe für die beiden Achsen vorerst aus dem Material ASA mit unserem 3D-Drucker zu drucken. Erst spätere Ausführungen sollten aus Aluminium mit einer CNC-Fräse hergestellt werden. Aufgrund der konstruktionstechnischen Komplexität von höheren Übersetzungen sollten die Plastikgetriebe zu Beginn aus einem Schneckengetriebe und einem weiteren Zahnradgetriebe mit einer Übersetzung von insgesamt 120:1 bestehen. Unsere Erwartungen für den provisorischen Antrieb waren gering, wir wollten uns aber zunächst noch nicht auf mechanische Genauigkeit konzentrieren. Als Motoren verwendeten wir NEMA-23 Schrittmotoren mit DM556 Schrittmotortreibern. Diese Kombination versorgte den Antrieb mit hervorragender Leistung und Präzision, selbst bei hohen Microstepping-Werten. Zudem sind solche Motoren und Treiber sehr weit verbreitet und deshalb preiswert. Als Mikrocontroller verwendeten wir ein ESP32-Entwicklungsmodul, da diese im privaten Maker-Bereich ebenfalls von großer Beliebtheit sind. Die bereits integrierten WLAN- und Bluetooth-Funktionen sorgen zusätzlich für flexible Konnektivität. Da wir das Teleskop auch

an Standorten ohne vorhandenen Stromanschluss verwenden wollten, hatten wir uns dazu entschieden, einen Lithium-Ionen-Akku aus einzelnen 18650-Batteriezellen zu bauen. So würden wir selbst über die optimale Spannung und Kapazität entscheiden können.

Da uns anfänglich die zuvor genannte Software OnStep nicht bekannt war, hatten wir vor, eine eigene Softwarelösung zu entwickeln. Die grobe Strukturierung dieser sollte wie folgt aussehen. Auf dem ESP32 Mikrocontroller läuft ein rudimentäres Programm, welches lediglich Befehle über Motorgeschwindigkeiten empfängt, und diese in Impulse für die Motortreiber umwandelt. Diese Befehle werden über eine USB-Verbindung von einem Laptop gesendet. Auf dem Laptop läuft ein weiteres Programm, welches die Berechnungen für die Motorgeschwindigkeiten übernimmt und uns ermöglicht, die Montierung zu steuern. Nachdem wir über Onstep erfuhren, stellte sich allerdings heraus, dass ein eigenes Programm gar nicht notwendig war, da Onstep als fertige Lösung alle von uns gewollten Funktionen unterstützte. Dennoch werde ich in Abschnitt 5.3.1 unseren Fortschritt bei der eigenen Softwareentwicklung darstellen.

Der Einfachheit halber versuchten wir, einen möglichst großen Anteil der bestehenden Dobson-Montierung beizubehalten. Jedoch erschien es uns sinnvoll, eines der zwei seitlichen tragenden Elementen, wie in Abbildung 5 ersichtlich, durch eine größere hölzerne Platte zu ersetzen. So würde mehr Platz für den Antrieb der Höhenachse zur Verfügung stehen. Der Antrieb für die Azimut-Achse sollte in den Innenbereich der Montierung eingebaut werden, zwischen den beiden zuvor genannten seitlichen Elementen.

Abb. 5: Seitliche Ansicht der Dobson-Montierung

5.3 Umsetzung

Die Umsetzung stellte sich im Laufe des Projekts als immer schwieriger heraus. Unsere hohen Anforderungen an die Präzision des Antriebs sorgten dafür, dass zunehmend Herausforderungen auftraten. Insbesondere die Planung und Konstruktion des Getriebes brachte einige

Schwierigkeiten mit sich. Aufgrund der Komplexität des Gesamtsystems ist es uns bislang noch nicht gelungen, einen funktionsfähigen Prototypen fertigzustellen. Nichtsdestotrotz werde ich im Folgenden unsere Fortschritte in der Softwareentwicklung und der Konstruktion des Antriebs darlegen. Aspekte wie die Stromversorgung wurden von uns noch nicht behandelt.

5.3.1 Implementierung der Software

Da wir die präzise Steuerung und Bewegung der Motoren für das fundamentalste Element des Programms hielten, hatten wir uns zu Beginn auf das Programm für den ESP32 konzentriert. Dieses schrieben wir in der Arduino Entwicklungsumgebung in der Programmiersprache C++. Um für die Motortreiber sauber und genau Pulse generieren zu können, verwendeten wir die Softwarebibliothek FastAccelStepper[50]. Diese ermöglichte uns die gleichzeitige Steuerung von mehreren Schrittmotoren mit Geschwindigkeits- und Beschleunigungswerten. Nach praktischer Überprüfung der Funktionsfähigkeit des ESP32 als Impulsgenerator für die Motortreiber stellte sich heraus, dass sich dieser ausgezeichnet als solcher eignet.

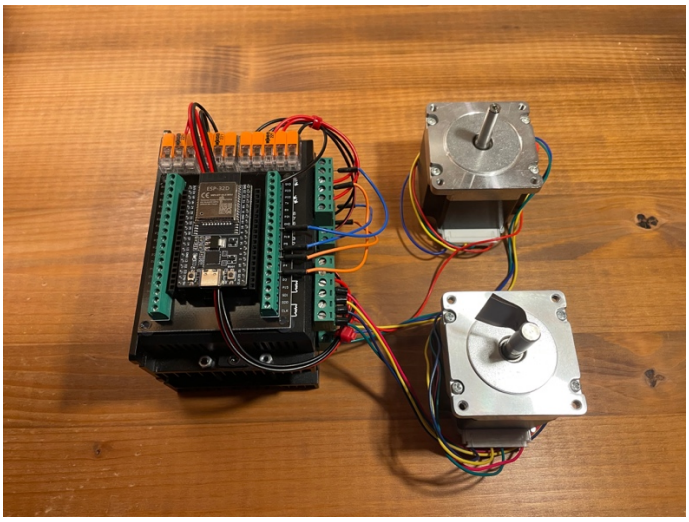

Abb. 6: Testaufbau der Schrittmotoren mit Treibern und Mikrocontroller

Das konnten wir mithilfe eines kurzen Testprogramms und einem temporären Aufbau der Schrittmotoren mit ihren Treibern feststellen. Dieser ist in der obigen Abbildung zu erkennen. Links befinden sich die beiden Treiber, der ESP32 Mikrocontroller sitzt darauf. Rechts zu sehen sind die beiden Schrittmotoren. Im Programmbeispiel in Abbildung 7 ist die Festlegung der

[50] s. https://github.com/gin66/FastAccelStepper

Motorwerte anhand der beiden Befehle *setSpeedInHz()* und *setAcceleration()* zu sehen. Mit dem Befehl *moveTo()* kann anschließend der Motor bewegt werden, wobei die komplexe Impulsgeneration von der FastAccelStepper-Bibliothek übernommen wird.

```
  // Geschwindigkeit und Beschleunigung festlegen
  altstepper->setSpeedInHz(3200); // in steps/sec
  altstepper->setAcceleration(10000); // in steps/sec^2
  // Position des Motors zu Beginn auf 0 setzen
  altstepper->setCurrentPosition(0);
}
// Schleife für die Motorbewegung
void loop() {
  // Motor nach Schrittposition 0 bewegen
  altstepper->moveTo(0);
  while (altstepper->isRunning()) {
  }
  // Motor nach Schrittposition 3200 bewewgen
  altstepper->moveTo(3200);
  while (altstepper->isRunning()) {
  }
}
```

Abb. 7: Ausschnitt eines einfachen Arduino-Programms zur Bewegung eines Schrittmotors

Um nun für unsere Nachführung von Nutzen zu sein, müssen diese Werte von einem externen Computer empfangen und bearbeitet werden können. Für diesen Zweck hatten wir uns aufgrund ihrer Einfachheit und Zuverlässigkeit für eine reguläre serielle Verbindung über ein USB-Kabel entschieden. Um die serielle Kommunikation zu implementieren, mussten wir dem Programm nur ein paar weitere Zeilen hinzufügen. Eine simple Funktion, die einen Geschwindigkeitswert aus einer seriellen Nachricht herausliest, ist in Abbildung 8 zu erkennen. Diese kann während des Betriebs der Nachführung in regelmäßigen Abständen aufgerufen werden.

```
void SerialLoop() {
  // Falls keine neue Nachricht vorhanden, Schleife abbrechen
  if (!Serial.available()) return;
  // Befehl in Variable "line" speichern
  String line = Serial.readStringUntil('\n');
  line.trim();
  // Falls Befehl leer, Schleife abbrechen
  if (line.length() == 0) return;
  // Falls Befehl mit "SPD" beginnt, folgenden
  // Wert in Variable "newSpeed" speichern
  if (line.startsWith("SPD")) {
    float newSpeed = line.substring(3).toFloat();
    // Richtung der Bewegung in Variable "alt_posdir" speichern
    if (newSpeed < 0) {
      alt_posdir = false;
    }
    else {
      alt_posdir = true;
    }
    // Motorgeschwindigkeit auf neue Geschwindigkeit setzen
    alt_speed = newSpeed;
    altstepper->setSpeedInHz((uint32_t)abs(alt_speed));
    // Mit Bestätigung der neuen Geschwindigkeit antworten
    Serial.print("OK SPD ");
    Serial.println(alt_speed);
  }
```

Abb. 8: Programmausschnitt zur Auslesung von Befehlen von einer seriellen Verbindung

Mithilfe des Befehls *Serial.available()* wird überprüft, ob eine neue Nachricht vorhanden ist. Falls nicht, wird die Schleife abgebrochen. Falls doch, wird die neue Nachricht in einer Variable gespeichert. Mit einer *if()* Abfrage wird festgestellt, um welche Art von Befehl es sich handelt. Im Beispiel wird der Befehl *SPD* behandelt, welcher dazu dient, die Geschwindigkeit, sowie die Richtung des Motors festzulegen. Nach einer Erweiterung um eine Abfrage für den Beschleunigungsparameter, sowie weitere Abfragen für den Motor der zweiten Achse ist das Programm einsatzbereit.

Als größere Herausforderung stellte sich das Schreiben des eigentlichen Steuerungsprogramms heraus. Dieses wollten wir in der populären Programmiersprache Python schreiben, da es für die von uns benötigten Berechnungen bereits leicht zu implementierende Softwarebibliotheken wie beispielsweise Astropy[51] gab. Bevor wir uns mit den Nachführungstechnischen Kalkulationen beschäftigten, wollten wir durch ein weiteres Testprogramm sicherstellen, dass wir eine

[51] s. https://www.astropy.org/index.html

funktionsfähige Kommunikation mit dem ESP32 herstellen konnten. Im folgenden Ausschnitt unseres Python-Programms ist zu sehen, wie ein Geschwindigkeitsbefehl aus der Konsole ausgelesen werden, und anschließend an die Funktion *send_command()* weitergegeben werden kann.

```
# Neuen Befehl in Variable "user_input" speichern
user_input = input("> ").strip()
if not user_input:
    continue
# Befehl in Art und Wert aufteilen
tokens = user_input.split()
# Falls Befehl mit "SPD" beginnt, Wert an
# Funktion "send_command" senden
if tokens[0].lower() == "spd" and len(tokens) == 2:
    try:
        speed = float(tokens[1])
        send_command(f"SPD {speed}")
    except ValueError:
        print("Invalid speed value")
```

Abb. 9: Programmauschnitt zur Auslesung von Geschwindigkeitsbefehlen aus der Python-Konsole

Wie in Abbildung 9 ersichtlich, ruft das Programm, nachdem ein bestimmter Befehl in die Konsole eingegeben wird, die Funktion *send_command()* (s. Abbildung 10) mitsamt dem Befehl auf. Diese dient dazu, den Befehl an den Arduino über die serielle Verbindung weiterzuleiten. Des Weiteren wird die Antwort des ESP32 ausgegeben, sodass bestätigt werden kann, dass der Befehl angekommen ist.

```python
def send_command(command: str):
    # Befehl an ESP32 senden
    command = command.strip()
    if not command:
        return
    ser.write((command + "\n").encode("ascii"))
    print(f">>> {command}")
    time.sleep(0.05)
    # Antwort ausgeben
    while ser.in_waiting:
        response = ser.readline().decode("ascii", errors="replace").strip()
        print(f"<<< {response}")
```

Abb. 10: Python-Funktion zum Senden von Befehlen an den ESP32

Um die Funktionsweise des Programms besser darstellen zu können, ist folglich ein Ausschnitt aus einem Terminal auf einem Laptop gegeben, indem das Python-Programm geöffnet wurde. Zu Beginn bestätigt das Programm, dass die serielle Verbindung eröffnet wurde, anschließend wird eine Liste der möglichen Befehle ausgegeben. Nun kann man in Echtzeit die Geschwindigkeit des Motors beliebig verändern. Im Beispiel befielt *SPD 3200* dem Arduino, den Motor im Uhrzeigersinn mit einer Geschwindigkeit von 3200 Schritten pro Sekunde zu bewegen.

```
robinfitz@Robins-MacBook-Pro Arduino serial test % python3 serialcom\ test.py
Opening serial connection...
Serial connection established.

Enter commands:
  SPD <value>   (speed, +/- steps/sec)
  quit          (exit)

> SPD 3200
>>> SPD 3200.0
<<< OK SPD 3200.00
> quit
Stopping motor...
>>> SPD 0
<<< OK SPD 0.00
Serial connection closed.
```

Abb. 11: Terminal-Fenster mit laufendem Python-Programm

Zu einer weiteren Entwicklung unserer eigenen Software kam es nicht, da uns das Programm OnStep bekannt wurde, welches ebenfalls mit einem ESP32-Mikrocontroller verwendet werden

kann. Die Steuerung unserer Schrittmotoren, sowie alle gängigen Funktionen einer Nachführung wie GoTo und Autoguiding sind bereits vollkommen integriert. Um das Programm an unsere Montierung anzupassen, mussten wir vor dem Hochladen nur einige Konfigurationsdateien ausfüllen. Somit hatten wir eine einsatzbereite Steuerungssoftware.

5.3.2 Konstruktion des Antriebs

Zunächst begannen wir mit dem scheinbar schwierigeren Antrieb für die Höhenachse. An das Teleskop, welches ursprünglich nur per Hand zu bewegen war, mussten wir unseren Motorantrieb befestigen. In der Abbildung rechts ist zu erkennen, wie wir hierfür ein mittiges Loch in das Teleskoprohr bohrten, an welches wir mit Muttern eine Gewindestange befestigten. Diese führte durch das seitliche Holzbrett, auf dessen anderen Seite sich das Getriebe befand. Wie zuvor besprochen, wollten wir zuerst versuchen, ein Schneckengetriebe aus Plastik auf unserem 3D-Drucker herzustellen. Auf dem Computer entwickelten wir mittels dem Programm Autodesk Fusion 360 die Zahnräder und die Schneckenwelle, sowie eine Konstruktion, mit der die Achsen auf dem Brett befestigt werden konnte. Nach einigen Testversuchen wurde jedoch deutlich, dass dieses Getriebe sehr ungenau ist. Selbst nach sorgfältiger Justierung der Einstellungsparameter für den 3D-Druck waren die Oberflächen der Zahnräder zu rau, um eine gleichmäßige Bewegung zu ermöglichen. Dennoch fertigten wir einen Prototypen an, mit dem Wissen, später auf andere Materialien und Fertigungsprozesse umsteigen zu müssen.

Abb. 12: Bohrung durch das Teleskoprohr

Eine der Schneckenwellen ist in Abbildung 13 während dem Druckvorgang zu erkennen. Die vertikale Ausrichtung, sowie eine besonders geringe Schichthöhe sollen für eine möglichst glatte Oberfläche sorgen. Das vorläufige Plastikgetriebe besteht aus einer Wellenübersetzung von 60:1 und einer weiteren Zahnradübersetzung von 2:1. In Abbildung 14 ist die Konstruktion zu sehen, wie sie am Teleskop befestigt ist. Somit war es uns letztendlich möglich, das Teleskop mit dem Getriebe zu bewegen. Allerdings litt der Antrieb unter starker Verzögerung. Wir konnten am Schneckenrad mehrere Zentimeter weit drehen, bevor sich das Teleskop bewegte.

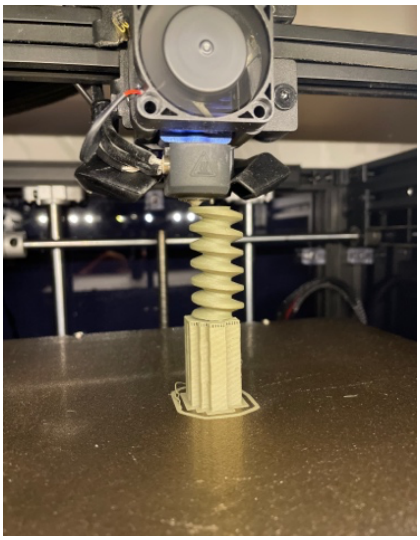

Abb. 13: Schneckenwelle im 3D-Drucker

Wir führten dieses Problem auf die Art zurück, wie wir die Plastikgetriebe, sowie das Teleskop selbst mit den Achsen verbunden hatten. Alle Komponenten waren lediglich mit Muttern von beiden Seiten an die Gewindestangen geschraubt worden. Da wir die Muttern aufgrund der Instabilität der Plastikzahnräder nicht ausreichend festziehen konnten, führte dies zu lockeren Verbindungen.

Abb. 14: Seitliche Ansicht des Getriebes der Höhenachse

Lösungen für die derzeitigen Probleme haben wir bislang noch nicht entwickelt. Mit dem Antrieb für die Azimut-Achse haben wir uns ebenfalls noch nicht befasst.

5.4 Herausforderungen und Lösungsansätze

Der Übersicht unseres Fortschritts zufolge werde ich einige der Herausforderungen zusammenfassen, und potenzielle Lösungen vorschlagen.

Die ausgeprägteste Schwachstelle unseres Baus ist wohl das Getriebe, welches momentan für eine für die Astrofotografie taugliche Nachführung absolut unzureichend ist. Die kleinen Rillen und Imperfektionen, die beim 3D-Druck im Amateurbereich entstehen, führten bei unserem Getriebe zu einer ungleichmäßigen Kraftübertragung. Dies würde wiederum die Entstehung von Sternspuren bewirken. Eine plausible Lösung wäre die Herstellung eines Getriebes aus Metall

mithilfe einer CNC-Fräse, wobei hier einige Kosten anfallen würden. Die Verbindung zum Teleskop an einem einzigen Bohrloch ist eine weitere Behinderung an der optimalen Steifigkeit des Antriebs. Hier müsste eine Art Flansch an mehreren Punkten an das Teleskoprohr geschraubt werden, an welches wiederum die Achse befestigt werden würde.

Die Komplexität des Antriebs in der Azimut-Achse ist ebenfalls nicht zu unterschätzen. Ein erheblicher Motordrehmoment ist erforderlich, da das gesamte Gewicht der Montierung bewegt werden muss, um das Teleskop nach rechts und links drehen zu können. Um diesen Effekt zu mindern, muss für eine möglichst geringe Reibung zwischen den unteren beiden runden Platten gesorgt werden, auf denen sich das Teleskop bewegt.

5.5 Aktueller Stand

Trotz der großen Herausforderungen, die uns bisher begleitet haben, arbeiten wir weiterhin am Bau unserer Montierung. Dank OnStep müssen wir uns nicht mehr darum kümmern, ein komplexes Steuerungsprogramm zu schreiben. Einige Verbesserungen im Bereich des Antriebs sind bereits in Arbeit. Unter anderem beschäftigen wir uns momentan mit der CAD-Konstruktion von einem Verbindungsflansch, welches wir an das Teleskoprohr befestigen werden.

Die Dauer bis zur Vollendung ist derzeit noch nicht vorhersehbar, dennoch arbeiten wir konsequent an der weiteren Optimierung und Umsetzung der verbleibenden Komponenten.

6. Vergleich mit einer kommerziellen Lösung aus dem Amateurbereich

Um die Eigenschaften unserer Konstruktion aus einer anderen Perspektive darstellen zu können, werde ich im Folgenden einen kurzen Vergleich mit einem kommerziellen computergesteuerten Dobson-Teleskop der Marke Sky-Watcher geben. Konkret geht es um das Stargate 16“. Die Montierung dieses Teleskops ist mit einem GoTo-System mit dem Markennamen SynScan ausgestattet.

Der Website[52] von Sky-Watcher sind einige Informationen über das Antriebskonzept zu entnehmen. Auffallend ist, dass nur geringfügige Unterschiede zu unserem eigenen Projekt festzustellen sind. Es wurde ebenfalls auf Schrittmotoren mit einer Grundauflösung von 200 Schritten pro Umdrehung gesetzt. Jedoch wird eine deutlich höhere Microstepping-Reduktion verwendet. Anstatt der 32 Microsteps unseres Aufbaus liegt die Zahl bei dieser SynScan-Montierung bei 256. Bemerkenswerterweise besitzt die Sky-Watcher Montierung eine Unterschiedliche Getriebeübersetzung in den beiden Achsen. Während das Verhältnis für den Azimut bei 854:1 liegt, wird für die Höhenachse ein Getriebe mit 1655:1 verbaut. In der Konstruktionsweise sind hingegen nur minimale Verschiedenheiten zu erkennen. Ähnlich wie bei unserer Montierung wird ein Schneckengetriebe mit einer weiteren kleineren Übersetzung kombiniert. Hier wird jedoch auf einen Riemenantrieb gesetzt, anstatt auf einfache Zahnräder. Im Allgemeinen kann gesagt werden, dass ein solches kommerzielles System mit unserem vergleichbar ist, jedoch zusätzlich über deutlich erhöhte Präzision im Antrieb verfügt.

Hinsichtlich der Steuerung der Nachführung ist anzumerken, dass das SynScan-System zusätzlich zu den Verbindungsmöglichkeiten über WLAN und USB noch über eine Handsteuerungseinheit verfügt. So kann das Teleskop auch ohne Computer verwendet werden.

[52] s. https://skywatcher.com/product/stargate-16-synscan/

7. Fazit

Die Entwicklung und Forschung an der optimalen Teleskopmontierung ist mit Sicherheit eine sehr komplexe Thematik. Dennoch ist sie für viele insbesondere deshalb so interessant, da hier mehrere unterschiedliche Bereiche der Wissenschaft aufeinandertreffen. Die Jagd nach dem perfekten Bild einer Galaxie bereitet einer Menge an Hobbyastronomen eine große Freude. Jedoch ist nicht auf den ersten Blick zu erkennen, dass hinter jeder Aufnahme eine so enorme Menge an Wissen und Können steckt.

Rückblickend konnten wir durch unser Projekt trotz der noch ausstehenden Fertigstellung einige wertvolle Erkenntnisse gewinnen. Die Erfahrungen, die wir während der Entwicklung und Umsetzung der verschiedenen Komponenten in den Bereichen der Mechanik und Elektrotechnik gesammelt haben, waren eine besondere Bereicherung. Zusätzlich wurde unsere Leidenschaft für die Astronomie weiter gestärkt.

8. Quellen

Literaturquellen:

Hanslmeier, Arnold: Einführung in Astronomie und Astrophysik. 4. Aufl. Graz: Springer 2020

Hanslmeier, Arnold: Den Nachthimmel erleben. Sonne, Mond und Sterne – Praktische Astronomie zum Anfassen. Graz: Springer 2015

Karttunen, Hannu; Kröger, Pekka; Oja, Heikki; Poutanen, Markku; Donner, Karl J.: Fundamental Astronomy. 6. Aufl. Helsinki: Springer 2017

Internetquellen:

Iovenitti, Simone: „Star Coverage“: A simple tool to schedule an observation when FOV rotation matters. Milan: 2021 https://arxiv.org/pdf/2108.07735 [Zugriff: 2.2.2026]

Jones, Douglas W.: Control of Stepping Motors. A Tutorial. 1989. http://homepage.divms.uiowa.edu/~jones/step/ [Zugriff: 25.1.2026]

Jones, Trevor: Choosing a Star Tracker for Astrophotography (My Top Choice in 2026). 7.9.2025. https://astrobackyard.com/star-tracker-astrophotography/ [Zugriff: 9.2.2026]

Onstep: Drive Design. 28.3.2025. https://onstep.groups.io/g/main/wiki/16264 [Zugriff: 9.2.2026]

Onstep: Home. 13.10.2025. https://onstep.groups.io/g/main/wiki/3860 [Zugriff: 18.2.2026]

Onstep: Connections and Applications. 23.4.2025. https://onstep.groups.io/g/main/wiki/3863 [Zugriff: 18.2.2026]

Onstep: Smart Web Server (Ethernet and WiFi). 20.4.2025. https://onstep.groups.io/g/main/wiki/26881 [Zugriff: 18.2.2026]

Walter, Moritz: HOW ACCURATE IS MICROSTEPPING REALLY?. 29.08.2016. https://hackaday.com/2016/08/29/how-accurate-is-microstepping-really/ [Zugriff: 10.2.2026]

Posch, Maya: ONE SMALL STEP: ALL ABOUT STEPPER MOTORS. 6.1.2025. https://hackaday.com/2025/01/06/one-small-step-all-about-stepper-motors/ [Zugriff: 10.2.2026]

Sebastian: Astrofotografie – Das Problem mit der Erddrehung und Lösungsmöglichkeiten. 8.12.2023. https://zitronentour.de/astro-montierung_kamera_ueberblick/ [Zugriff: 16.2.2026]

NASA: The Scale of an Image with a Telescope. https://spacemath.gsfc.nasa.gov/weekly/10Page38.pdf [Zugriff: 16.2.2026]

Oriental motor: Stepper Motors, Stepper Motor Drivers and Controllers. https://www.orientalmotor.com/stepper-motors/index.html#stepperMotors [Zugriff: 19.2.2026]

Kellerer, Leonhard: Designing a Gearbox for a Star Tracking Mount. 9.10.2024. https://lkellr.github.io/LeosProjectArchive/posts/designing_a_gearbox_for_a-_star_tracking_mount/ [Zugriff: 24.2.2026]

californiaskys.com: Understanding Polar & GoTo Alignments. 2.9.2021. https://www.californiaskys.com/blogs/understanding-polar-goto-alignments [Zugriff: 24.2.2026]

Melsheimer, Frank: Comparing Telescope Drive Technologies. https://www.dfmengineering.com/news_telescope_gearing.html [Zugriff: 24.2.2026]

Zermelo: Levelling alt-az mounts for goto operation. 10.4.2021. https://stargazerslounge.com/topic/375443-levelling-alt-az-mounts-for-goto-operation/ [Zugriff: 24.2.2026]

Abbildungsverzeichnis:

Abbildung 1: Übersicht der Koordinatensysteme. Von Jon Peli Oleag, Übersetzt von Manlleus (ca) – Eigenes Werk, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=159627536 [Zugriff: 24.2.2026]

Abbildung 2: Äquatoriale Montierung mit Beschriftung. Von Kapege.de - Eigenes Werk, CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=1472700 [Zugriff: 24.2.2026]

Abbildung 3: Kommerzielle Dobson-Montierung. https://commons.wikimedia.org/w/index.php?curid=1088041 [Zugriff: 24.2.2026]

Abbildung 4: Star Trails um den Himmelsnordpol. Eigenaufnahme.

Abbildung 5: Unsere modifizierte Dobson Montierung. Eigenaufnahme.